The Journal of Systems and Software 83 (2010) 868–882

Contents lists available at ScienceDirect

The Journal of Systems and Software

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

A symbolic fault-prediction model based on multiobjective particle swarm optimization

André B. de Carvalho, Aurora Pozo *, Silvia Regina Vergilio

Computer Science Department, Federal University of Paraná (UFPR), CP 19:081, CEP: 81531-970 Curitiba, Brazil

ARTICLE INFO

Article history:
Received 20 May 2009
Received in revised form 17 November 2009
Accepted 22 December 2009
Available online 22 January 2010

Keywords:
Fault prediction
Particle swarm optimization
Multiobjective
Rule learning algorithm

ABSTRACT

In the literature the fault-proneness of classes or methods has been used to devise strategies for reducing testing costs and efforts. In general, fault-proneness is predicted through a set of design metrics and, most recently, by using Machine Learning (ML) techniques. However, some ML techniques cannot deal with unbalanced data, characteristic very common of the fault datasets and, their produced results are not easily interpreted by most programmers and testers. Considering these facts, this paper introduces a novel fault-prediction approach based on Multiobjective Particle Swarm Optimization (MOPSO). Exploring Pareto dominance concepts, the approach generates a model composed by rules with specific properties. These rules can be used as an unordered classifier, and because of this, they are more intuitive and comprehensible. Two experiments were accomplished, considering, respectively, fault-proneness of classes and methods. The results show interesting relationships between the studied metrics and fault prediction. In addition to this, the performance of the introduced MOPSO approach is compared with other ML algorithms by using several measures including the area under the ROC curve, which is a relevant criterion to deal with unbalanced data.

1. Introduction

Software plays a crucial role in all areas and human activities. Due to this, tasks for quality assurance, such as software testing, can be considered fundamental. However, test is a very expensive task, which consumes a lot of effort and time in the software development. To reduce testing costs, a good strategy is to focus testing efforts on some modules of the software, such as classes or methods, with high likelihood of being faulty. On the other hand, guidelines to design modules with low fault-proneness are also desired. These issues can be tackled by extracting knowledge from historical data and constructing a model for fault-proneness prediction to be applied in future projects.

Most prediction models found in the literature are statistical, and are generally based on a set of Object-Oriented (OO) design metrics, for instance: the set proposed by Chidamber and Kemerer (CK-metric suite) (Chidamber and Kemerer, 1994) and a set of McCabe, Halstead and line count metrics. However, the prediction accuracy of these models was unsatisfactory to gain confidence. Because of this, most recent works have introduced the use of Machine Learning (ML) techniques (Prez-Miana and Gras, 2006; Fenton et al., 2007; Lounis and Ait-Mehedine, 2004; Pai and Dugan, 2007; Thwin and Quah, 2005; Zhou and Leung, 2006; Elish and Elish, 2008; Zhan and Reformat, 2007; Vandecruys et al., 2008; Gondra, 2008; Menzies et al., 2007; Lessmann et al., 2008; Xing et al., 2005). Those works present promising results, however, until now, the ML-based works show two main disadvantages: most prediction models are not easily interpreted by the programmers and testers; and most approaches require a pre-process step in order to obtain a balanced dataset. Other point is that the results reported by almost works only focus in some aspects of the prediction models, like accuracy.

To reduce the mentioned disadvantages, this paper introduces a novel approach for fault prediction. This approach is based on Multiobjective Particle Swarm Optimization (MOPSO) algorithms to induce rules from the data. Rules are one of the most used forms to represent knowledge extracted from a data set. This is because of their simplicity, intuitive aspect, modularity, and can be obtained directly from a data set (Fawcett, 2001). Furthermore, the induced rules are symbolic, i.e., they are represented by a logical expression and can be easily interpreted by humans. And, as mentioned in Lessmann et al. (2008), comprehensible models reveal the nature of detected relationships and help to improve the overall understanding of software failures and their sources.

Besides, as far as we know, this is the first time that fault prediction is addressed by a multiobjective algorithm. This fact allows to obtain rules with specific properties by exploring Pareto dominance concepts (Knowles et al., 2006) and all rules are generated

* Corresponding author.
E-mail addresses: andrebc@inf.ufpr.br (A.B. de Carvalho), aurora@inf.ufpr.br (A. Pozo), silvia@inf.ufpr.br (S.R. Vergilio).

0164-1212/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2009.12.023

in a single run. Furthermore, all rules are generated independently and there is no pre-defined order to analyze then. As consequence, the constructed model is more intuitive and can be used by most software engineers and testers.

To validate our ideas we present a MOPSO algorithm, named MOPSO-N. One important feature of MOPSO-N is its ability to handle both numerical and discrete attributes. So, it does not need a pre-processing discretization step and the original data obtained through the software process could be directly used in the rule learning process. As remarked by Lessmann et al. (2008), a wide range of discretization algorithm has been proposed in the data mining literature and the selection of one is not an easy task, moreover, its application would multiply the computational effort of the study. On the other hand, for each numerical attribute, MOPSO-N tries to discover the best range of values for certain class and to obtain better rule sets.

MOPSO-N was first introduced in Carvalho et al. (2008) with the goal of predicting fault-proneness of classes. Now, in the present paper, MOPSO-N is described in detail and the results presented previously are summarized. In addition to this, another context is addressed: fault-proneness of methods. The generated rules are analyzed to identify interesting relationships between the metrics and fault-proneness. The use of these relationships are herein illustrated to reduce testing efforts, and to design methods and classes with low likelihood of being faulty, a use not explored before.

In addition to the initial results presented in our previous work, the MOPSO-N approach is evaluated in a larger number of case studies. In addition to this, the MOPSO-N approach is compared with other ML-based approaches reported in the literature. A broader set of measures is used to compare the results, including the Area Under the ROC curve (AUC) (Ferri et al., 2002; Rakotomamonjy, 2004).

AUC has been traditionally used in medical diagnosis since the 1970s, and in recent years has been used increasingly in ML and data mining research. AUC is considered a relevant criterion to deal with unbalanced data, misclassification costs and noisy data because AUC has an attractive property: it is insensitive to changes in class distribution. Recent works (Lessmann et al., 2008; Menzies et al., 2007) point out the existence of a large number of unbalanced data sets in the fault-prediction context and the importance of considering AUC. Then, the study herein presented considers the AUC and the results in the conducted experiments are analyzed through the state-of-the art hypothesis testing methods (Demšar, 2006) showing that the MOPSO-N approach induces accurate and comprehensible fault-prediction models, even dealing with unbalanced data.

The remainder of this paper is organized as follows: Section 2 gives a brief survey on related works on fault-prediction. Section 3 presents the basic MOPSO concepts and, Section 4 describes the MOPSO-N algorithm in details. Section 5 shows how the MOPSO-N approach was evaluated, the used evaluation measures, software metrics, and adopted methodology. The experimental results are illustrated and discussed in Sections 6 and 7, in the context of, respectively, methods and classes. Section 8 illustrates the use of MOPSO-N to guide test strategies. Finally, Section 9 presents the threats to validity and Section 10 concludes the paper and discusses future works.

2. Related work

Software testing is a fundamental software engineering activity for quality assurance that is traditionally very expensive. Research and development in mining repositories can be used to tackle this issue. Mining repository means to extract knowledge from past project data. This knowledge can be a model for fault-proneness prediction, and then, this model can be applied to future projects. Based on this concept, a wide range of statistical models have been developed and applied to predict faults in software. Among them, the most related works are that ones which address the fault-proneness prediction of classes, methods or modules (Thwin and Quah, 2005; Alshayeb and Li, 2003; Basili et al., 1996; Briand et al., 1998, 2000).

In this context, the Metrics Suite for Object-Oriented Design, known as CK, of Chindamber and Kemerer (Chidamber and Kemerer, 1994) has been largely used. These metrics are defined by properties such as complexity, inheritance, coupling and cohesion. For instance, Basili et al. (1996) investigated the impact of the CK suite on the prediction of fault-prone classes using logistic regression.

To improve performance, Machine Learning (ML) techniques have been explored. They can automatically acquire knowledge from fault data and generate prediction models that are really discovered. Guo et al. (2004) proposed random forest technique, a well-known ensemble technique for accuracy improvement, to predict fault-proneness of software systems. They applied this technique on NASA software fault data sets.[1] These same data sets were used by Pai and Dugan (2007) that explored Bayesian networks. Other works that explored Neural and Bayesian networks are: (Prez-Miana and Gras, 2006; Fenton et al., 2007; Lounis and Ait-Mehedine, 2004; Pai and Dugan, 2007; Thwin and Quah, 2005; Zhou and Leung, 2006). Other technique that has been successfully explored is Support Vector Machine (SVM) (Elish and Elish, 2008; Xing et al., 2005). A recent work (Gondra, 2008) presents results of a comparative study on the effectiveness of Neural Networks and SVM to predict fault-prone modules. In such study SVM presented a superior performance. In Singh et al. (2009), other SVM work in fault-prone is presented, this work presents a study to validate the results of the SVM and find the relation between OO metrics and fault-proneness models. All these works present promising results but models based on BN, Neural networks, SVM and random forest are not symbolic, and then, the results have a difficult interpretation. In the same vein of the mentioned works, other innovative approaches, such as ant colony, have been investigated (Zhan and Reformat, 2007; Vandecruys et al., 2008). But they need a very difficult pre-processing step to deal with unbalanced data.

In addition to the prediction models, Menzies et al. (2007) and Lessmann et al. (2008) propose baselines to conduct comparisons between different techniques when dealing with the NASA MDP data sets. The first work discourages the use of accuracy for unbalanced data sets and suggests the use of receiver-operator (ROC) curves. The second one presents a large benchmarking of 22 different classification models. This work uses AUC as the main measure to compare the techniques. Both works conclude that the classification results of most methods do not differ significantly and the selection of a classification model should be based on several additional criteria like computational efficiency, ease of use, and comprehensibility.

Analyzing the main related works on prediction of fault-proneness of classes or methods, it can be observed that most of them consider CK-suite and the fault data sets of NASA. The main disadvantages of the existent ML approaches can be summarized as: the produced models are difficult to understand; some of them require a pre-processing step; most approaches cannot deal with unbalanced data, a common characteristic of the fault data sets; they only focus some restrict evaluation aspects, usually accuracy; and they address the fault-prediction problem as a single-objective, that is, with a simple solution, not considering different properties of the possible solutions. To overcome these limitations we

[1] Available on http://www.mdp.ivv.nasa.gov/.