テキストを用いた不具合モジュール予測手法のチューニングについて

三 宅 陵 平[†1] 天 嵜 聡 介[†1]

ソフトウェア開発において不具合を効率よく摘出することが求められている．そのためには，不具合を含む可能性が高いモジュールを優先してレビューやテストを行うことが重要である．不具合モジュール予測手法の一つにテキストベースの手法が提案されている一方で，その予測精度に関しては改善の余地がある．本研究では，複数の側面からこのアプローチによる予測精度の向上に寄与する要因について調査した．ワークショップでは更なる精度向上の方策について議論したい．

The effects of tuning text-based fault-prone prediction models

Ryohei Miyake [†1] and Sousuke Amasaki[†1]

Efficient fault removal plays an important role in software development. The efficiency relies on how fast a review and testing detect software modules prone to be troubled. The good prioritization is thus essential. The text-based fault-prone method is one of such prioritization methods. While its effectiveness was appealed in the past-study, there is room for improving its performance. This study examined factors effective to performance improvement. We would like to discuss how to improve the performance.

1. はじめに

ソフトウェア開発では，不具合の混入を防ぐこと，効率的に不具合を除去することが求められる．出荷前のソフトウェアから不具合を除去するテスト工程の効率化は特に重要である．限られた時間で効率的にソフトウェアから不具合を除去するためには，不具合を含む可能性が高い箇所 (モジュール) を優先的にテストする必要がある．

あるモジュールが不具合を含む可能性を機械学習法などによって様々なメトリクスから予測する不具合モジュール予測手法の研究が盛んである[1)]．不具合モジュール予測手法の一つにテキストベースの手法[2)] が提案されている．テキストベースの手法は，ソースコードをテキストとみなして語の出現傾向から不具合モジュールを予測する手法である．同手法について良好な予測精度が示されている一方で，様々なチューニング要因があり，それら要因が予測精度に及ぼす影響については必ずしも明らかでない．

本研究では，テキストベースの手法に対してチューニングを行い，予測精度の向上に有効な要因を調査した．

†1 岡山県立大学
Okayama Prefectural University

2. チューニング要因

2.1 学習モデル

文献 2) では予測に Naive Bayes 法とロジスティック回帰モデルを利用していた．一方で，自然言語処理の分野ではテキストベースの予測問題では，SVM など強力な手法を用いることが一般的である．現状，テキストを用いた不具合モジュール予測問題では SVM を用いることが予測精度に及ぼす影響が明らかでない．精度に大きな差がない場合，モデルの学習がより高速な Naive Bayes 法などの手法が実用的な観点から適していると言える．

本研究では，SVM と Naive Bayes 法について予測精度の違いを評価した．SVM のカーネルとして，線形カーネル，多項式カーネル (2 次，3 次)，RBF カーネルの 4 種類を用いた．SVM の実装としては `libsvm` を用いた．また，前処理として変数選択を行っている．

2.2 語の扱い

自然言語処理において文書の類似度を扱う際，N-gram を用いることがある．N-gram とは，単語の n 個の並びを一つの単位として扱う方法である．例えば，「This is a pen」という文における 2-gram は,「This is」,「is a」,「a pen」の 3 つとなる．ソースコードにおける語の並びについて考慮することで，予測精度が向上する可能性が考えられる．本研究では，1-gram,

表 1 10-fold CV 予測における精度 (BIRT)

学習モデル	1-gram	2-gram	3-gram
SVM(linear)	0.57	0.63	0.67
SVM(poly-2)	0.63	0.68	0.69
SVM(poly-3)	0.43	0.54	0.60
SVM(RBF)	0.54	0.48	0.41
Naive Bayes	0.42	0.44	0.44

表 2 10-fold CV 予測における精度 (TPTP)

学習モデル	1-gram	2-gram	3-gram
SVM(linear)	0.64	0.71	0.70
SVM(poly-2)	0.62	0.67	0.67
SVM(poly-3)	0.40	0.51	0.52
SVM(RBF)	0.47	0.41	0.37
Naive Bayes	0.47	0.47	0.48

2-gram, 3-gram について調査した．

3. 実　　験

本研究では，10-fold cross-validation による評価実験を行った．データセットとしては，文献 2) で使用されたオープンソースソフトウェアのうち，Eclipse BIRT と Eclipse TPTP を対象とした．ストップワードの除去などデータセットの前処理は文献 2) に従った．

また，不具合を含むモジュールが少ないことを考慮し，不具合を含まないモジュールをアンダーサンプリングして学習を行った．アンダーサンプリングによって学習データにおける不具合モジュールの割合は 50%となっている．

4. 結　　果

10-fold CV による実験結果を表 1,2 に示す．数値はすべて F 値である．

表 1 について，Naive Bayes 法では n-gram の使用は精度に影響していないことが見て取れる．SVM では，線形カーネルおよび多項式カーネルを用いた場合には n-gram の使用が精度の向上に寄与していることが読み取れる．一方，RBF カーネルを用いた場合には，n-gram を使用することで逆に精度が低下した．Naive Bayes 法と SVM を比べると，SVM の方が精度が高かった．特に，比較的単純な線形カーネルを用いた場合で最も精度が良かった．

表 2 について，Naive Bayes 法では n-gram の使用は精度に影響していないことが見て取れる．SVM では，Eclipse BIRT の場合と同様に，線形カーネルおよび多項式カーネルを用いた場合に n-gram の使用が精度の向上に寄与していた．RBF カーネルを用いた場合も Eclipse BIRT の場合と同様に，n-gram を使用することで逆に精度が低下した．Naive Bayes 法と SVM を比べると，SVM の方が精度が高かった．特に，比較的単純な線形カーネルを用いた場合で最も精度が良かった．

今後さらに多くのデータセットによる検証が必要であるものの，2 つのデータセットのいずれにおいても同様の傾向が観察できたことから，この傾向には一定の一般性があると考えられる．

SVM による予測が良い結果を示した理由として考えられるのは，テキストベースの不具合モジュール予測が複雑な問題であること，そして，NaiveBayes 法より SVM の方が汎化能力が高いことである．

一方，N-gram の採用が予測精度の向上に寄与したのは，近い位置にある構文要素の共起関係が不具合の多寡と関連しているためだと考えられる．つまり，不具合を含みやすいプログラムの書き方をある程度捉えることに成功しているためであると考えられる．具体的にどのような N-gram が有効であったかの分析は今後の課題である．

5. おわりに

本研究では，テキストベースの不具合モジュール予測手法について，精度向上に寄与するチューニング要因について調査した．その結果，従来法で用いられていた Naive Bayes 法よりも SVM を用いた方が精度が向上することを明らかにした．また，SVM を用いた場合，n-gram を使用することでさらに精度向上が見込めることを明らかにした．

ワークショップでは，他に精度向上に影響するチューニング要因にどのようなものがあるのか，議論していきたい．

参　考　文　献

1) 畑　秀明，水野　修，菊野　亨：不具合予測に関するメトリクスについての研究論文の系統的レビュー，コンピュータソフトウェア，pp.106–117 (2012).
2) Hata, H., Mizuno, O. and Kikuno, T.: Fault-prone module detection using large-scale text features based on spam filtering, *Empirical Software Engineering*, Vol. 15, No. 2, pp. 147–165 (2010).