Chapter 1

INTRODUCTION

Software quality assessment is growing in this era due to the demand of object oriented modularity. Estimation of software quality is a major concern today. Development practitioners normally use software metrics and fault data from a previous system release or similar software project to construct quality classification or fault prediction models. It can be categorized by different accuracy measurement of software model like F-Measure, Odd Ratio and Power.

1.1 Introduction

To foresee about the great nature of the projects or the product, module is an important term. Software quality estimation is most important research areas in the field software engineering that attracts the experts. A lot of methods have already been developed in the area of software quality estimation. The main objective of software quality estimation is to discover the tasks within software which are prone to errors so as to minimize the maintenance cost of the software and these precautions of errors will improve the quality of software. Software maintenance is time and resource consuming activity. Tracking the defect as early as possible in a software life cycle will not only improve the effective cost but will also help to achieve the customers’ satisfaction and reliability of the software developed. Subsequent to examining a few exploration woks around there, we go over with two principle issues. First is the commotion and second which is the principle issue is the parameter on which you can sort the projects for the improvement \(^1\).

There are for the most part two sorts of clamor in the class of information quality and programming parameters. The first is worry with the mislabeled programming modules, brought on by programming engineers neglecting to
distinguish, neglecting to report, or basically disregarding existing programming flaws \(^2\). Uprooting such boisterous occasions can altogether enhance the execution of adjusted programming quality-estimation models. Another principle test is that, in certifiable programming ventures, we have to discover the parameters on that we can assess the quality. So there is the expectation system on the premise we can learn and anticipate. So there is the need of managed and unsupervised methodology, which we will examine in the ensuing subsections \(^3\).

1.1.1 Clustering

In the category of unsupervised learning methods we generally use clustering techniques, which is a good way for finding the alikeness measure. Algorithms which are used for Clustering can group the software modules according to the values of their software metrics. The assumptions for software-engineering are that for same properties will have similar software measurements and so will likely to form clusters or they are in the same group. Similarly, non-similar modules will likely group together.

Instead of inspecting and labeling software modules one at a time, the expert can inspect and label a given cluster as a whole; he or she can assign all the modules in the cluster the same quality label. Such a strategy eases the tediousness of the labeling task, which is compounded when modules are numerous. For each cluster the modules referenced by the clustering algorithm arrange in such manner that they follow some similar patterns. When the labels are available either in terms of project or in terms of software modules, we can apply the predicting parameters like we use in COCOMO Model for improving the software quality. The clustering algorithm approaches is also a niche for addressing noise detection and removal for software quality classification. It can be a preprocessing step in analyzing training data quality before building a classifier.
1.1.2 Software Metrics

To establish a proper Software Measurement Methods, a clear understanding of the appropriate data collection, analysis and reporting requirements are needed. The starting point is the identification of the audiences for measurement and their unique needs. We also identify the key measures, source of data, analyze and interpret the metrics, report the information parameters, available testing tools and how we apply the ongoing process on the available tools. So our measurement framework accomplishes mainly four things 1) Metrics 2) Analysis 3) Final Qualification 4) Improvement Analysis as shown in figure 1.1. There are several researchers which has devoted much research to developing and studying effective software metrics that characterize a software system’s complexity. Those researchers and practitioners have analyzed and provide some study with several software complexity metrics, most of them are used for program code for expressing software complexity. These software metrics have been used widely and successfully used though some issues related to their effectiveness remain open research problems.

Figure 1.1: Software Measurement Framework
Chapter 1: Introduction

1.1.3 Evolutionary computation

Evolutionary computation is getting to be regular in the arrangement of troublesome, real-world issues in industry, pharmaceutical, and software modulation. This dissertation surveys a portion of the down to earth points of interest to utilizing developmental calculations as contrasted and great strategies of advancement or counterfeit consciousness. Particular favorable circumstances incorporate the adaptability of the strategies, and in addition the capacity to self-adjust the quest for ideal arrangements on the fly. As desktop PCs increment in pace, the use of transformative calculations will get to be normal.

1.2 Motivation

Our main aim to find better object oriented model suite. For this we use clustering, chi square test and particle swarm optimization (PSO). Depending on the size we first determine a realistic range for number of clusters $K$. Such an approach is important for software projects where resources are relatively limited and finite. Means each program module represent a single entity, on which we apply the unsupervised learning or clustering. In our study, the provided data statistics include object oriented features based module, as well as the size of each cluster. Of course, in the interactive process, the clustering analyst can suggest other useful information to the expert, who in turn may ask for additional data properties or object oriented features like class, object, inheritance etc. We note that the cluster representatives are not immediately available for similarity-based clustering approaches, which further supports our choice of centroid- based clustering.

In our study, the clustering analyst is a mean to centroid then we apply Chi Square test. A Chi Square test measures whether there is a significant difference between the effects of two categorical independent variables on a
Chapter 1: Introduction

categorical dependent variable. The interactive process can be expedited if the clustering analyst and the software engineering expert are the same person; however, one has to be careful in staying away from introducing bias into the analysis. PSO is used to optimize the result to the maximum threshold.

1.3 Objectives

In this dissertation the main objectives of our approach are-

1. To utilize all the important object oriented parameter which will be refined by the software quantifiers so the modularity results are enhanced.
2. To develop the modules based on the object oriented properties instead of coupling measures so it takes less time.
3. To develop an algorithm which is capable of estimating and testing the modularity by applying random particle swarm optimization for testing the optimized value and the obtained results and found to be improved.

1.4 Approach

The following approaches are used in our dissertation:

1. Programming modularity’s has been calculated.
2. Then which program is qualified for the further testing and optimization we have used chi-square test. The chi-square test is a statistical test that can be used to determine whether observed frequencies are significantly different from expected frequencies. Based on the outcome of the chi-square test we will either reject or fail to reject the null hypothesis.
3. Then selected modules are qualified for the further software metric analysis.
4. Then for the further data classification we have used software metrics F-measure, PO and OR. These measures are the most commonly used
“single number” measures in Information Retrieval. It provides the significance of the qualified data in that class of origin.

5. When you build a model for a classification problem you almost always want to look at the accuracy of that model as the number of correct predictions from all predictions made. This is the classification accuracy. But the classification accuracy alone is typically not enough information to make this decision.

6. Then PSO is applied to check the optimal values which will check the final class modularity rank.

1.5 Problem Statement

By Shtern [4] there are a few issues which are confronting by the scientists. The greater parts of the analysts perform their work on a little arrangement of information or a product framework. The pertinence is dependably a question mark when we apply the same procedure for the enormous database. Accordingly, it is unrealistic to sum up the assessment results to other programming frameworks. The software defects are considered to be among the few direct measurements of software processes and products.

Information Quality characterizes as "wellness for reason" [5][6]. Since this reason for existing is subjective and essential to consider, information quality's attributes or measurements are subjective and can't be evaluated free of the general population who utilize the information. Redman [7] records more than 10 measurements. Redman mentioned the fact that information quality measurement rely upon the client's perspective of the information, directing towards an explanation behind the absence of information quality measurements. Redman classified his measurements into four gatherings:

- Dimensions related to the data model,
- Dimensions related to the data values,
- Dimensions related to data presentation and
Chapter 1: Introduction

- Dimensions related to information technology.

Manago and Kodrato \[8\] expressed that clamor is available when an information base does not genuinely mirror. They are demonstrating that the reasons for commotion lead investigators to fabricate off base models. By definition, clamor is incorrect data, absence of data or problematic data. The term questionable data is intriguing; further, the data is not inaccurate, but rather problematic. It is a determinant variable of Object Oriented parameter.

In the proposed work, software quality is explored on the object oriented parameters to analyze their impact on software quality and using optimization technique to determine optimum choice of the software. Total forecast of good projects or module is absent for the clients, which give better measured quality. There is need of more slants towards the program arranged parameter; this is the principle inspiration of our work. Based on the findings, effective ways and means are suggested to improve software quality and thus enhance the overall quality of Projects.

1.6 Scope of the work

Investigation of Data quality is a vital issue which has been tended to as information warehousing, information mining and data frameworks. An endeavor to enhance characterization exactness by pre-bunching did not succeed. In any case, blunder rates inside of bunches from preparing sets were emphatically related with mistake rates inside of the same groups on the test sets. This marvel could maybe be utilized to create certainty levels for expectations. In this dissertation we proposed an efficient algorithm for predicting good programs based on object oriented clustering with Chi-Square test classification for goodness-of-fit test. In this approach we first select the object oriented programs based on user module requirement. We select those java programs on the basis of clustering of object oriented parameters like class, object, inheritance and dynamic behavior. Then we find the precision
Chapter 1: Introduction

based on Chi-Square test, if precision \((p)\) is greater than 0.5 then we accept the hypothesis otherwise we reject it. Then we check the program capability under three parameters; first is F-measure (FM), second is odds ratio (OR) and third is power (PO). Based on the three parameters we can find the better programming approach.

1.7 Limitations

The main limitations of this dissertation are it may confuse in the selection of dynamic memory allocations and objects and second it works well with java programs but may be difficult with other programming languages as the semantics of all the programming language is different.

1.8 Structure of Dissertation

The dissertation is organized into six chapters. Chapter 2 provides the background necessary for the rest of the dissertation and the Literature Survey. It reviews related concepts in the context of the work presented in this dissertation. Chapter 3 provides the background necessary for the rest of the dissertation and the analysis. It reviews related concepts in the context of the work presented in this dissertation. Chapter 4 discusses the software and the algorithm some detailed issues about the proposed work will be presented, including the examples and algorithms with result discussions. Chapter 5 describes the experiment and its result part as well as the description of software system snapshot developed in the research for better understanding of the approach. Chapter 6 describes the conclusion drawn from this dissertation as well as some insights into future research on this topic. Finally references are given.