Lifecycle-Based Software Defect Prediction Technology

Xiangshu Peng[1], Zhiming Ma[1](✉), Ning Zhang[2], Yaoxian Huang[3], and Menglin Qi[1]

[1] School of Computer Science and Technology, Xinjiang Normal University, Urumchi 830054, China
2298882664@qq.com

[2] School of Electricity and Computer, Jilin JianZhu University, Changchun 130000, China

[3] School of Computing, Harbin Finance University, Harbin 150000, China

Abstract. In order to improve the efficiency and quality of software testing, aiming at various factors affecting software reliability, how to find defective modules and optimize them in the early stage of software development has become an urgent problem to be solved, This paper introduces the software defect prediction technology based on life cycle. According to the measurement elements affecting software reliability, relevant internal indicators and design defects, find the defect module, lock it in advance, adopt machine learning technology and reasonably allocate limited resources, which is conducive to evaluate the software design scheme, optimize the design strategy, reduce design changes and improve the software operation process, It plays a role in cost evaluation, resource management, scheme determination and quality prediction in software management. It is hoped to provide some theoretical support and practical reference for the development of software defect prediction.

Keywords: Software defect predict · Software metric · Software lifecycle · Introduction to the study

1 Introduction

With the popularization of information technology in all walks of life, the number and complexity of all kinds of computer systems and software are growing rapidly, its development technology is constantly updated and developed, the operating environment is becoming more and more complex and severe, and if the software is defective in the running process, it may lead to serious consequences, and the reliability of software operation has become an important guarantee for the sustainable development of various industries. Software reliability mainly refers to the prediction of software defects. In each stage of the whole software life cycle (mainly including market customer demand analysis stage, content design stage, coding implementation stage and operation and maintenance stage), defects are not evenly or randomly distributed in the software; Aiming at the prediction technology of defect distribution, this paper uses machine learning technology to construct software defect prediction model, in order to improve software testing efficiency and ensure software reliability.

2 Analysis of Software Prediction Technology

2.1 Definition of Software Prediction Technology

Software prediction is not complex in the actual development process. It mainly needs to run the analysis of a certain program under the specified conditions, find out whether there are certain design errors in the program in the actual development process, test the software quality accordingly, and evaluate whether the program meets the design requirements in the actual operation process, Therefore, software prediction technology will become more important in the actual development process. The update, improvement and innovation of software prediction technology play a very important role in improving the quality of software. The purpose of software prediction is: (1) to analyze and observe defects to improve the quality of software. (2) Verify whether certain requirements are met in the actual development process. (3) In the actual development process, the forecaster should have a certain confidence in the software quality itself.

2.2 Principles of Software Prediction Technology

(1) the test shows that there are defects; When testing, the scope of testing should be comprehensive. (2) Early detection and intervention (low cost and high efficiency). (3) Defect clustering: 80% defects are concentrated in 10% modules. (4) In the actual development process, when a detection method fails to find relevant defects in time, the test cases should be replaced in time. (5) There must be a certain relationship between testing activities and testing background. (6) No shortcomings are fallacies (useful systems) in the actual development process. When testing software, we should analyze its advantages and disadvantages, find shortcomings and problems, and make relevant improvement and innovation in time.

2.3 Main Workflow of Software Prediction Technology

Carry out reasonable planning and control of the software in the actual test process, reasonably plan the test scope, classify and innovate the test plan, and analyze and count the relevant data of the project to be tested, such as the input and output documents, product description, main functions of the software in the actual development process, It includes reasonable analysis and control of the output documents, integration and statistical analysis of all resources (human, material, financial, etc.) invested in the whole test plan, and certain evaluation and prediction of relevant risks. In addition, the development environment of the project should be considered when selecting the training set to further promote software quality and prediction efficiency.

3 Software Reliability Measurement

To predict software defects, the first is to select the measurement element of software reliability, which should focus on simplicity and practicality, face each stage of software development, run through the whole development process (demand analysis, design, implementation and testing), and focus on reliability. Software reliability is mainly composed of three activities: software error proofing, finding and troubleshooting faults, and measuring software to achieve the best reliability; The measurement of software reliability is mainly analyzed from the technical measurement, As shown in Table 1 is Technical metrics for each stage of the software lifecycle. And then subdivided from the four stages of software development process. (1) Requirements reliability measurement: customer requirements include software technology, quality and various functions. The key lies in the preparation of requirements description. Finding and correcting requirements errors will consume huge human and material resources and reduce the quality of software. Therefore, requirements description requires structure, stability, high quality, easy understanding and easy application. (2) Design and implement reliability measurement: Based on the complexity and scale of the module, analyze the code structure and function system, and determine the appropriate development language and format, so as to find and predict the defects and errors of that module. The evaluation of the combination of scale and complexity is more effective. (3) Test reliability measurement: in order to ensure the integrity of software development function, the evaluation of test plan is adopted to reduce the lack and error of expected function.

Table 1. Technical metrics for each stage of the software lifecycle

Software lifecycle phase	Technical metrics
The requirements analysis phase	1: The team's level of development skills and experience 2: The correctness and rationality of demand decomposition
Content design phase	1: The average number of fan-ins of the module 2: The average number of fan-outs of the module 3: The average ring complexity of the module 4: The average coupling of the module
The coding implementation phase	1: The length of the module code 2: Develop language types 3: Coding rules

4 Lifecycle-Based Software Defect Prediction Technology

4.1 Existing Research

Life cycle software defect prediction technology is a very important technology, which plays a very important role in ensuring and improving software quality. Software defect prediction technology based on life cycle belongs to an important type of this technology. Learn from and learn other types of software defect prediction technology, analyze and improve it, so as to better promote the development of technology. Document [6] proposed a software defect prediction method based on software development cycle and PCA-BP fuzzy neural network. The results show that compared with the prediction method based on BP neural network, the PCA-BP Neural Network prediction method combined with principal component analysis method has faster convergence speed and higher prediction accuracy. Document [1] proposed a software defect prediction model construction method for cost sensitive classification. The decision tree classifier for cost sensitive classification is constructed by bagging multiple random sampling training samples, and then the software module defect prediction is carried out after voting integration. The results show that this method significantly reduces the false positive rate on the premise of ensuring the defect prediction rate, AUC and F values are better than the existing methods. Document [2] proposed a cross project software defect prediction method based on multi-source data, obtained multi-source similar project data, and realized the prediction model based on Naive Bayesian algorithm. The results show that the performance of this method is better than the traditional WP method.

4.2 Experimental Analysis

The first is the collection of test data, collecting measurement data of 16 flight control software, totaling 613 modules. The sixteen software serves as training samples for the neural network, and the last eight software serves as the software to be predicted. Using PCA-BP network model and BP network model, defect prediction is made for several stages of software life cycle. We compare and analyze the convergence speed of the neural network and the accuracy of software defect prediction. The convergence speed of the neural network is reflected by the mean of the sum of the errors between the expected output and the real output of the network. The formula is as follows (1).

$$e = \frac{1}{m}\sum_{j=1}^{n}\left(Y_j - Y_{j(\text{prediction})}\right)^2 \tag{1}$$

Formula: m denotes the number of sample software, Y_j represents the actual value of the jth software defect density, $Y_{j(\text{prediction})}$ represents the predicted value of the jth software defect density, and the e represents the average of the sum of squared errors between the predicted value and the actual value. The training error line graph is shown in Fig. 1 and Fig. 2. The horizontal coordinate represents the number of training sessions and the vertical coordinate represents the average of the sum of squares of training errors. The results show that the convergence speed of PCA-BP network is not different from that of BP network in the stage of requirement analysis, but PCA-BP network converges faster than BP network during the training of coding phase.

4.3 Index Analysis

Two indicators are designed: number of developers (NOD) and coding quality lock (locq). Bayesian network is used to analyze the relationship between each index and error trend. The results show that RFC (class response), LOC (number of lines of code) and locq are the most effective metrics, and CBO (coupling between objects), WMC (weighting method per class) and LCOM (lack of method cohesion) have little effect on defect tendency. Two initial feature subsets are obtained by lg based filtering method

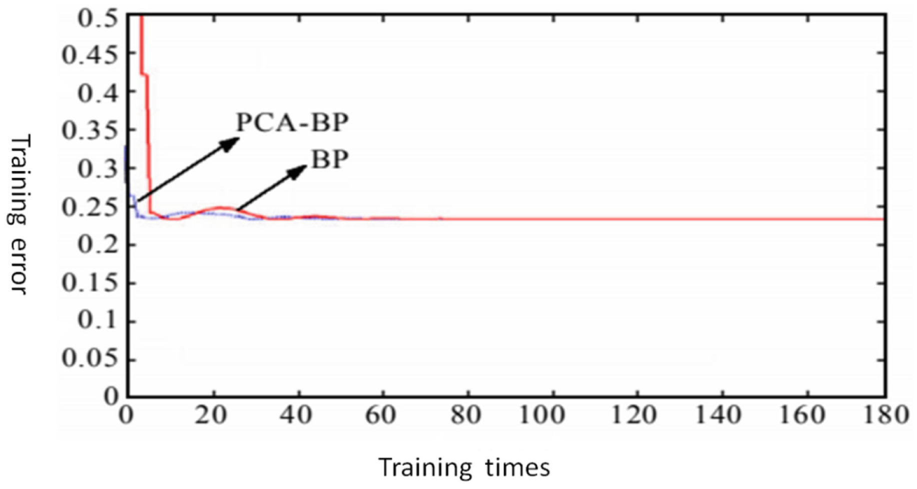

Fig. 1. Neural network training error curve in demand analysis stage

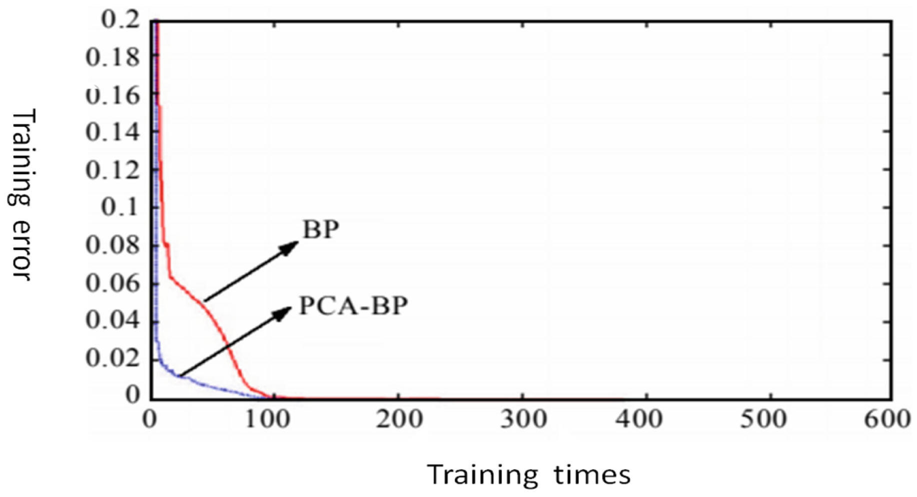

Fig. 2. Training error curve of neural network in coding stage

(MIM) and correlation coefficient based filtering method (SBS); Then, these subsets are combined into global optimization. (GA) defs (differential evolution case feature selection) or PSO (particle warming optimization) are applied to the combination subset respectively, and an effective feature subset is obtained by taking variance as the stop criterion.

4.4 Method Analysis

The filtering method is to calculate the chi square test (CS) or information gain (Ig) or correlation coefficient score or Pearson correlation value of each measurement element, and select the feature according to these values. The selection process is independent of subsequent learners. The packaging method takes the learner's performance as the evaluation criterion for selecting the best feature subset. The encapsulation method takes the learner's performance as the evaluation function, so the performance of the encapsulation method is better than the filtering method. However, the time cost of encapsulation method is large, while the filtering method has nothing to do with learners and the cost is small, so the generalization ability is stronger than encapsulation method. Fesch (feature selection using clustering of hybrid data) is used to solve cross-project defect prediction problems. The method is divided into two phases. In the first stage, the original feature set is clustered by the Peak Density Clustering (DPC) method. In the second stage, local feature density (LDF), feature distribution similarity (SFD), and feature class correlation (rel) are used to deal with cluster defects. Defect life cycle is a whole process from the creation of a defect to the end of the defect. During this process, defects may undergo different processing scenarios depending on development and product strategies: continue to repair it).

Scenario 1: Confirm Error Resolution Test Submission Defect [new], Develop Confirm Defect [open], Develop Resolve Defect [fixed], Test Regression Defect, Close Defect [closed]; Scenario 2: Validation fails, defects still exist, Test submission defects, development confirmation defects [open], development resolution defects [fixed], testing regression defects, assigning development to resolve [reopen]; Scenario 3: The closing defect reappears and the tester reopens the closing defect; Scenario 4: Develop delayed processing test submission defects [new], develop validation defects [open], delayed processing (publishing); Scenario 5: Refuse to process test submission defects [new], develop validation defects [open], reject (reject). To understand software defects, first understand the concept of software defects, second understand the detailed characteristics of software defects, and finally understand the properties of software defects. The next higher level is learning to use tools to manage software defects.

5 Software Defect Prediction Technology Defect Report Analysis

High quality defect reports can help developers quickly locate problems and fix them. Convenient for testers to count, analyze, track and manage defects; It is an important communication tool between testers and developers. Therefore, if our testers find a defect during the test, they need to record the defect and submit a defect report. It mainly includes recording software defects, then classifying and summarizing the defects, then tracking software defects, and finally analyzing and summarizing the defects. Important components of a defect report are number (defect ID), Title (summary), date of detection, module to which the defect belongs (subject), release of detection, and assignment of tasks. Defect states include; (1) Describe the status of the defect at this time: when the tester finds a defect and submits it to the development manager. (2) Open: The state after the development manager acknowledges and accepts the defect (if the development manager finds that it is not a defect, it will reject it and the defect state will be rejected). (3) Fix: The developer receives the bug and the status of the fix. (4) Off: Testers perform tests to repair defects and verify that the status is tested (if the tester verifies that a defect has not been repaired, that is, if the test fails, the defect status will be reopened and the developer will continue to repair it). (5) Defect severity.

6 Conclusion

This paper mainly introduces the concept of software defect prediction technology based on life cycle, the selection of measurement elements and the early prediction method of software reliability using PCA-BP fuzzy neural network, which plays a great role in improving and ensuring software quality and sustainable development, but there are still some problems in the research of software defect prediction technology, We also need to constantly analyze and summarize experience in the event, so as to promote the continuous development and progress of the industry. I hope this paper can provide certain theoretical support and practical reference for relevant staff.

References

1. Yong, L., Zhiqiu, H., Bingwu, F., Yong, W.: Software defect prediction method for cost sensitive classification. Comput. Sci. Explor. **8**(12), 1442–1451 (2014)
2. Li, Y., Liu, Z., Zhang, H.: Overview of cross project software defect prediction methods. Comput. Technol. Dev. **30**(03), 98–103 + 121 (2020)
3. Yong, L., Zhiqiu, H., Yong, W., Bingwu, F.: Cross project software defect prediction based on multi-source data. J. Jilin Univ. (Engineering Edition) **46**(06), 2034–2041 (2016)
4. Li, Y.: Software defect prediction combined with under sampling and integration. Comput. Appl. **34**(08), 2291–2294 + 2310 (2014)
5. Li, Y., Liu, Z., Zhang, H.: Overview of integrated classification algorithms for unbalanced data. Comput. Appl. Res. **31**(05), 1287–1291 (2014)
6. Wu Chao, X., Jianping, C.L.: Software defect prediction technology based on life cycle. Comput. Eng. Des. **30**(12), 2956–2959 (2009)
7. Tao, M.: Research on feature selection method for software defect prediction. Jilin University (2020)
8. Lina, G., Shujuan, J., Li, J.: Research progress of software defect prediction technology. J. Softw. **30**(10), 3090–3114 (2019)
9. Cai, L., Fan, Y., Meng, Y., Xia, X.: Research progress of real-time software defect prediction. J. Softw. **30**(05), 1288–1307 (2019)
10. Shen, P.: Research on software defect prediction method based on machine learning. Southwest University (2019)
11. Wang, T.: Research on software defect prediction based on measurement. Wuhan University (2018)
12. Li, L.: Research on cross version software defect prediction technology. Nanjing University of Aeronautics and Astronautics (2018)
13. Zou, J.: Research and application of feature selection method for software defect data. China University of Petroleum (East China) (2017)
14. Lu, G.: Research on software defect prediction technology based on deep learning. Nanjing University of Aeronautics and Astronautics (2017)
15. Cheng, M.: Research on some key technologies of software defect prediction. Wuhan University (2016)