Journal of Southeast University (English Edition) Vol. 34, No. 3, pp. 295 - 302 Sept. 2018 ISSN 1003—7985

Analogy-based software effort estimation using multi-objective feature selection

Chen Xiang[1,2] Lu Fengyan[1] Shen Yuxiang[1] Xie Junfeng[1] Wen Wanzhi[1]

([1] School of Computer Science and Technology, Nantong University, Nantong 226019, China)
([2]State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China)

Abstract: The feature selection in analogy-based software effort estimation (ASEE) is formulized as a multi-objective optimization problem. One objective is designed to maximize the effort estimation accuracy and the other objective is designed to minimize the number of selected features. Based on these two potential conflict objectives, a novel wrapper-based feature selection method, multi-objective feature selection for analogy-based software effort estimation (MASE), is proposed. In the empirical studies, 77 projects in Desharnais and 62 projects in Maxwell from the real world are selected as the evaluation objects and the proposed method MASE is compared with some baseline methods. Final results show that the proposed method can achieve better performance by selecting fewer features when considering MMRE (mean magnitude of relative error), MdMRE (median magnitude of relative error), PRED (0.25), and SA (standardized accuracy) performance metrics.

Key words: software effort estimation; multi-objective optimization; case-based reasoning; feature selection; empirical study

DOI: 10.3969/j.issn.1003 - 7985.2018.03.003

Software effort estimation (SEE) is one of crucial activities in software project management. If the estimation of the software effort is not accurate, it will be difficult to plan, monitor and control the project stringently. Dejaeger et al. [1] proposed different methods for SEE, such as expert judgment, parameter models, and machine learning methods. Machine learning methods are mostly adopted SEE methods in both industry and research communities. These methods include analogy-based methods, regression-based methods (such as classification and regression trees, ordinary least squares), and search-based methods (such as genetic programming, Tabu search). This paper mainly focuses on analogy-based software effort estimation (ASEE)[2-4].

ASEE is mainly motivated by case-based reasoning[5]. The fundamental assumption of ASEE is that when given a new project to estimate its effort, the most similar historical projects are selected to predict the effort of this new project by using a specific similarity function. It was first proposed by Shepperd and Schofield[6] and then the ANGEL tool was developed by them to incorporate this method. In particular, they used the Euclidean distance to find the projects similar to the target project and used a brute force algorithm to find the optimal feature subset.

The similarity function is one of the most important components in ASEE; therefore, the choice of project features has a large impact on the similarity measure. Feature selection aims to find the optimal feature subset that can achieve better performance. Nowadays, researchers have tried to apply feature selection to ASEE[6-8]. However, none of them modeled this problem as a multi-objective optimization problem. To the best of our knowledge, we first apply feature selection to ASEE using multi-objective optimization. In particular, we mainly consider two objectives. One objective is designed to minimize the number of selected features. The other objective is designed to maximize the performance of ASEE. Based on these two optimization objectives, we proposed the MASE method, which is mainly based on NSGA-Ⅱ[9]. To verify the effectiveness of our proposed method, we design and conduct a series of empirical studies. We choose Desharnais and Maxwell datasets, which include 139 projects from real world. We use MMRE, MdMRE, and PRED(0.25) metrics to evaluate the performance of ASEE. The proposed method is compared with some classical baseline methods (such as NOFS, BSFS, FSFS, SOFS). In particular, no feature selection (NOFS) does not consider feature selection. Backward-selection-based feature selection (BSFS) and forward-selection-based feature selection (FSFS)[7] denote wrapper-based feature selection methods using greedy backward selection and greedy forward selection strategies. Single objective-based feature selection (SOFS) denotes a wrapper based feature selection method using single objective optimization. We find that the proposed method MASE can select fewer features and achieve better performance. Also, the computational cost of MASE is acceptable.

Received 2018-01-12, **Revised** 2018-05-01.

Biography: Chen Xiang (1980—), male, doctor, associate professor, xchencs@ntu.edu.cn.

Foundation items: The National Natural Science Foundation of China (No. 61602267, 61202006), the Open Project of State Key Laboratory for Novel Software Technology at Nanjing University (No. KFKT2016B18).

Citation: Chen Xiang, Lu Fengyan, Shen Yuxiang, et al. Analogy-based software effort estimation using multi-objective feature selection [J]. Journal of Southeast University (English Edition), 2018, 34(3): 295 - 302. DOI: 10.3969/j.issn.1003 - 7985.2018.03.003.

In this paper, a novel wrapper-based feature selection method MASE using multi-objective optimization for ASEE is proposed. Empirical studies are designed and performed using 139 projects from the real world to verify the effectiveness of the proposed method.

1 Background and Related Work

1.1 Analogy-based software effort estimation

Analogy-based software effort estimation[2-3] is a popular approach in software effort estimation. Given target new project P to estimate the effort, it selects the most similar k projects $\{P_1, P_2, \ldots, P_k\}$ from historical projects by using a similarity function. Then, by using an adaptation strategy, it can estimate the effort $\hat{E}$ based on the effort of the selected k historical projects $\{E_1, E_2, \ldots, E_k\}$. From this process, we can find that there are three influence factors for ASEE: the similarity function, the number of most similar projects (i. e., analogies), and the adaptation strategy.

The similarity function measures the level of similarity between projects. Most researchers use the Euclidean distance or Manhattan distance as the similarity function. The number of analogies K refers to the number of the most similar projects that will be used to generate the estimation of the target project. $K = \{1, 2, 3, 4, 5\}$ can cover most of the suggested numbers. After the analogies are selected, the estimated effort for the target project is determined by the adaptation techniques[10], such as the closest analogy, the mean (or median) of the analogies, or the inverse distance weighted mean.

1.2 Feature selection for software effort estimation

Data preprocessing is a fundamental stage of the ASEE, such as feature selection, case selection, missing-data treatments[11-12]. Feature selection aims to identify and remove as many as possible irrelevant and redundant features. Here, an irrelevant feature is a feature which has little correlation with the estimated effort. A redundant feature is a feature which contains information from one or more other features. The existing feature selection methods can be classified into filter-based and wrapper-based methods. The wrapper-based methods evaluate the goodness of a feature subset using the performance of the model, and the filter-based methods use general characteristics of datasets to evaluate the quality of the feature subset.

In previous studies, researchers attempted to apply feature selection to improve the performance of ASEE. Shepperd and Schofield[6] employed an exhaustive search to find the optimal feature subset. Kirsopp et al. [7] considered hill climbing and forward sequential selection strategies. Later, Li et al. [8] proposed a hybrid feature selection method. In particular, they obtained the feature subsets to maximize mutual information in the internal stage and they searched for the best number of feature subsets in the external stage by minimizing the performance of ASEE on the training set. Moreover, feature selection is also used for other machine learning-based software effort estimation methods[13-14].

Different from previous studies, we formalize the feature selection in ASEE as a multi-objective optimization problem and then propose an MASE method.

2 The Proposed Method MASE

Our research is motivated by the idea of search-based software engineering (SBSE)[15]. The concept of SBSE was first proposed by Harman et al[15]. It has become a hot research topic in recent software engineering research. SBSE has been applied to many problems throughout the software life cycle, from requirement analysis and software design to software maintenance. The approach is promising since it can provide automated or semi-automated solutions in situations with large complex problem spaces, which have multiple competing or even conflicting objectives.

When applying feature selection to ASEE, we consider two objectives. One objective is considered from the benefit aspect and it aims to maximize estimation accuracy. The other objective is considered from the cost aspect and it aims to minimize the number of selected features.

To facilitate the subsequent description of our proposed MASE method, we first give some definitions concerning about the multi-objective optimization algorithm.

Definition 1 (Pareto dominance) Supposing that FS_i and FS_j are two feasible solutions, we call FS_i the Pareto dominance on FS_j, if and only if:

$$\mathrm{benefit}(\mathrm{FS}_i) < \mathrm{benefit}(\mathrm{FS}_j) \text{ and } \mathrm{cost}(\mathrm{FS}_i) \leqslant \mathrm{cost}(\mathrm{FS}_j)$$

or

$$\mathrm{benefit}(\mathrm{FS}_i) \leqslant \mathrm{benefit}(\mathrm{FS}_j) \text{ and } \mathrm{cost}(\mathrm{FS}_i) < \mathrm{cost}(\mathrm{FS}_j)$$

where FS_i and FS_j denote the selected feature subsets. Function benefit( ) returns the estimation accuracy of ASEE when using the given feature subset. Here, we use MMRE performance metric, which will be introduced in Section 3. The smaller the value, the better the performance. Function cost( ) returns the number of selected features.

Definition 2 (Pareto optimal solution) A feasible solution FS is a Pareto optimal solution, if and only if there is no other feasible solution FS^* which is the Pareto dominance on FS.

Definition 3 (Pareto optimal set) This set is composed of all the Pareto optimal solutions.

Definition 4 (Pareto front) The surface composed of the vectors corresponding to all the Pareto optimal solutions is called the Pareto front.

Researchers have proposed many different multi-objective algorithms (MOAs). These algorithms use evolutionary algorithms to construct the Pareto optimal set. Our proposed MASE is mainly designed based on NSGA-Ⅱ[9], which is a classical MOA.

Before introducing our method MASE in detail, we first show the coding schema of the chromosome. We can encode a feasible solution into n bit string if a dataset has n features. If the value of the i-th bit is 1, it means that the i-th feature is selected. Otherwise, if the value is 0, it means that the i-th feature is not selected. To compute the benefit of the solution, we can utilize ASEE to compute the MMRE on the training set based on the selected features.

NSGA-Ⅱ first initializes population. The population has N chromosomes and each chromosome is randomly generated. Then, it uses classical evolutionary operators to generate new chromosomes. In particular, the crossover operator will randomly choose two chromosomes according to crossover probability, perform crossover operation, and generate two new chromosomes. The mutation operator will randomly choose a chromosome according to mutation probability, perform the mutation operation, and generate one new chromosome. Later, it performs a selection operation to select high-quality chromosomes based on the Pareto dominance analysis and put these chromosomes into the new population. This process is optimized by using the fast non-dominated sorting algorithm and the concept of crowding distance. After a sufficient population evolution, it will satisfy the termination criterion and converge to stable solutions. Finally, it returns all the Pareto optimal solutions in the current population.

Here, all the Pareto optimal solutions are constructed based on the training set, and we use these selected feature subsets to compute MMRE by using ASEE on the testing set.

3 Experimental Setup

To verify the effectiveness of our proposed method, we design the following three research questions:

RQ1: Compared with the existing classical methods, does our proposed MASE have an advantage in improving ASEE performance?

RQ2: Compared with the existing classical methods, can our proposed MASE select fewer features?

RQ3: Compared with the existing classical methods, does our proposed MASE have an advantage in computational cost?

3.1 Experimental subjects

We choose two representative datasets in our experimental studies. They are the Desharnais dataset and Maxwell dataset. These two datasets are widely used in previous ASEE research[6,8,11]. The features used to estimate software effort are mainly designed based on project complexity, the development techniques used, and the number of developers and their experience.

The Desharnais dataset includes eight numerical features and 81 projects. Four out of 81 projects have been excluded due to the missing feature values. This process results in 77 complete projects for experiments. The features' names and their description can be summarized, as shown in Tab. 1. The unit of development effort of the project is 1 000 h.

Tab. 1 Features used by Desharnais dataset

Name	Description
TeamExp	Team experience measured in years
ManagerExp	Manager's experience measured in years
YearEnd	Year of completion
Transactions	Number of transactions
Entities	Number of entities
PointsAdjust	Adjusted function points
Envergure	Development environment
PointsNonAjust	Unadjusted function points

The Maxwell dataset includes 62 projects data from one of the largest commercial banks in Finland. The features of this dataset are described in Tab. 2. Most of features are categorical. Only features Time, Duration and Size are numerical. The categorical features can be further classified into ordinal features and nominal features. Nlan and T01-T15 are ordinal features. App, Har, Dba, Lfc, Source and Telonuse are nominal features. The similarity function between different projects can be found in Section 4. The unit of development effort of the project, which is from specification until delivery, is an hour.

Tab. 2 Features used by Maxwell dataset

Name	Description
Time	Time
App	Applicationtype, 1 = infServ, 2 = TransPro, 3 = CustServ, 4 = ProdCont, 5 = MIS
Har	Hardwareplatform, 1 = PC, 2 = Mainfrm, 3 = multi, 4 = mini, 5 = network
Dba	Database, 1 = relational, 2 = sequential, 3 = other, 4 = none
Lfc	User interfact, 1 = GUI, 2 = Text UI
Source	Where developed, 1 = inhouse, 2 = outsourced
Telonuse	Telon use, 0 = no, 1 = yes
Nlan	Number of different development languages used, 1, 2, 3, 4
T01	Customer participation, the nominal value of T01 to T15 is 1 = very low, 2 = low, 3 = nominal, 4 = high, 5 = very high
T02	Development environment adequacy
T03	Staff availability
T04	Standards use
T05	Methods use
T06	Tools use
T07	Software's logical complexity
T08	Requirements volatility
T09	Quality requirements
T10	Efficiency requirements
T11	Installation requirements
T12	Staff analysis skills
T13	Staff application knowledge
T14	Staff tool skills
T15	Staff team skills
Duration	Duration of project from specification until delivery measured in months
Size	Function points measured using the experience method

3.2 Performance metrics

Performance metrics are essential in evaluating the performance of SEE methods. Several performance metrics have been proposed. MMRE (mean magnitude of relative error), MdMRE (median magnitude of relative error) and PRED(0.25) are the three most used metrics in previous ASEE research[6, 8, 11].

The MMRE metric is defined as

$$\text{MMRE} = \frac{1}{n}\sum_{i=1}^{n} \text{MRE}_i \tag{1}$$

where n denotes the number of projects requiring effort estimation. MRE_i is defined as

$$\text{MRE}_i = \left|\frac{E_i - \hat{E}_i}{E_i}\right| \tag{2}$$

where E_i denotes the actual effort of the i-th project, and $\hat{E}_i$ denotes the estimated effort of the i-th project. Small MMRE value indicates a low level of estimation error.

The MdMRE metric is defined as

$$\text{MdMRE} = \text{median}(\{\text{MRE}_1, \text{MRE}_2, \dots, \text{MRE}_n\}) \tag{3}$$

This metric returns the median value of all the MREs and it is less sensitive to outliers.

The PRED(0.25) metric is defined as below:

$$\text{PRED}(0.25) = \frac{1}{n}\sum_{i=1}^{n}\begin{cases}1 & \text{MRE}_i \leqslant 0.25\\ 0 & \text{otherwise}\end{cases} \tag{4}$$

This metric returns the percentage of predictions that estimated effort falls within 25% of the actual effort.

Shepperd and MacDonell[16] reported that the MRE-based performance metrics (such as MMRE) is biased towards SEE methods since they have an underestimation problem. Therefore, we further consider SA (standardized accuracy) measure which is recently defined by Langdon et al[17]. SA is defined as

$$\text{SA} = \left(1 - \frac{\text{MAR}}{\text{MAR}_{P_0}}\right) \times 100\% \tag{5}$$

where MAR is the mean absolute residual, which is defined as $\frac{1}{n}\sum_{i=1}^{n} |E_i - \hat{E}_i|$. MAR_{P_0} is defined as $\frac{2}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{j<i} |E_i - E_j|$. Here, the large SA value indicates the low level of estimation error.

4 Experimental Design

In this section, we first introduce the baseline methods we want to compare, and then we show the parameters' value of MASE and the setting of ASEE. Finally, we illustrate the validation schemas in the two datasets.

To verify the effectiveness of the proposed method, we mainly consider the following four baseline methods: The first baseline method does not consider feature selection (i.e., use original features). We use NOFS to denote this method. The second baseline method is a wrapper-based feature selection method using single objective optimization. It uses a genetic algorithm to find the optimal feature subset, which tries to optimize the performance of ASEE. We use SOFS to denote this method. The third and fourth methods are wrapper-based feature selection methods using greedy strategies. We use BSFS and FSFS to denote these two methods. In particular, FSFS starts from an empty set, and then it sequentially selects an optimal feature by using hill climbing until the performance of ASEE cannot be further improved. BSFS starts from full features, and then it sequentially removes a feature by using hill climbing until the performance of ASEE cannot be further improved. However, these two methods have a nesting effect and it will result in a local optimum. Once a feature is selected (or removed), this feature will not be removed (or selected) in the next iteration. These methods are previously used by Kirsopp et al.[7] for ASEE.

Our method MASE is proposed based on NSGA-Ⅱ[9]. Parameters name and their value used in our empirical studies are summarized in Tab. 3.

Tab. 3 Parameter and their value for MASE

Parameter	Value
Population size	100
Maximum iteration number	100
Crossover probability	0.6
Mutation probability	0.1

Based on the selected feature subset by using different feature selection methods, such as baseline methods or MASE, we use ASEE to estimate the effort of each target project. In this paper, we set the number of analogies to be 3. We use the Euclidean distance to measure the level of similarity between two projects. Supposing that the feature vector for project a is $\{f_{a,1}, f_{a,2}, \dots, f_{a,n}\}$, and the feature vector for project b is $\{f_{b,1}, f_{b,2}, \dots, f_{b,n}\}$. The distance of project a and project b can be defined as

$$\text{sim}(a, b) = \sqrt{\sum_{i}^{n} \delta(f_{a,i}, f_{b,i})} \tag{6}$$

where $\delta(f_{a,i}, f_{b,i})$ can be defined based on the feature type (i.e., numeric, nominal, or ordinal) as below:

$$\delta(f_{a,i}, f_{b,i}) = \begin{cases}\left(\dfrac{f_{a,i} - f_{b,i}}{\max_i - \min_i}\right)^2 & \text{Numeric or ordinal}\\ 0 & \text{Nominal and } f_{a,i} = f_{b,i}\\ 1 & \text{Nominal and } f_{a,i} \neq f_{b,i}\end{cases} \tag{7}$$

where $\max_i$ and $\min_i$ denotes the maximal value and the minimal value of the i-th feature, respectively. Here, we can find that all the types of features are normalized into [0, 1]. The used adaptation technique is the mean of the analogies.

For the Desharnais dataset, we use the validation schema suggested by Mair et al.[18]. By following their proposed splitting schema, 70% of the instances are used as the training set, and the remaining 30% of the instances are used as the testing set. This splitting process is repeated three times independently. For the Maxwell dataset, we use the validation schema suggested by Sentas et al.[19]. In particular, the 50 projects completed before 1992 form the training set, and the remaining 12 projects completed from 1992 to 1993 are used as the testing set.

5 Result Analysis

5.1 Analysis for RQ1

Since there is randomness in MASE, SOFS, BSFS, and FSFS, we carry out these methods 10 times independently and generate multiple solutions. For a fair comparison, we choose the solution which achieves best performance for the training set based on the MMRE metric. The performance comparison among different feature selection methods on the Desharnais dataset is shown in Tab. 4. The best result for each metric is bolded. The results show that the MASE method can achieve better performance in most cases when considering MMRE, MdMRE, PRED(0.25) and SA metrics on each split schema. For MMRE metric, MASE can improve 26.6%, 19.3%, 22.1% and 19.3% on average when compared to NOFS, SOFS, BSFS and FSFS. For MdMRE metric, MASE can improve 28.6%, 24.6%, 20.8% and 25.2% on average when compared to NOFS, SOFS, BSFS and FSFS. For PRED(0.25) metric, MASE can improve 35.4%, 25%, 42.6% and 25% on average when compared to NOFS, SOFS, BSFS and FSFS. For SA metric, MASE also achieves the best performance.

Tab. 4 Comparison among different feature selection methods on the Desharnais dataset

Method	Split schema 1				Split schema 2				Split schema 3			
	MMRE	MdMRE	PRED(0.25)	SA	MMRE	MdMRE	PRED(0.25)	SA	MMRE	MdMRE	PRED(0.25)	SA
MASE	**0.35**	**0.28**	**0.50**	**0.59**	**0.39**	**0.34**	0.38	**0.41**	**0.39**	**0.33**	**0.42**	**0.41**
NOFS	0.55	0.45	0.29	0.48	0.46	0.47	**0.42**	0.37	0.53	0.41	0.25	0.33
SOFS	0.38	0.32	0.46	0.50	0.46	0.43	0.29	0.33	0.56	0.51	0.29	0.23
BSFS	0.42	0.43	0.29	0.51	0.52	0.42	0.33	0.29	0.51	0.35	0.29	0.31
FSFS	0.38	0.33	0.46	0.47	0.46	0.43	0.29	0.40	0.56	0.51	0.29	0.29

To further analyze the MRE values according to the testing set, we show the box plot of MRE values for each project on the Desharnais dataset in Fig.1. Our proposed method MASE has a lower median value and smaller inter-quartile range of the MRE values.

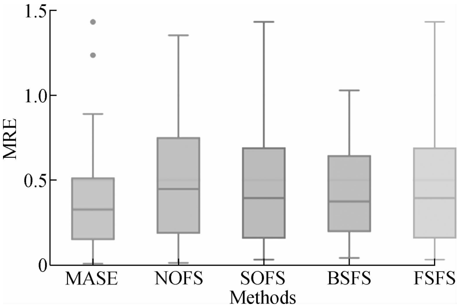

Fig. 1 The box plot of MRE values for different feature selection methods on the Desharnais dataset

The performance comparison among different feature selection methods on Maxwell dataset is shown in Tab. 5. The results show that the MASE method can also achieve better performance when considering MMRE, MdMRE, PRED(0.25) and SA metrics. For MMRE metric, MASE can improve 18.9%, 28.6%, 26.8% and 14.3% on average when compared to NOFS, SOFS, BSFS, and FSFS. For MdMRE metric, MASE can improve 21.9%, 43.2%, 28.6% and 30.6% on average when compared to NOFS, SOFS, BSFS and FSFS. For PRED (0.25) metric, MASE can improve 51.5%, 100%, 19% and 51.5% on average when compared to NOFS, SOFS, BSFS and FSFS. For SA metric, MASE also achieves the best performance.

Tab. 5 Comparison among different feature selection methods on the Maxwell dataset

Method	MMRE	MdMRE	PRED(0.25)	SA
MASE	**0.30**	**0.25**	**0.50**	**0.66**
NOFS	0.37	0.32	0.33	0.37
SOFS	0.42	0.44	0.25	0.29
BSFS	0.41	0.35	0.42	0.12
FSFS	0.35	0.36	0.33	0.52

Then, we show the box plot of MRE values for each project in the testing set on the Maxwell database in Fig. 2. From this figure, we can find that our proposed method MASE has a lower median value of the MRE values.

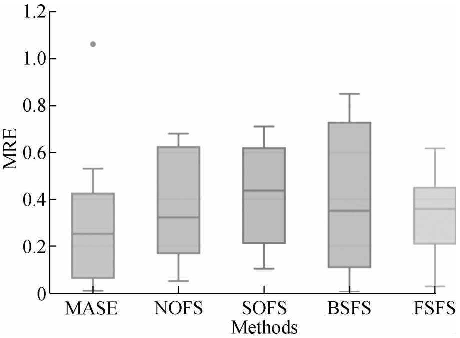

Fig. 2 The box plot of MRE values for different feature selection methods on the Maxwell dataset

5.2 Analysis for RQ2

In this subsection, we analyze the number of selected features for different feature selection methods. The result of the Desharnais dataset in three different split schemas can be found in Tab. 6 and the result of the Maxwell dataset can be found in Tab. 7.

Tab. 6 Number of selected features in three split schemas for the Desharnais dataset

Method	MASE	SOFS	FSFS	BSFS
Split schema 1	1	2	3	5
Split schema 2	1	2	2	6
Split schema 3	1	2	2	4

Tab. 7 Number of selected features for Maxwell dataset

Method	MASE	SOFS	FSFS	BSFS
Split schema 3	2	11	3	18

From these two tables, we can find that our proposed method MASE can select fewer features (i. e., 1 to 2 features) when compared with the baseline methods. To further analyze the selected features, we can find that: For the Desharnais dataset, PointsNonAjust feature is frequently selected; for the Maxwell dataset, Duration and Size features are frequently selected.

5.3 Analysis for RQ3

In this subsection, we mainly analyze the running time for different feature selection methods. All the methods are implemented by using the Weka package. These methods are run on Win 10 operation system (Intel i7-6500U CPU with 8 GB of memory). The results on the Desharnais and Maxwell datasets are shown in Tab. 8. From this table, we can find that the running time of SOFS is the highest and the running time of BSFS is the lowest. Our proposed method MASE has similar running time with FSFS and the running time of these two methods is between that of method SOFS and BSFS. In this table, the maximum execution time of MASE is 7 061 ms and this time is acceptable. Moreover, from the RQ1 and RQ2 analysis, using MASE can help to achieve better performance by selecting fewer features.

Tab. 8 Time needed to select feature subset for Desharnais and Maxwell datasets

Method	Desharnais 1	Desharnais 2	Desharnais 3	Maxwell
MASE	277	263	284	334
SOFS	327	355	368	7 061
BSFS	63	51	58	261
FSFS	252	269	267	372

5.4 Discussion

In addition to the Desharnais and Maxwell datasets used by previous studies[6,8,11], to show the generality of our empirical results, we additionally choose two datasets (i. e., NASA93[13] and usp05[20]) from the promised repository. Some datasets are not used since they have few projects or features. Then, we consider the same experimental design used for the Desharnais dataset. Due to the limit of paper length, the considered metrics and characteristics of these datasets can be found in Refs. [13, 20]. The comparison results of these two datasets can be found in Tab. 9 and Tab. 10. The best result for each metric is bolded. Final results also verify the competiveness of the proposed MASE method.

Tab. 9 Comparison among different feature selection methods on NASA93 dataset

Method	Split schema 1				Split schema 2				Split schema 3			
	MMRE	MdMRE	PRED(0.25)	SA	MMRE	MdMRE	PRED(0.25)	SA	MMRE	MdMRE	PRED(0.25)	SA
MASE	**0.87**	**0.40**	**0.36**	**0.58**	**0.53**	0.50	**0.32**	**0.58**	0.77	**0.38**	**0.36**	**0.63**
NOFS	1.19	0.58	0.18	0.56	3.72	0.78	0.11	0.27	2.88	0.86	0.14	0.31
SOFS	0.94	0.43	0.32	0.58	0.68	0.38	0.32	0.40	0.69	0.54	0.25	0.50
BSFS	3.07	0.53	0.25	0.53	2.75	0.81	0.14	0.28	3.17	0.88	0.07	0.33
FSFS	1.03	0.56	0.25	0.58	0.53	0.49	0.29	0.40	0.76	0.55	0.21	0.52

Tab. 10 Comparison among different feature selection methods on usp05 dataset

Method	Split schema 1				Split schema 2				Split schema 3			
	MMRE	MdMRE	PRED(0.25)	SA	MMRE	MdMRE	PRED(0.25)	SA	MMRE	MdMRE	PRED(0.25)	SA
MASE	**0.65**	**0.29**	0.41	**0.57**	1.00	**0.25**	**0.51**	**0.71**	**0.48**	**0.23**	**0.52**	**0.64**
NOFS	0.83	0.33	**0.43**	0.54	**0.52**	0.36	0.41	0.39	1.19	0.29	0.46	0.55
SOFS	0.88	0.35	0.39	0.53	0.85	0.36	0.48	0.40	0.62	0.25	0.49	0.57
BSFS	0.86	0.35	0.41	0.52	0.63	0.40	0.39	0.51	1.19	0.29	0.46	0.55
FSFS	0.66	0.31	0.41	**0.57**	0.57	**0.25**	0.52	0.41	0.59	0.27	0.49	0.59

5.5 Threats to validity

In this subsection, we mainly discuss the potential threats to the validity of our empirical studies. Threats to external validity are about whether the observed experimental results can be extended to other subjects or not. To guarantee the representative of our empirical subjects, we choose the Desharnais and Maxwell datasets which have been widely used by other researchers[6,8,11]. Moreover, we also choose another two datasets[13,20] to verify the generality of the empirical results. Threats to internal validity are mainly concerned with the uncontrolled internal factors that might have influence on the experimental results. To reduce this threat, we implement all the methods following the method description and use test cases to verify the correctness of our implemented MASE and other baseline methods. Moreover, we use advanced third-party libraries, such as Weka package. Threat to constructing validity is about whether the performance metrics used in the empirical studies reflect the real-world situation. We used MMRE, MdMRE, and PRED(0.25) to evaluate the performance of the prediction model. These performance metrics have been widely used in current ASEE research[6,8,11]. Moreover, we also compare our method using SA measures[16-17].

6 Conclusion

In this paper, we propose a wrapper-based feature selection method MASE using multi-objective optimization to improve the performance of ASEE. Empirical results based on projects from the real world show the competitiveness of our proposed method. In the future, we plan to extend our research in several ways. First, we want to consider more commercial projects or open-source projects in Github[21] to verify whether our conclusion can be generalized. Secondly, we want to consider other multi-objective optimization algorithms for MASE, such as SPEA[22]. Finally, we want to conduct a sensitivity analysis to analyze the influencing factors in our proposed method, such as the number of analogies.

References

[1] Dejaeger K, Verbeke W, Martens D, et al. Data mining techniques for software effort estimation: A comparative study[J]. *IEEE Transactions on Software Engineering*, 2012, **38**(2): 375 − 397.

[2] Keung J. Software development cost estimation using analogy: A review[C]//*Australian Software Engineering Conference*. Gold Cost, Australia, 2009: 327 − 336.

[3] Idri A, Amazal F, Abran A. Analogy-based software development effort estimation: A systematic mapping and review[J]. *Information and Software Technology*, 2015, **58**: 206 − 230. DOI: 10.1016/j.infsof.2014.07.013.

[4] Idri A, Hosni M, Abran A. Improved estimation of software development effort using classical and fuzzy analogy ensembles[J]. *Applied Soft Computing*, 2016, **49**: 990 − 1019. DOI: 10.1016/j.asoc.2016.08.012.

[5] Kolodner J. *Case-based reasoning*[M]. San Francisco, USA: Morgan Kaufmann Publisher, 1993.

[6] Shepperd M, Schofield C. Estimating software project effort using analogies[J]. *IEEE Transactions on Software Engineering*, 1997, **23**(11): 736 − 743. DOI: 10.1109/32.637387.

[7] Kirsopp C, Shepperd M J, Hart J. Search heuristics, case-based reasoning and software project effort prediction [C]//*Genetic and Evolutionary Computation Conference*. New York, USA, 2002: 1367 − 1374.

[8] Li Y F, Xie M, Goh T N. A study of mutual information based feature selection for case based reasoning in software cost estimation[J]. *Expert Systems with Applications*, 2009, **36**(3): 5921 − 5931. DOI: 10.1016/j.eswa.2008.07.062.

[9] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-Ⅱ [J]. *IEEE Transactions on Evolutionary Computation*, 2002, **6**(2): 182 − 197. DOI: 10.1109/4235.996017.

[10] Phannachitta P, Keung J, Monden A, et al. A stability assessment of solution adaptation techniques for analogy-based software effort estimation[J]. *Empirical Software Engineering*, 2017, **22**(1): 474 − 504. DOI: 10.1007/s10664-016-9434-8.

[11] Huang J, Li Y F, Xie M. An empirical analysis of data preprocessing for machine learning-based software cost estimation[J]. *Information and Software Technology*, 2015, **67**: 108 − 127. DOI: 10.1016/j.infsof.2015.07.004.

[12] Jing X Y, Qi F, Wu F, et al. Missing data imputation based on low-rank recovery and semi-supervised regression for software effort estimation[C]//*International Conference on Software Engineering*. Austin, USA, 2016: 607 − 618.

[13] Chen Z, Menzies T, Port D, et al. Feature subset selection can improve software cost estimation accuracy[J]. *ACM SIGSOFT Software Engineering Notes*, 2005, **30**(4): 1 − 6. DOI: 10.1145/1082983.1083171.

[14] Mendes E, Watson I, Triggs C, et al. A comparative study of cost estimation models for web hypermedia applications[J]. *Empirical Software Engineering*, 2003, **8**(2): 163 − 196. DOI: 10.1023/A:1023062629183.

[15] Harman M, Mansouri S A, Zhang Y. Search-based software engineering: Trends, techniques and applications [J]. *ACM Computing Surveys*, 2012, **45**(1):: 1 − 61. DOI: 10.1145/2379776.2379787.

[16] Shepperd M, MacDonell S. Evaluating prediction systems in software project estimation[J]. *Information and Software Technology*, 2012, **54**(8): 820 − 827. DOI: 10.1016/j.infsof.2011.12.008.

[17] Langdon W B, Dolado J, Sarro F, et al. Exact mean absolute error of baseline predictor, MARP0[J]. *Information and Software Technology*, 2016, **73**: 16 − 18. DOI: 10.1016/j.infsof.2016.01.003.

[18] Mair C, Kadoda G, Lefley M, et al. An investigation of machine learning based prediction systems[J]. *Journal of Systems and Software*, 2000, **53**(1): 23 − 29. DOI: 10.1016/s0164-1212(00)00005-4.

[19] Sentas P, Angelis L, Stamelos I, et al. Software productivity and effort prediction with ordinal regression[J]. *Information and Software Technology*, 2005, **47**(1): 17 − 29. DOI: 10.1016/j.infsof.2004.05.001.

[20] Li J Z, Ruhe G, Al-Emran A, et al. A flexible method for effort estimation by analogy[J]. *Empirical Software Engineering*, 2007, **12**(1): 65 − 106. DOI: 10.1007/s10664-006-7552-4.

[21] Qi F, Jing X Y, Zhu X, et al. Software effort estimation based on open source projects: Case study of Github[J]. *Information and Software Technology*, 2017, **92**: 145 − 157. DOI: 10.1016/j.infsof.2017.07.015.

[22] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach[J]. *IEEE Transactions on Evolutionary Computation*, 1999, **3**(4): 257 - 271. DOI: 10.1109/4235.797969.

面向基于类比的软件工作量估算的多目标特征选择方法

陈　翔[1,2]　陆凤燕[1]　沈宇翔[1]　谢隽丰[1]　文万志[1]

([1]南通大学计算机科学与技术学院,南通 226019)
([2]南京大学计算机软件新技术国家重点实验室,南京 210023)

摘要:将基于类比的软件工作量估算(ASEE)中的特征选择问题建模为多目标优化问题.其中一个优化目标是最大化工作量估算的精度,另一个优化目标是最小化选出的特征数.基于这2个潜在矛盾的优化目标,提出了一种新颖的包裹式特征选择方法MASE.在实证研究中,选择了实际项目,包括Desharnais的77个项目和Maxwell的62个项目作为评测对象,并将MASE方法与经典基准方法进行了比较.最终结果表明:基于MMRE,MdMRE,PRED(0.25)和SA评测指标,MASE方法可以选出更少的特征,且预测精度更高.

关键词:软件工作量估算;多目标优化;基于案例的推理;特征选择;实证研究

中图分类号:TP311