An Ensemble Approach of Simple Regression Models to Cross-Project Fault Prediction

Satoshi Uchigaki
Graduate School of Information Science
Nara Institute of Science and Technology
Nara, Japan
satoshi-u@is.naist.jp

Shinji Uchida
Department of Information Engineering
Nara National College of Technology
Nara, Japan
uchida@info.nara-k.ac.jp

Koji Toda
Department of Computer Science and Engineering
Fukuoka Institute of Technology
Fukuoka, Japan
toda@fit.ac.jp

Akito Monden
Graduate School of Information Science
Nara Institute of Science and Technology
Nara, Japan
akito-m@is.naist.jp

Abstract- In software development, prediction of fault-prone modules is an important challenge for effective software testing. However, high prediction accuracy may not be achieved in cross-project prediction, since there is a large difference in distribution of predictor variables between the base project (for building prediction model) and the target project (for applying prediction model.) In this paper we propose an prediction technique called "an ensemble of simple regression models" to improve the prediction accuracy of cross-project prediction. The proposed method uses weighted sum of outputs of simple (e.g. 1-predictor variable) logistic regression models to improve the generalization ability of logistic models. To evaluate the performance of the proposed method, we conducted 132 combinations of cross-project prediction using datasets of 12 projects from NASA IV&V Facility Metrics Data Program. As a result, the proposed method outperformed conventional logistic regression models in terms of AUC of the Alberg diagram.

Keywords: fault-prone module prediction, product metrics, empirical study

I. INTRODUCTION

Prediction of fault-prone modules is an important challenge for effective testing and/or software inspection [1]. Various multivariate modeling techniques, which are applicable to fault-prone module prediction, have been proposed, including multivariate logistic-regression (MLR) model [4], linear discriminant analysis [5], neural network model [6], etc. These models are constructed from a fit dataset, which contains product metrics and fault data of modules of a past software project [2] [3].

However, for a new development project and/or an enhancement project that had not recorded data of past releases, high prediction accuracy cannot be expected. It is because defect prediction works well if models are trained with a sufficiently large amount of data and applied to a single software project [7]. Indeed, Zimmermann et. al. conducted 622 combinations of cross-project prediction, but only 3.4% of them showed enough prediction performance [7]. They suggested that difference in distribution of metrics between fit dataset and test dataset influences the predictive accuracy.

To improve the prediction performance of cross-project prediction, we need to improve the generalization ability of a predictor model. One of the methods to solve this problem is normalization of dataset. Kuramoto showed that the prediction performance was improved using normalization of metrics in MLR models [8].

This paper proposes a prediction method called "an ensemble of simple regression models" to improve the generalization ability of MLR models. While "strong" multivariate models can cause overfitting problem, our technique uses an ensemble of "very weak" (1-variable) models to avoid overfitting. In our technique, we use weighted average of outputs of simple models based on the goodness of fit of the models. We also use normalized values of predictor variables.

This paper evaluates the proposed method using datasets of 12 projects from NASA IV&V Facility Metrics Data Program [10]. We compared predictive accuracy with normalized MLR models [8].

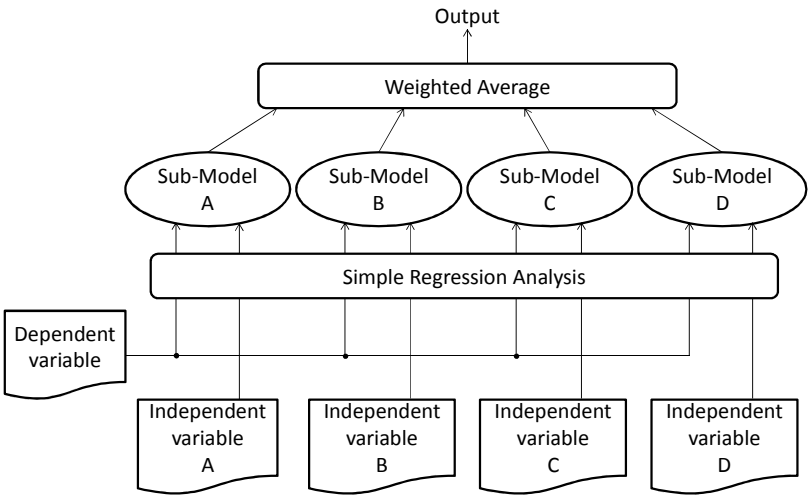

Figure 1. Overview of the proposed method

In what follows, Section 2 describes the proposed fault-prone module prediction method. Section 3 explains an experiment to evaluate the proposed method. Section 4 gives the discussion of the result. Finally Section 5 summarizes this paper.

II. Fault-Prone Module Prediction Model

A. An Ensemble of Simple Regression Models

Figure 1 shows the overview of the proposed method. The proposed method predicts a probability of having a fault in a module. Given a fit dataset for model construction, the proposed method constructs sub-models (simple logistic regression model) each using one of the software metrics. The weighted average of outputs of sub-models is used as a final output of the proposed method. The weights are computed based on the goodness of fit of each sub-model. While "strong" multivariate models can cause overfitting problem, our technique uses an ensemble of "very weak" sub-models to avoid overfitting in cross-project prediction.

The simple regression model is a technique to predict a dependent variable using single independent variable. While multivariate models like MLR model need to make sure that independent variables are certainly independent each other, our method can use a set of predictor variables that are dependent each other since each sub-model contain just one predictor variable.

The output y of our method is defined as the following formula (1).

$$y = \frac{\sum_{i=1}^{n} w_i f_i(x_i)}{\sum_{i=1}^{n} w_i} \qquad (1)$$

where y is a probability of having a fault in a module, n is the number of sub-models, $f_i(x_i)$ is i-th sub-model whose predictor variable is x_i, and w_i is a weight for i-th sub-model. In this paper we use as w the contribution ratio of the sub-model. The contribution ratio is one of the criteria of goodness of fit of a model. Its value range is [0,1].

In this paper, we use a simple logistic regression model as a sub-model $f_i(x_i)$, which outputs the probability of having a fault in a module. The model is defined by the following formula (2).

$$f_i(x_i) = P(y|x_i) = \frac{1}{1+e^{-(\alpha x_i+\beta)}} \qquad (2)$$

where, y is a dependent variable. x is an independent variable. α and β are regression coefficients. $P(y|x_i)$ is probability that y takes 1 with respect to the independent variable x_i.

B. Normalization of Metrics

Enough predictive accuracy is not achieved if the distribution of the metrics of test dataset is different from fit dataset's one. [5]. Kuramoto reported that by normalizing software metrics in fit dataset and test dataset can improve prediction accuracy of logistic regression models [8].

In this paper, we apply metrics normalization to our prediction method. The normalization is performed by two steps. The first step is logarithmic transformation and the second step is Z-score transformation. The purpose of these steps is to make the distribution of metric values closer to the standard normal distribution.

The logarithmic transformation changes a metric value into its logarithm by formula (3), where X is a metric value and Y is a transformed value. Note that 1 is added before log-transformation since some metrics contain zero (such as cyclomatic number.)

Table 1 Summary of NASA project data

project	language	# of metrics	Total SLOC	# of modules	# of bugs	% faulty
CM1	C	40	17K	505	48	9.50%
JM1	C	21	457K	10878	2107	19.40%
KC1	C++	21	43K	2107	325	15.40%
KC3	Java	40	8K	429	43	10.00%
MC1	C&C++	39	67K	4625	68	1.50%
MC2	C	40	6K	161	52	32.30%
MW1	C	40	8K	403	31	7.70%
PC1	C	40	26K	1059	76	7.20%
PC2	C	40	27K	4505	23	0.50%
PC3	C	40	36K	1511	160	10.60%
PC4	C	40	30K	1347	178	13.20%
PC5	C++	39	162K	15414	503	3.30%

$$Y = \log_{10}(X + 1) \qquad (3)$$

The Z-score transformation is performed by formula (4), where $\bar{Y}$ is the average of Y and ν is the standard deviation of Y. After this transformation, the average of Z becomes 0 and the standard deviation of Z becomes 1.

$$Z = \frac{Y - \bar{Y}}{v} \qquad (4)$$

III. Experimental Evaluation

A. Outline

The goal of the experiment is to evaluate the prediction performance of the proposed method . In this experiment, cross-project fault-prone module prediction was performed using three approaches; the proposed method, the proposed method with normalization, and the MLR (multivariate logistic regression) with normalization, which we refer to as "conventional method."

B. Datasets

The target dataset is the NASA IV&V Facility Metrics Data Program(MDP) datasets [10]. We used datasets of 12 projects. Details of each project are shown in Table 1. JM1 and KC1 consist of 21 software metrics while MC1 and PC5 consist of 39 software metrics. The remaining 8 projects had 40 software metrics. Therefore, we used 21 software metrics common to the 12 projects. In this paper, a "module" is a "source file."

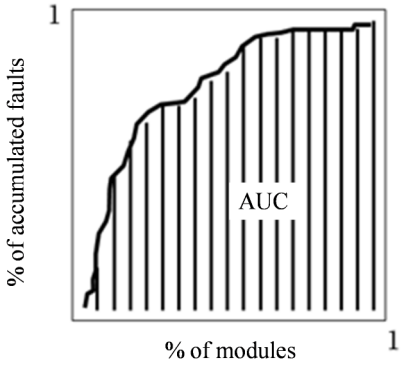

Figure 2. An example of the Alberg diagram

C. Evaluation Criteria

In this experiment, we used the area under the curve (AUC) of the Alberg diagram to evaluate the prediction performance. The Alberg diagram shows the percentage of accumulated number of faults when modules are ordered with respect to the probability of having a fault [9]. Figure 2 shows an example of AUC of the Alberg diagram. The larger AUC indicates better prediction performance. Also, AUC=0.5 means that the prediction performance is as worst as the random prediction.

D. Experiment Procedure

The fault-prone module prediction was performed to 132 combinations of project pairs. The procedure of the experiment is shown below.

Step1: Normalizing all software metrics of 12 project datasets.

Table 2 AUC of the conventional method (multivariate regression with metrics normalization)

		Fit Data											
		CM1	JM1	KC1	KC3	MC1	MC2	MW1	PC1	PC2	PC3	PC4	PC5
Test Data	CM1		0.749	0.726	0.695	0.717	0.726	0.769	0.716	0.725	0.691	0.53	0.749
	JM1	0.426		0.576	0.661	0.532	0.615	0.604	0.603	0.631	0.618	0.562	0.599
	KC1	0.619	0.733		0.768	0.63	0.743	0.733	0.653	0.647	0.708	0.417	0.741
	KC3	0.72	0.79	0.698		0.739	0.765	0.786	0.761	0.725	0.786	0.61	0.784
	MC1	0.669	0.812	0.717	0.844		0.707	0.802	0.813	0.774	0.833	0.78	0.792
	MC2	0.651	0.603	0.631	0.616	0.552		0.625	0.529	0.638	0.601	0.405	0.626
	MW1	0.728	0.745	0.703	0.735	0.635	0.703		0.723	0.738	0.752	0.388	0.742
	PC1	0.664	0.75	0.598	0.758	0.742	0.624	0.722		0.626	0.763	0.66	0.686
	PC2	0.451	0.803	0.86	0.86	0.735	0.831	0.849	0.612		0.86	0.659	0.791
	PC3	0.668	0.754	0.624	0.792	0.731	0.655	0.754	0.751	0.703		0.688	0.702
	PC4	0.474	0.714	0.577	0.777	0.706	0.597	0.665	0.638	0.638	0.731		0.65
	PC5	0.893	0.918	0.925	0.939	0.851	0.938	0.935	0.791	0.858	0.897	0.61	

Table 3 AUC of the proposed method

		Fit Data											
		CM1	JM1	KC1	KC3	MC1	MC2	MW1	PC1	PC2	PC3	PC4	PC5
Test Data	CM1		0.745	0.747	0.727	0.752	0.746	0.757	0.751	0.748	0.75	0.708	0.746
	JM1	0.655		0.645	0.661	0.658	0.654	0.665	0.658	0.648	0.649	0.664	0.643
	KC1	0.726	0.746		0.733	0.75	0.734	0.748	0.754	0.728	0.73	0.74	0.741
	KC3	0.77	0.778	0.775		0.78	0.773	0.776	0.781	0.77	0.782	0.781	0.768
	MC1	0.774	0.798	0.792	0.833		0.768	0.798	0.813	0.752	0.803	0.822	0.743
	MC2	0.634	0.631	0.632	0.616	0.628		0.631	0.626	0.638	0.634	0.609	0.648
	MW1	0.749	0.76	0.758	0.742	0.777	0.726		0.777	0.717	0.757	0.754	0.696
	PC1	0.658	0.679	0.675	0.684	0.703	0.658	0.684		0.657	0.715	0.696	0.659
	PC2	0.845	0.856	0.854	0.837	0.859	0.847	0.851	0.853		0.856	0.848	0.849
	PC3	0.706	0.733	0.725	0.736	0.752	0.706	0.743	0.75	0.686		0.745	0.678
	PC4	0.63	0.66	0.645	0.715	0.672	0.631	0.645	0.687	0.626	0.706		0.635
	PC5	0.853	0.921	0.925	0.898	0.921	0.913	0.922	0.936	0.871	0.898	0.916	

Step2: Constructing fault-prone detection models. In each model, one of 12 projects is used as a fit (training) dataset.

Step3: Prediction of the probability of having a fault. For each model, all the other projects are used as test datasets.

Step4: Evaluation of prediction performance using AUC of the Alberg Diagram.

E. Results and its Analysis

The prediction performance of each model is shown in Table 2, 3 and 4. The average of the AUC, the percentage of improvement of the average AUC (compared with the conventional method), and standard deviation of the AUC are shown in table 5. Below describes our findings in Table 2, 3, 4 and 5.

- The prediction performance of the proposed method was improved in 83 cases out of 132 compared with the conventional method.
- The prediction performance of the proposed method "with normalization" was improved in 98 cases out of 132 compared with conventional method.
- The improvements of the average AUCs were 0.037 (5.3%) in the proposed method and 0.043 (6.1%) in the proposed method with normalization, compared with the conventional method.
- The standard deviation of the AUC of the proposed method became smaller than the conventional method, which indicates that the proposed method did stable prediction.

Fig.3 shows the histograms of AUC of the conventional

Table 4 AUC of the proposed method with metrics normalization

		Fit Data											
		CM1	JM1	KC1	KC3	MC1	MC2	MW1	PC1	PC2	PC3	PC4	PC5
Test Data	CM1		0.734	0.741	0.739	0.744	0.74	0.747	0.748	0.738	0.744	0.724	0.74
	JM1	0.632		0.631	0.641	0.632	0.627	0.639	0.626	0.641	0.625	0.621	0.649
	KC1	0.752	0.749		0.752	0.75	0.748	0.753	0.749	0.745	0.748	0.736	0.753
	KC3	0.779	0.777	0.776		0.79	0.776	0.781	0.791	0.789	0.792	0.802	0.781
	MC1	0.793	0.78	0.788	0.794		0.777	0.798	0.814	0.795	0.817	0.833	0.8
	MC2	0.633	0.633	0.631	0.629	0.625		0.631	0.624	0.629	0.626	0.608	0.629
	MW1	0.759	0.742	0.752	0.752	0.764	0.743		0.77	0.744	0.77	0.733	0.754
	PC1	0.678	0.672	0.673	0.676	0.691	0.667	0.683		0.675	0.704	0.708	0.674
	PC2	0.861	0.841	0.858	0.857	0.86	0.858	0.86	0.856		0.861	0.858	0.853
	PC3	0.729	0.72	0.72	0.725	0.742	0.715	0.736	0.752	0.724		0.757	0.727
	PC4	0.652	0.653	0.649	0.658	0.676	0.641	0.652	0.681	0.667	0.683		0.658
	PC5	0.937	0.937	0.936	0.936	0.937	0.936	0.937	0.938	0.934	0.937	0.931	

Table 5 The average and the standard deviation of AUC

	Average	% of improvement	Standard Deviation
Conventional	0.703	—	0.109
Proposed	0.740	5.26%	0.082
Proposed with normalization	0.746	6.12%	0.089

Table 6 The number of cases for each AUC value range

		Conventional	Proposed	Proposed with normalization
AUC value	to 0.3	0	0	0
	0.3 to 0.4	1	0	0
	0.4 to 0.5	5	0	0
	0.5 to 0.6	10	0	0
	0.6 to 0.7	40	44	41
	0.7 to 0.8	56	63	62
	0.8 to 0.9	15	19	18
	0.9 to 1.0	5	6	11

method, the proposed method, and the proposed method with normalization. Table 6 shows the case distribution of AUC in each model. In the conventional method, there exist cases of AUC < 0.5, which means the prediction performance is worse than the random prediction. Since the prediction performance cannot be known beforehand, this

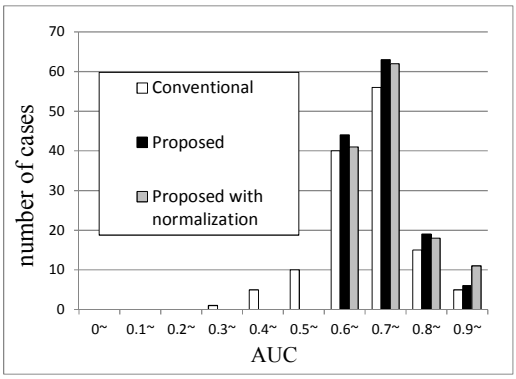

Figure 3. Histogram of AUC

indicates that the conventional method cannot be used for cross-project prediction since it often produces worse prediction than the random.

On the other hand, in the proposed method and the proposed method with normalization, all the cases are larger than 0.6, which means that the proposed method is much more practically useful than the conventional method.

IV. CONCLUSION

To improve the prediction accuracy of cross-project fault-prone module prediction, this paper proposed a modeling technique called "an ensemble of simple regression models." Our main idea is to use an ensemble of "very weak" (1-variable) models to avoid overfitting to the fit (training) dataset. In addition, we also employed metric normalization method to improve the prediction performance.

To evaluate the prediction performance of the proposed method, we conducted 132 combinations of cross-project

prediction using datasets of 12 projects from NASA IV&V Facility Metrics Data Program. As a result, the proposed method outperformed conventional multivariate logistic regression models in terms of AUC of the Alberg diagram. Moreover, while the prediction by the conventional method contained cases of AUC < 0.5, which means worse than the random prediction, the proposed method achieved AUC > 0.6 for all 132 predictions.

The major limitation of this paper is that we used only NASA MDP datasets. Our future work is to confirm our result using other datasets

REFERENCES

[1] Li, P.L., Herbsleb, J., Shaw, M. and Robinson, B., ”Experiences and Results from Initiating Field Defect Prediction and Product Test Prioritization Efforts at ABB Inc.” , Proc. 28th Int’l Conf. on Software Engineering (ICSE’06), pp.413–422 ,2006.

[2] Nagappan, N., Ball, T., and Zeller, A., ”Mining metrics to predict component failures”, Proc. Int’l Conf. on Software Engineering (ICSE’06), pp.452-461 ,2006.

[3] Kamei, Y., Sato, H., Monden, A., Kawaguchi, S., Uwano, H., Nagura, M., Matsumoto, K., Ubayashi, N., ”An Empirical Study of Fault Prediction with Code Clone Metrics”, In Proc. Joint Conference of International Workshop on Software Measurement and International Conference on Software Process and Product Measurement (IWSM/Mensura2011), pp.55-61, November 2011.

[4] John, C.Munson., and Taghi M.Khoshgoftaar., ”The detection of fault-prone programs. “, IEEE Trans. Softw. Eng., Vol. 18, No. 5, pp. 423–433, 1992.

[5] Fisher, R.A., “The Use of Multiple Measurements in Taxonomic Problems”, Annals Eugenics, Vol.7, Part II, pp.179-188, 1936.

[6] Rumelhart, D.E., Hinton, G.E. and Williams, R.J., “Learning Representations by Back-propagating Errors”, Nature, Vol.323, pp.533-536, 1986.

[7] Zimmermann, T., Nagappan, N., Gall, H., Giger, E. and Murphy, B., ”Cross-project Defect Prediction.”, The 7th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE’09), pp.91-100 ,2009.

[8] Kuramoto, T., Matsumoto, S., Kamei, Y., Monden, A., Matsumoto, K., ”Performance Improvement of Fault-prone Module Detection by Normalization of Software Metrics”, The Special Interest Group Technical Reports, IPSJ Special Interest Group on Software Engineering, Vol.2009-SE-166, No.11, 2009

[9] Ohlsson, N. and Alberg, H., ”Predicting Fault-Prone Software Modules in Telephone Switches.”, IEEE Trans. on Software Engineering, Vol.22, No.12, pp. 886–894 ,1996.

[10] NASA IV&V Facility Metrics Data Program, available from http://mdp.ivv.nasa.gov/