ABSTRACT

Software quality metrics are methods that quantitatively determine the extent to which software process, product or project possess a certain quality attribute. They are used to measure software engineering products (design, source code, etc.), processes (analysis, design, coding, testing, etc.) and professionals (efficiency or productivity of an individual designer). Techniques and methods that identify and predict faults using these quality metrics has gained wide acceptance in the past few decades as they have direct impact on the software product’s time, cost and scope.

The high usage of software system poses high quality demand from users, which results in increased software complexity. Fault prediction is a strategy to identify faulty parts of a program, so that the testing process can concentrate only on those regions. This will improve the testing process and indirectly help to reduce development life cycle risks, project risks, resource and infrastructure costs.

Usage of software metrics to evaluate the quality of software design has attracted software industries as they help to assess large software system quickly at low cost. Increased usage of Object Oriented (OO) paradigm has been envisaged in recent software products, which has increased the need for new quality metrics to be devised. Traditional metrics, like CK (Chidamber and Kemerer) metrics and MOOD (Metrics for Object Oriented Design) metrics, do
not consider object oriented paradigms like inheritance, encapsulation and passing of message and therefore do not perform well with fault prediction.

The increase in software complexity and software size increase the demand for new metrics to identify flaws in the design and code of software system. This research work analyzes and proposes metrics for measuring class complexities that can be used as a medium to identify design defects. The primary objective of the research work is to combine software engineering, software metrics and data mining classification concepts to predict faulty modules in object oriented software.

Both traditional metrics and extended metrics are used during fault proneness prediction. The traditional metrics are considered as simple metrics, CK metrics, MOOD metrics and program complexity metrics. The simple metrics considered are LOC (Total number of lines), BR (Number of methods), NOP (Total Number of Unique Operators), NOPE (Total Number of Unique Operands), RE (Readability with Comment percentage) and VO (Volume). The CK Metric suite consists of six metrics, namely, Weighted Methods per Class, Depth of Inheritance Tree, Number of children, coupling between object classes, Response for a Class and Lack of Cohesion in Methods. The MOOD consists of Method hiding factor, Attribute hiding factor, Method inheritance factor, Attribute inheritance factor, Polymorphism factor and Coupling factor metrics. The Cyclomatic Complexity and Fan-In Fan-Out Complexity (Henry's and Kafura's) were used as program complexity metrics.
Four extended metrics based on flow of information, friend class/function, inheritance and cohesion are also used. The four extended metrics are Class Method Flow Complexity Measure (CMFCM), Friend Class Complexity Metric (FCCM), Class Complexity from Inheritance (CCI) and Class Complexity from Cohesion Measure (CCCM).

Thus a total of 24 quality metrics, which are grouped into three categories, are used during prediction. The three categories are Existing Quality Metrics Feature set (EQMF), Extended Quality Metrics Feature set (ExQMF) and Fused Quality Metrics Feature set. In order to solve the problem of the curse of dimensionality, feature reduction techniques are used. Two feature reduction techniques, namely, feature selection and dimensionality reduction are used. The feature fusion and selection algorithm is based on genetic algorithm, while the dimensionality problem was solved using Sensitivity Analysis using Sensitivity Casual Index (SCI). Normalization of feature sets was also performed.

The study proposes the use of machine learning algorithm to analyze the performance of the proposed metrics in predicting faulty modules in object oriented programs. The prediction model is designed as an ensemble-binary classification task, where a module is predicted as faulty (complex) or not-faulty (normal). The present study uses homogeneous and heterogeneous ensemble classifiers for fault prediction which combines three classifiers, namely, Feed Forward Back Propagation Artificial Neural Network (BPNN), Support Vector Machine (SVM) and K-Nearest Neighbour (KNN). Homogeneous classifiers are
considered as, those models having the same classification methodology but different feature vectors, while heterogeneous classifiers are models with each classifier using a different classification methodology.

Input feature vector is created by combining twenty existing metrics, namely, simple metrics (6), Mood Metrics (6), CK Metrics (6) and Program Complexity metrics (PCM) (2), along with the four proposed metrics. Four selection methods, namely, Sequential Selection, Random Selection with No Replacement, Selection with Bagging and Selection with Boosting, were used during design of homogeneous classifiers. The result of the classifiers in the ensemble model is combined using an enhanced majority voting scheme.

During performance evaluation, four classification performance metrics, namely, accuracy, precision, recall and F-measure and speed of fault detection, were used during evaluation. The experimental results proved that the inclusion of the proposed metrics has increased the efficiency of the ensemble classification model with all performance parameters. Further, the application of ensembling concept has also improved the performance of the fault prediction. This is evident from the high accuracy obtained when compared with single classifier models.