6B-1

最適な Error-prone モジュール予測器の識別手法の提案

松浦優† 小形真平‡ 海谷治彦‡ 海尻賢二‡
信州大学大学院工学系研究科情報工学専攻‡

1. はじめに

Error-prone モジュールの適切な予測は、ソフトウェアのテストや保守作業の効率向上において重要である。予測精度は訓練データや学習アルゴリズムの選択によって変わり、全てのプロジェクトに対して一律に最適なものは存在しないとされている。そのため、予測対象となるプロジェクト毎に考えなければならないが、その度に一から考えるのでは手間も時間もかかる。

本研究では、検査プロジェクトに対し適切な Error-prone モジュール予測器を識別するための識別器を作り、識別器によって選択された予測器を用いて予測を行う手法について実験を行った。本手法は Zhimin[1]が提案した手法をベースとしている。

2. データセット

本研究で対象とするデータは、SourceForge で公開されている 20 種類のプロジェクトである(ant, atn-ivy, ant-ivyde, DavMail, DeSmuME, DrJava, ffdshow, firebird, gnome-panel, gnome-session, inkscape, JCS, jEdit, Saros, Shareaza, SMPlayer, squirrel-sql, tomcat, VASSALEngine, WinMerge)。

また、メトリクスは Understand を使って収集した 24 種類を使用した(AvgCyclomatic, MaxCyclomaticStrict, CountLine, CountLineBlank, RatioCommentToCode, MaxCyclomaticModified, AvgCyclomaticModified, AvgEssential, CountDeclFunction, CountStmtExe, CountStmt, CountLineCodeDecl, CountSemicolon, CountLineCode, AvgCyclomaticStrict, CountLineCodeExe, MaxCyclomatic, CountLineComment , CountDeclClass, CountDeclClass, CountStmtDecl, SumCyclomaticStrict, SumCyclomatic, SumCyclomaticModified, SumEssential)。

Proposal of discrimination technique of optimal Error-prone module predictor.
† Yuu Matsuura, Shinpei Ogata, Hatuhiko Kaiya, Kenji Kaijiri
‡ Shinshu University, Graduate school of Science and Technology

3. 実験

3.1 手法

本研究で行う Error-prone モジュール予測手法について述べる。まず、様々な[訓練データ-アルゴリズム-検査データ]の組み合わせで予測を行い、予測結果を得る。その予測結果を任意の閾値によって True/False の二値に分ける。さらに、プロジェクトの各メトリクスを集約させることで新たな識別器訓練データセットを作る。この識別器訓練データセットをアルゴリズムによって学習させることで識別器を作る。ここまでが準備段階である。

次に、実際に予測を行いたいプロジェクトの検査データの各メトリクスを集約させる。識別器訓練データセットの中の[集約訓練データ-アルゴリズム]を抽出し、実際に予測を行いたい検査データと合わせて識別器検査データを作る。このデータを識別器にかけることで、各[集約訓練データ-アルゴリズム-集約検査データ]の組ごとに T/F の予測結果を得る。予測結果が T となった[訓練データ-アルゴリズム]の組み合わせで作った予測器を検査データに適用して実際の予測結果を得る、というのが流れである(図 1)。

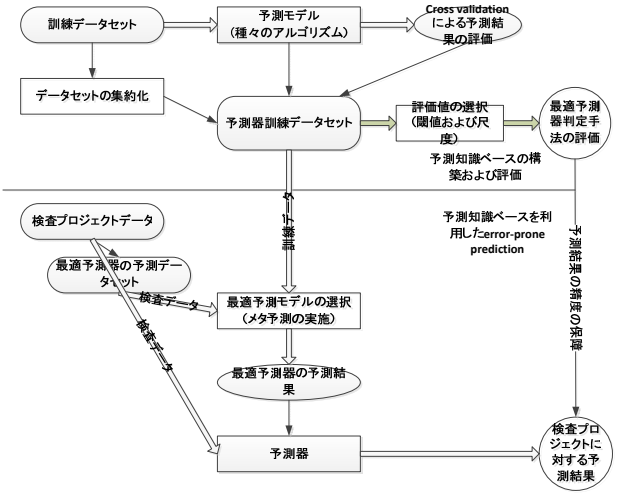

図 1 Error-prone モジュール予測手法の流れ

集約させる値は各メトリクスに対する分散、平均、中央値、第一四分位点、第三四分位点の 5 種類とした。さらにプロジェクトの特性として、

ファイル数、開発人数、そして三段階に分けたバグ含有率を用いている。アルゴリズムについては、BayesNet, IB1,MultiClassClassifier, MultilayerPerceptron, J48 の 5 種類を使用した。予測には WEKA を用いた。

3.2 実験内容と結果

まず、識別器が適切な予測器を正確に識別出来ているかの実験を行った。その実験結果の一部を表 1 に示す。訓練データの数値は recall の閾値である。

表 1 識別結果

訓練データ	アルゴリズム	precision
Recall_0.6_new.arff	MultiClass Classifier	0.838
Recall_0.7_new.arff	MultiClass Classifier	0.814
Recall_0.8_new.arff	MultiClass Classifier	0.808
Recall_0.7_new.arff	J48	0.796
Recall_0.9_new.arff	**J48**	**0.793**
Recall_0.8_new.arff	J48	0.783
Recall_0.6_new.arff	Multilayer Perceptron	0.761
Recall_0.6_new.arff	J48	0.754
Recall_0.9_new.arff	MultiClass Classifier	0.725

表 1 を見ると、recall の閾値を 0.9 という高い値にしても 0.8 に近い precision が得られる場合があった。これは、本手法によって選択された予測モデルの組み合わせを使えば、約 8 割の確率で recall が 0.9 以上の予測が出来ることを意味している。このように、高い precision を得られたケースが他にも見られ、今回対象としたプロジェクトの中では本手法が適切な予測器を識別することが出来ることを確かめられた。

次に、実際にどのような[訓練データ-アルゴリズム]の組み合わせが識別器によって選択されているかの一例を表 2 に示す。この例での検査対象プロジェクトは DrJava、閾値は recall が 0.7 である。表 2 は、「訓練データ」を「アルゴリズム」によって学習させた予測器に検査データ DrJava をかけた場合に、どのような予測結果(precision, recall)が得られたかということを示している。これを見ると、選択される訓練データには偏りがあることが分かる。選択された訓練データはバグ含有率が高いものが多く、recall の高い予測をする際には効果的に働いたものと考えられる。

表 2 選択された組み合わせ

訓練データ	アルゴリズム	precision	Recall
Gnome-panel	BayesNet	0.066	1
VASSAL Engine	Multilayer Perceptron	0.066	1
VASSAL Engine	MultiClass Classifier	0.066	0.976
VASSAL Engine	J48	0.069	0.964
VASSAL Engine	BayesNet	0.092	0.916
Gnome-session	Multilayer Perceptron	0.076	0.88

注意すべき点は、表 2 のように複数の組み合わせが選択された場合、どれを実際の予測に使うのかということである。表 1 のように識別を行って precision が 0.8 だったとしても、2 割は recall が閾値以下であるため、複数選択された場合には適切でより良いものを選択しなければならない。

4. おわりに

本研究では、識別器によって適切な予測器を識別することが出来る可能性を示した。しかし今回対象としたのは 20 種類のデータセットだけであり、一般性を考えれば他のプロジェクトデータでも実験を行ってみる必要がある。また、recall のみを閾値としているが、その条件を満たす予測の組み合わせの中には precision が著しく悪いものも多く、両方を考慮した手法への拡張を考える必要がある。予測に用いるアルゴリズムについても、今回は最適なものの選択は考えず種類を決めてしまったが、アルゴリズムによる精度の違いは存在するため、より良い予測のためには考える必要がある。

参考文献

[1] Zhimin He, et al: An investigation on the feasibility of cross-project defect prediction, ASE 19, 2 2012.