Expert Systems with Applications 36 (2009) 10727–10734

Contents lists available at ScienceDirect

Expert Systems with Applications

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

An early software-quality classification based on improved grey relational classifier

Nan-Hsing Chiu *

Department of Information Management, Ching Yun University, 229 Chien-Hsin Rd., Jung-Li, Taoyuan, Taiwan

ARTICLE INFO

Keywords:
Software-quality classification
Grey relational analysis
Decision support systems
Particle swarm optimization

ABSTRACT

The inherent uncertainty and incomplete information of the software development process presents particular challenges for identifying fault-prone modules and providing a preferred model early enough in a development cycle in order to guide software enhancement efforts effectively. Grey relational analysis (GRA) of grey system theory is a well known approach that is utilized for generalizing estimates under small sample and uncertain conditions. This paper examines the potential benefits for providing an early software-quality classification based on improved grey relational classifier. The particle swarm optimization (PSO) approach is adopted to explore the best fit of weights on software metrics in the GRA approach for deriving a classifier with preferred balance of misclassification rates. We have demonstrated our approach by using the data from the medical information system dataset. Empirical results show that the proposed approach provides a preferred balance of misclassification rates than the grey relational classifiers without using PSO. It also outperforms the widely used classifiers of classification and regression trees (CART) and C4.5 approaches.

1. Introduction

The software development process imposes major impacts by providing an acceptable software quality in accordance with the specification as user required. ng a high quality and reliable software product is becoming an important issue in the software industry. Prior to the system test, identifying possible faulty software components or software modules can effectively improve the testing efforts. Software-quality classification thus aims to evaluate software-quality level and to indicate software problems at an early stage. This enables software project managers to effectively allocate the finitely testing resources on high-risk modules.

During these decades, various software-quality classification techniques have been developed for targeting high-risk modules based on different software metrics (Garcia, Roman, Penalvo, & Bonilla, 2008; Khan, Shamail, Awais, & Hussain, 2006; Myrtveit, Stensrud, & Shepperd, 2005). Among the existing software-quality approaches, these models provide a decision support for software project managers in order to focus software enhancement efforts on the possible faulty software modules during development. However, most software products have incomplete information and uncertain relations between software metrics and the software quality. How to target high-risk modules and provide a preferred model with balance of misclassification rates for software project managers on the basis of finite resources is one of the most crucial challenges in the software industry.

The GRA is an important approach in the grey system theory that has become very popular in many fields (Lin, Huang, Lin, Chen, & Wang, 2008; Lu & Wevers, 2007). It is essentially believed to have captured the similarity measurements or relations between the case being estimate and historical cases for solving classification problems. However, identifying the significant factors that strongly influence dependent variable is a crucial issue for GRA approach (Huang, Chiu, & Chen, 2008). One strategy to specify significant software metrics is to build a learning mechanism whose algorithm can determine the optimal degree of importance in order to provide a preferred classifier with balance of misclassification rates for software project managers.

Particle swarm optimization (PSO) is a search algorithm that has been successfully applied to a large number of difficult optimization problems (Maitra & Chatterjee, 2008; Yisu, Knowles, Hongmei, Yizeng, & Kell, 2008). Using the improved GRA (IGRA) in order to provide a preferred model for project managers, the current study aims to examine the potential benefits of software-quality classification. PSO is adopted in the IGRA learning processes to determine suitable weights for each metric to provide a preferred model for software-quality classification. The present study begins with a discussion of the related works on the software-quality classification in Section 2. The construction processes of the software-quality classification model using the IGRA is presented in Section 3. Section 4 tackles the description of the databases used in this empirical study. Section 5 compares the classification abilities of IGRA, GRA without improvement and other software-quality

* Tel.: +886 3 4581196x7321; fax: +886 3 4683904.
E-mail address: nhchiu@cyu.edu.tw

0957-4174/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2009.02.064

classification approaches found in the literature. Finally, Section 6 summarizes the present study and outlines further works on relevant topics.

2. Related work

During these decades, a large number of software-quality classification models have been proposed in the literature. These approaches include classification and regression trees (CART; Khoshgoftaar, Allen, Jones, & Hudepohl, 2000), C4.5 decision trees (Khoshgoftaar & Seliya, 2004), and many other methods (Garcia et al., 2008; Khan et al., 2006; Myrtveit et al., 2005). These software-quality classification models often classify software modules whether they are likely to be fault prone (*fp*) or not fault prone (*nfp*), and evaluate their classification abilities in terms of type I error (*TIE*) rate and type II error (*TIIE*) rate. The *TIE* rate is an error of misclassifying actual *nfp* module to *fp* module, whereas the *TIIE* rate is an error of misclassifying actual *fp* module to *nfp* module. The targeting of enhancement efforts will be inefficient if the *TIE* rate is high. On the other hand, many *fp* modules will not be detected for enhancements when the *TIIE* rate is very high.

A software-quality classification model may not be useful to guide software improvement efforts if either type of misclassification rate is high. From a practical point of view, a proper software-quality classification model often required a good balance between these two types of misclassifications (Khoshgoftaar & Seliya, 2003). In addition to good balance between these two misclassification rates, the characteristics of software metrics also play a crucial role on the predictive performances (Khoshgoftaar & Seliya, 2004). One of the most crucial challenges in the classification of software quality is targeting high-risk modules accurately from the vague and incomplete information. It is extremely difficult to accurately identify relevant software metrics that strongly influence software quality because the degree of influence is imprecise in nature (Rodriguez, Ruiz, Cuadrado-Gallego, Aguilar-Ruiz, & Garre, 2007).

In 1982, Deng initiated the grey system theory that is applicable to study uncertainties in system models, analyze relations between systems, establish models, and make forecasts and decisions (Deng, 1982). In the grey system theory, the GRA is essentially believed to have captured the similarity measurements or relations between case being estimate and historical cases in order to make forecasts (Huang et al., 2008). The grey relational grade (GRG) of GRA shows the degree of relationship between the case being estimated and each case in the historical dataset. For a classification model based on GRA approach, the retrieval of the best comparative case for forecasting relies on a relational measure that takes into account of the GRG between pairs of cases.

GRA is an effective approach of analyzing the relationship between cases with less data that can overcome the disadvantages of statistical method (Chang, Tsai, & Chen, 2003). It compensates the shortcoming of statistical regression in the case when experiments are ambiguous or when the experimental method cannot be carried out exactly (Lin & Ho, 2003). Huang et al. (2008) examines the potentials of the software effort estimation using GRA for the problem of the software projects with incomplete information, less data and uncertain relations (Huang et al., 2008). Their experimental results show that the software effort estimation using the GRA approach presents more precise estimates over the results using the case-based reasoning, classification and regression trees, and artificial neural networks methods.

A grey relational classifier is crucial to identify the most prominent independent variable that leads to the effective case retrieval. However, most of the GRA utilized the equal weight on each metric that do not use significant weight for each independent variable (Chang, Wen, Chen, & Chang, 2000; Jiang, Tasi, & Wang, 2002). Chang, Hung, Tzeng, and Lin (2004) improved this issue by deriving weights based on the experience of experts in the GRA (Chang et al., 2004), but the determination of the weights highly depended on their subjective opinions. This problem is mainly due to the difficulty in determining the appropriate weights to the independent variable that may contain irrelevant or redundant information for software-quality classification. Generally, more important independent variables should be assigned larger weights in compared with less important ones in order to determine the significant software metrics for software-quality classification.

PSO algorithm is a search method that is an easy implementation for solving optimization problems. In 1995, Kennedy and Eberhart introduced it as a novel evolutionary computation technique with a capability to optimize complex numerical problems (Kennedy & Eberhart, 1995). The concept of PSO was motivated from the simulation of birds flocking that mimics the social behavior of flying birds and their means of information exchange with companions to explore the optimized solution. Each potential solution is seen as a particle with a certain velocity and flies through the problem space, and adjusts its flight toward the globally optimal solution according to its own flying experience and its companions' flying experience. This approach explores optimal solution in complex search spaces through the interaction of individuals in a population of particles. It does not need complex operators such as crossovers and mutations in compared with the evolutionary computation technique of genetic algorithms. Instead of genetic algorithms, it only requires simple mathematical operators and computationally inexpensive runtime (Wang, Yang, Teng, Xia, & Jensen, 2007).

This approach has been successfully applied to a large number of combinatorial optimization problems and different applications. Medeiros and Schirru (2008) proposed a machine vision-based roundness measuring method that applies the PSO to compute the roundness of a circular work piece in the quality control and inspection problems (Medeiros & Schirru, 2008). Their results reveal that the PSO method effectively solved the problems and outperforms genetic algorithms method in both accuracy and efficiency. Sha and Hsu (2008) applied PSO for solving the open job shop problem. Their computational results show that the PSO found many good solutions of the unsolved problems. Wang et al. (2007) propose a feature selection strategy based on the rough sets and PSO which compares with a genetic algorithm approach (Wang et al., 2007). Their results show that PSO is efficient for rough set-based feature selection.

PSO is particularly attractive for investigating suitable weights in that particle swarms may discover the best weight as they fly within the problem space. Liu, Zhu, Zhang, and Wang (2004) propose a weighted fuzzy rule-based system, in which the weights of rules are estimated using the PSO algorithm (Liu et al., 2004). The empirical result shows that weighted fuzzy rules can lead to better fuzzy system in compared with non-weighted fuzzy rules. Zhang, Zhang, Lok, and Lyu (2007) applied the PSO algorithm to train the weights of feed forward neural network (Zhang et al., 2007). Their experimental results show that the proposed approach is better than the gradient descending method in convergent speed and convergent accuracy.

The performance of the relational measures and the weights of the each software complexity metric are critical to the reasoning process for GRA. PSO is an useful method that can explore good solutions for solving many complexity problems, especially for discovering weights. However, little research has attempted to adopt the PSO approach to optimize weights in the GRA for software-quality classification. Thus, this paper aims to investigate the effects of the software-quality classification based on an IGRA approach that integrates the GRA with PSO for providing a practical