ATLANTIS PRESS

Advances in Engineering Research, volume 141

5th International Conference on Mechatronics, Materials, Chemistry and Computer Engineering (ICMMCCE 2017)

Software Defect Prediction Model Research for Network and Cloud Software Development

Yejuan Yang [1,a,*]

[1] Department of Information Engineering, Jianghai Polytechnic College, Yangzijiang South Road No. 5, Yangzhou, China

[a] 914118925@qq.com

*corresponding author

Keywords: Oriented Network and Cloud Development; Software Defect Prediction; Multi Source Project; Naive Bayes Algorithm.

Abstract. With the process of software development and application basing on network and cloud, the change of software development requires new software defect prediction method for these kinds of software development, which can solve the problems of the traditional software defect prediction method based on target project, such as the same predict background and higher cost of defect tagging. A new software defect prediction method based on multi source data oriented network and cloud development process is proposed. This method selects the predictive candidates from multi-source projects which have similar characteristics as objective projects, and then guides the training data selection by the software modules, finishes the prediction based on Naive Bayesian algorithm. Finally through algorithm experiment this method is proved superior to the traditional WP prediction model.

1. Introduction

With the application of information technology in various fields of technology and the arrival of the Internet era, software has become an indispensable technical support for people's production and life. Because the software development process engineering, and with the development of application, the complexity of the software is more and more high, so the problem of software products will inevitably encounter various, usually these defects may be hidden in the normal operation in normal operation will not show it, but in certain circumstances can damage the function of the software, the normal use and good experience to affect users, even cause serious consequences to the user, has brought huge losses. For example, in 2000 the radiotherapy machine in the Cancer Research Center of USA has killed 8 people and injured 20, due to software defects and errors in the calculation about the amount of radiation [1]. According to the research report pointed out that the United States NIST the United States in 2007 due to software defects caused by the economic losses reached $180 billion [2]. The existence of software defect has brought great trouble to the software enterprise, which has affected the development of software engineering.

In order to make the software company to produce high quality products, to avoid or prevent software defects caused by adverse effects on the application system, the researchers in the field of software engineering has done a lot of research, reason and process perspectives arising from the software defects, to explore the solution of software defects[1-4], including the occurrence and trend through the development and application of various software defect management tools to effectively manage software defects, these research results help software enterprises and developers, testers should try to avoid the defects in the software engineering development process, to a great extent. Foreign researchers in basic research work, proposed a variety of software defect prediction model, including: the scale of software defect prediction technology, complexity of software defect prediction technology, forecast technology, based on machine learning metrics based on multidimensional software defect prediction technology based on technology. But with the development of network, software development and application of cloud based software

development method, the current is gradually from the enterprise development, enterprise cooperation development into a one-stop cloud software development platform, all the tools, services and software development process can be obtained from the software development of cloud platform. So that the division of labor between the software industry is more refined, the software industry chain lengthening, so that some enterprises are more focused on their business content, the other part of the enterprise is more focused on the development and implementation of the algorithm. This fine division of labor makes software defects as a quality problem in the process of software engineering development.

Due to the development of members and their development background there is a great difference for the network cloud software project development process, software development team, at different stages of the project belong to different developers, the software defect has the characteristics of different markers and expression. This paper proposes a network oriented transport development cross project data software defect prediction method (multi-projects filter predictor for network development, MFP-ND), the method based on the different developers in the development process according to the development stage of their complete data, software defects model suitable for target project selection from their project experience, prediction and evaluation of software defects, to effectively improve the prediction performance.

2. Software Defect Feature Tags and Transformations for Network Cloud Development

The essence of software defect prediction and analysis process is a kind of intelligent judgment software through machine learning, through a certain quantity of similar defects had marked the software module of sample learning, so as to establish a prediction model for a character defect, and according to the model to determine whether the current module contains defects. Therefore, before the forecast, acquisition and label learning sample data must first solve software defects, namely detailed software testing on the existing software source code, obtain the defects of each module, and then the defects measurement and marking. On this basis, to extract a variety of code level statistics related to software module defect metrics (e.g., code length, code branch number and code complexity) as the feature data description module of software defects, so the code is transformed into a set of defect feature vector. For different stages of software projects from different developers, each developer will build different models according to their own development experience. This model needs to be based on the existing project data, so it is necessary to deal with the developer's project and its software defect metrics.

All software items that need to be handled in software defect prediction are labeled as a set of data: $S=\{(x_{ij}, y_i)\}$, i=1, 2,... N indicates the number of software modules in the project; j=1, 2, ... M, said the software metric attribute; x_{ij} represents a i software module of the j attribute; y_i said the i defect marking software module, y_i =1 said the defects of the module, the module y_i =0 said no defects in all projects. For modeling the source project can be marked as S_m, $MS=\{S_{m(k)}\}$ is multi-source the project can be marked as a set of data, including: k=1, 2, ... K said the number of source, project, and software module of each source in the project have been completed attribute measure and defect dimension; objectives of the project can be marked as S_t, which is NULL defect marking software module, software defect prediction classification goal is for each software module of the S_t defect in the tag value.

For the software project completed by different developers, developers should be able to cooperate in order to improve the prediction and analysis of software defects, but every developers because of its own development experience, software defect prediction model construction will be different, and thus give the cooperation exchange inconvenience. Due to the similarity of the data source of the software defect source, the data can be exchanged and transformed by mapping the source data. Different developers (enterprise) project set data marked as $S_{(z)}=\{(x_{(z)ij}, y_{(z)i})\}$, z=1, 2,... , Z indicates the different developer numbers for project cooperation. A developers and B developers for data exchange can be achieved through the transfer function $x_{(a)ij}=F(x_{(b)ij})$. Different developers (enterprises) of the multi source project marked as $MS_{(z)}=\{S_{(z)m(k)}\}$, z indicates that the

project cooperation of different developer numbers.

3. Software Defect Prediction Modelling for Network Cloud Development

For different software project developers, each developer in the project developers have the similar data model based on prediction is established, the need for developers to the project project defect prediction data from the data acquisition model is suitable for T_D St *MS* set, T_D set for T_D. Acquisition of existing software module defects in *MS* prediction the data can be divided into two steps: first, according to the project feature selection and St alpha source project similar from *MS*, a candidate of portfolio Sp; then extract data from S_p T_D. Select the appropriate algorithm for T_D prediction model is constructed based on forecasting project defects, finally realize the goal of the project *St* software defect classification module in the forecast.

(1) Source selection algorithm for software project defect prediction

Prediction of cross project software defects, between source and target attribute measure distribution project project there is a certain correlation [5], so the maximum minimum value, and the mean and standard deviation of 4 data distribution data to define the feature vector of the $\boldsymbol{C}$ project. To help achieve the feature similarity sort source and target selection of software project project the distribution of software module is a module attribute, expressed as $\boldsymbol{C} = \{c_{max}, c_{min}, c_{mean}, c_{std}\}$, the distribution characteristics of each project data attribute values can be used in $c(F_m(S))$to represent the feature vector of the *S* project, which can be defined as follows:

$$\boldsymbol{C}(S) = \{c_{max}(F_{m1}(S)), \ldots, c_{std}(F_{m1}(S)), \ldots, c_{max}(F_{mM}(S)), \ldots, c_{std}(F_{mM}(S))\} \quad (1)$$

Where *M* is the number of measurement attributes; $\boldsymbol{C}(S)$ dimension is $p = 4M$, the $\boldsymbol{C}(S)$ software module is not included in the defect information, thus can also be used to define characteristics of the goal of the project.

Because the cosine distance is not sensitive to the absolute value of the eigenvector, the difference is more in the direction of the vector. In order to modify the standard of measurement which may exist between different projects, the cosine distance is used to calculate the feature similarity of the project. For item feature vector $\boldsymbol{C}_1 = \{\boldsymbol{C}_{1(1)}, \boldsymbol{C}_{1(2)}, ..., \boldsymbol{C}_{1(p)}\}$ and $\boldsymbol{C}_2 = \{\boldsymbol{C}_{2(1)}, \boldsymbol{C}_{2(2)}, ..., \boldsymbol{C}_{2(p)}\}$, the similarity is calculated as follows:

$$\boldsymbol{F}_S(\boldsymbol{C}_1, \boldsymbol{C}_2) = \frac{\sum_{i=1}^{p} (\boldsymbol{C}_{1(i)} \times \boldsymbol{C}_{2(i)})}{\sqrt{\sum_{i=1}^{p} (\boldsymbol{C}_{1(i)})^2} \times \sqrt{\sum_{i=1}^{p} (\boldsymbol{C}_{2(i)})^2}} \quad (2)$$

Selection of software defect prediction project source data, feature vector of the $\boldsymbol{C}(S_{m(k)})$ multiple sources will have the project developers feature vector $\boldsymbol{C}(S_t)$ in turn with the objectives of the project to calculate the similarity between $F_S(\boldsymbol{C}(S_{m(k)}), \boldsymbol{C}(S_t))$, and the similarity of the row order selection sort on the former source project S_p alpha.

(2) Source selection algorithm for software module defect prediction

The software project data source selection is completed further extracting module of the project defects in the forecast data, usually by the nearest neighbor algorithm from multiple projects in software modules as basic data model[6], i.e. the software module objectives of the project guide from the source data selection. Because the number of software modules in the project the number of multi-source far greater than the target project software module, which contains more information of software defect prediction samples, will calculate the number of defects increases, resulting in more accord with the characteristics of defect target project data.

This paper uses multi-source project guide the way to realize the software module selection algorithm, specific operation is as follows: the candidate projects with the goal of the project consists of a mix operation data set $S_{mix}=S_p \cup S_t$; then K-means clustering algorithm is adopted to form a number of software modules in cluster $ID\text{-}M=K_m(S_{mix}, \beta)$; each cluster cluster ID_t and ID_p each candidate project module marking target module project the candidate item; retrieval module, when the candidate projects and project module module containing the same cluster （$S_p(ID_p)=S_t$

(ID_t), the source items in the S_p module as the model data of T_D=$\sum S_p(ID_p)$. The final software defect prediction data source selection has narrowed the range of options of software modules at the project level, so it can effectively avoid the problem of low efficiency of algorithm. In the face of a large number of project data source.

(3) MFP-ND software defect prediction algorithm

Because the software defect data attribute of the conditional independence assumption does not affect the performance of the final model [7], so this paper adopts Naive Bayesian Bayes theorem and property prediction algorithm based on conditional independence assumption to realize software defect oriented network cloud development prediction model.

The objective of the naive Bayes algorithm is based on the attribute values of software modules (x_1, x_2, ..., x_N), get the module maximum probability value of the existence of defects in $P(y \mid x_1, x_2, ..., x_N)$, because the NB algorithm assumes that the value of a given object is independent of each other, so there is a maximum probability formula[8]:

$$P(y \mid x_1, x_2, , x_N) = \frac{\prod_{i=1}^{N} P(x_i \mid y)P(y)}{P(x_1, x_2, , x_N)} \tag{3}$$

I is a software module when the missing attribute value, the $P(x_i \mid y)$ is set to 1, said the software module does not participate in the training model. In addition, the attribute values for the software module of continuous value can represent the probability distribution of the continuous attribute class $P(x_i \mid y)$ software module by Gauss the distribution, the distribution of the mean and variance of 2 sigma two parameters, so the defect marker y and the Xi property of the class conditional probability is equal to

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[\frac{(x_i - \mu)^2}{2\sigma^2}\right] \tag{4}$$

The type (4) is a continuous probability density function of the Xi property from a fixed value probability of 0, can be used to calculate the conditional probability $\varepsilon P(x_i \mid y)$ which is the probability of Xi between the interval [x_i, x_i+ε], where ε is defective instead. (y=1) and defect free (y=0) class two are needed to calculate the constant multiplication factor, prediction probability will not affect the final output value.

4. Software Defect Prediction Experiment

According to the requirement from the PROMISE database selected object oriented 26 versions of 7 open-source software projects the software defects measurement data for the experimental analysis, the characteristics of each project software module includes 20 measurement attributes and the number of defects of [9]. The source tag attributes data and the target data are collected from different companies, belong to different data distribution, so it can be used for the cooperation and development reflect the situation between different software developers, with engineering data of a company to predict the transfer of software belongs to network oriented cloud development defect engineering data of another company in the forecast.

Table 1. Experimental Data Information

Item	Modules	Defect-prone	Defect-prone/%	Item	Modules	Defect-prone	Defect-prone/%
ant1.3	125	20	16	log4j1.0	135	34	25
ant1.4	178	40	22	log4j1.1	109	37	34
ant1.5	293	32	11	log4j1.2	205	189	92
ant1.6	351	92	26	lucene2.0	195	91	47
ant1.7	745	166	22	lucene2.2	247	144	58
camel1.0	339	13	4	lucene2.4	340	203	60
camel1.2	608	216	36	poi1.5	237	141	59
camel1.4	872	145	17	poi2.0	314	37	12
camel1.6	965	188	19	poi2.5	385	248	64
ivy1.1	111	63	57	poi3.0	442	281	64
ivy1.4	241	16	7	synapse1.0	157	16	10
ivy2.0	352	40	11	synapse1.1	222	60	27
jedit3.2	272	90	33	synapse1.2	256	86	34
jedit4.0	306	75	25	velocity1.4	196	147	75
jedit4.1	312	79	25	velocity1.5	214	142	66
jedit4.2	367	48	13	velocity1.6	229	78	34
jedit4.3	492	11	2				

(1) Performance measurement

In this paper, using precision, recall rate is widely used in the field of software defect prediction of the false positive rate and F-measure index to evaluate the prediction of defect prediction performance of the model. The goal of the project results usually can be divided into 4 types: TP (True positive) said the software defects are predicted to have defects (False negative; FN) said the defective modules are predicted to be free of defects; FP (False positive) said the defect free module is predicted to have defects; TN (True negative) said the performance index of the defect free module is predicted to be free of defects. Then the defect prediction, the recall rate is correctly predicted as the ratio of the defect model the number of defective module number and real RC=TP/ (TP, +FN); the accuracy is correctly predicted to have defects and the number of module is predicted to be defective rate module number, P=TP/ (TP+FP); sham The positive rate is incorrectly predicted as the ratio of the number of defective modules to the number of true defect free modules, pf= FP/ (FP+TN). For defect prediction model, the recall rate and precision is good. But the recall rate and precision often influence each other, cannot get high numerical. Therefore, the available F-measure will recall rate and precision of comprehensive evaluation. F-measure is defined as the average precision and recall rate of harmonic defect prediction performance, F_m =[(α+1)*RC*P]/(RC+α*P) type (0, + infinity), usually take a =1. RC, P, PF and Fm values are between 0 and 1. When the PF value is low, the other 3 of the value is higher, the better the performance prediction model.

(2) Experimental results

In the MFP-ND software defect prediction algorithm experiment, respectively, the α ={10, 20,... For N/10}, the number of candidates, where N is the number of items in the software module. Multi level data selection in order to verify the actual performance of β =10. The algorithm of MFP-ND model, in addition to the proposed NB algorithm, the experiment also chose 2 representative algorithms (LR algorithm, support vector regression logic SVM machine algorithm was verified).

Table 2 is a comparison of median model index different experimental settings of the 3 algorithms, the 5 algorithms of the recall rate, accuracy and comprehensive F-measure overall approached were better than the latter, especially the decision models in the practice of the recall rate advantage is more obvious. The results show that the choice to feature defect project more software module similar to the software project data from similar characteristics, verify the validity of this algorithm. The results from the recall rate, MFP-ND model of NB algorithm is better than the other 2 algorithms based on the α =30 when the recall rate of the most stable, the accuracy of

results except SVM method worse, the accuracy rate of the other 2 algorithms, F-measure results show that the MFP-ND model of the NB algorithm and BN algorithm performance is better than the other algorithms. The median values from table 2 we can see the results based on NB algorithm In the alpha =30 comprehensive performance is better than the other algorithms.

Table 2. The Median Value of the Model under Different Indexes

Set index	NB	LR	SVM	Set index	NB	LR	SVM
HFP-10 (R)	0.841	0.371	0.041	HFP-20 (F)	0.405	0.349	0.048
HFP-10 (P)	0.276	0.435	0.279	HFP-30 (R)	0.853	0.304	0.012
HFP-10 (F)	0.392	0.316	0.074	HFP-30 (P)	0.320	0.465	0.105
HFP-20 (R)	0.863	0.450	0.026	HFP-30 (F)	0.472	0.365	0.021
HFP-20 (P)	0.324	0.511	0.198				

5. Conclusion

In this paper, a software defect prediction model based on multi-source data is proposed. The model and algorithm of model data obtained using classification strategy, Naive Bayesian prediction model is constructed. Through defect prediction experiments show that this method can replace the WP method for practice based on other classification strategy data selection method can also be used in multi-source data network project of Yunkai sharing and collaboration based prediction library provides a new way for him. Software defect data collaboration to build a unified future sharing.

Acknowledgements

Our thanks to Advanced Training Item for Professional Leaders of Higher Vocational College (2016GRFX008) for supporting us to finish this research.

References

[1] Maganioti, A.E., Chrissanthi, H.D., Charalabos, P.C., Andreas, R.D., George, P.N. and Christos, C.N. (2010) Cointegration of Event-Related Potential (ERP) Signals in Experiments with Different Electromagnetic Field (EMF) Conditions. Health, 2, 400-406.

[1] Okutan A, Yildz OT. Software defect prediction using Bayesian networks. Empirical Software Engineering, 2014, 19 (1):154-181.

[2] Shepperd M, Bowes D, Hall T. Researcher bias: The use of machine learning in software defect prediction. IEEE Transactions on Software Engineering, 2014, 40 (6):603-616.

[3] Zhang H, Nelson A, Menzies T. On the value of learning from defect dense components for software defect prediction. Proceedings of the 6th International Conference on Predictive Models in Software Engineering, 2010: 1-9.

[4] Ma Ying, Luo Guangchun, Zeng Xue, et al. Transfer learning for cross-company software defect prediction. Information and Software Technology, 2012, 54 (3): 248-256.

[5] Herbold S. Training data selection for cross-project defect prediction. Proceedings of the 9th International Conference on Predictive Models in Software Engineering, Baltimore, 2013:1-10.

[6] Peters F, Menzies T, Marcus A. Better cross company defect prediction. Proceedings of the Tenth International Workshop on Mining Software Repositories, San Francisco, CA, USA, 2013: 409-418.

[7] Turhan B, Bener A. Analysis of Naive Bayes. Assumptions on software fault data: an empirical study. Data & Knowledge Engineering, 2009, 68(2): 278-290.

[8] Pang-Ning T, Steinbach M, Kumar V. Introduction to Data Mining. New York: Pearson, 2005:

231-236.

[9] Jureczko M, Spinellis D. Using object-oriented design metrics to predict software defects. Fifth International Conference on Dependability of Computer Systems DepCoS, Poland, 2010: 69-81.

[10] Menzies T, Butcher A, Cok D, et al. Local versus global lessons for defect prediction and effort estimation. IEEE Transactions on Software Engineering, 2013, 39(6):822-834.

[11] Turhan B, Menzies T, Bener A B, et al. On the relative value of cross-company and within-company data for defect prediction. Empirical Software Engineering, 2009, 14(5): 540-578.

[12] He Z, Shu F, Yang Y, et al. An investigation on the feasibility of cross-project defect prediction. Automated Software Engineering, 2012, 19(2):167-199.