異なるコードクローンメトリクスを用いた欠陥モジュール予測の試み

A Challenge to Use Different Code Clone Metrics on Fault-prone Software Module Prediction

角田 雅照*　　梶村 和輝†　　亀井 靖高‡　　沢田 篤史§

あらまし　本研究では欠陥モジュール予測の精度を高めることをゴールとして，コードクローンメトリクスを予測モデルの説明変数として用いる．コードクローンが多い場合，モジュールの保守性が低下し，修正時に欠陥を混入する危険性が高まる．このため，コードクローンの多寡を考慮することにより，モデルの精度が向上することが期待される．いくつかの先行研究では，Kamiya らの検出方法に基づいてメトリクスを算出しているが，本研究では別の方法を用いて算出し，説明変数として用いることを試みる．

1　はじめに

大規模プロジェクトでは，欠陥を早期に発見できるようなテスト計画を立案するために，数学的モデルを用いて欠陥モジュールを予測することが行われる．ここで予測とは各モジュールが欠陥を含むか，含まないかを判別することをいう．モデルの予測精度が低い場合，テストが計画通りに進まずに進捗が遅れるなどの可能性があるため，精度を高めることは重要な課題のひとつである．

本研究では，モデルの予測精度を高めることを目指し，説明変数として用いるメトリクスについて着目する．説明変数として用いるメトリクスは，欠陥の有無と関連を持つ可能性が高いもの，例えば複雑度などが用いられるが，本研究ではコードクローンに着目し，説明変数として用いる．コードクローンとは，モジュール内，またはモジュール間における類似したコード片のことを指す．あるコードクローンを修正する必要が生じた場合，その他のクローンの箇所も同様に修正しなければならないことが多いが，それらの修正が漏れることがある．すなわち，コードクローンが多いとモジュールの保守性が低下し，修正時に欠陥を混入する危険性が高まることとなる．このため，コードクローンメトリクスを説明変数として用いることにより，モデルの精度が向上する可能性がある．いくつかの研究では[1][5]，コードクローンメトリクスを用いることにより予測精度を向上させることが試みられている．

コードクローンを検出，除去することは，ソフトウェアの保守性の向上に重要であるため，これまで様々な検出方法が提案されてきた[11]．コードクローンメトリクスはクローン検出方法が異なる場合，同一のモジュールを測定対象とした場合でも値が異なる．予測モデルの説明変数としてコードクローンメトリクスを用いた研究[1][5]では，Kamiya

* Masateru Tsunoda, 近畿大学
† Kazuki Kajimura, 近畿大学
‡ Yasutaka Kamei, 九州大学
§ Atsushi Sawada, 南山大学

らの検出方法[8]に基づいて算出している．ただし，Kamiya らの検出方法以外を用いて算出し，説明変数として用いた場合の効果はこれまで明らかではない．本研究では Kamiya らの検出方法以外を用いてコードクローンメトリクスを算出し，説明変数として用い，予測精度に違いが生じるかどうかを確かめる．

2 欠陥モジュール予測

欠陥モジュールを予測するための方法として，線形判別分析，ロジスティック回帰分析，分類木の 3 つの方法を用いる．これらの予測方法は，欠陥モジュールを予測するために広く用いられているものである．

線形判別分析: 線形判別分析は，一次式によりデータを 2 つのグループに分けることにより，欠陥モジュールなどの予測を行う方法である．線形判別分析で作成されるモデルを以下に示す．

$$y = a_1 x_1 + \cdots a_n x_n + b \qquad (1)$$

ここで y は目的変数であり，欠陥の有無を表す．モデル構築時には欠陥ありを 1，欠陥なしを-1 などで表す．x_i $(i = 1, ..., n)$は説明変数である．a_n は係数，b は定数項を表す．予測時には閾値を設定する．例えば閾値を 0 とした場合，y の値が 0.5 を上回る場合，該当モジュールは欠陥モジュールであると予測し，それ以外の場合は欠陥モジュールではないと予測する．

ロジスティック回帰分析: ロジスティック回帰分析は，ロジスティック式に基づいた予測モデルを構築する方法である．ロジスティック回帰分析により作成される予測モデルを以下に示す．

$$y = \frac{1}{1 + e^{-(a_1 x_1 + \cdots a_n x_n + b)}} \qquad (2)$$

ここで y は目的変数，x_i $(i = 1, ..., n)$は説明変数を表す．a_n は係数，b は定数項，e は自然対数の底を表す．予測の結果は，確率として得られる．例えば，予測結果が 0.7 の場合，予測対象のモジュールが fault-prone である確率は 70%であると判断する．

分類木：分類木は，説明変数であるモジュールの複雑度などの値によって分岐を繰り返すことにより，目的変数の値，すなわち fault-prone であるかどうかを予測するモデルである．作成されるモデルは木構造となる．例えば，モジュールのコード行数が 1000 行を超えない場合は欠陥なしとし，超える場合はさらにサイクロマチック数（モジュールの複雑度）が 20 を超えない場合は欠陥なし，超える場合は欠陥ありなどと予測する．

分類木の作成方法には，CHAID，C4.5 など，様々なものが提案されており，それぞれ分岐の基準などが異なる[9]. 本研究ではパラメータの設定が容易である CART を用いる．

3 コードクローンメトリクスを用いた予測

コードクローンの検出方法は単一ではなく，複数の方法が提案されている．検出方法は主に以下の 4 種類に分類される[6].

- 文字/行の並び: ソースコードをそのまま文字列として扱い，文字，または行の並びが一致しているかどうか，コードがクローンかどうかを判定する．
- トークンの並び:ソースコードをトークンに置き換え，トークンが一定長以上連続して一致している場合，コードクローンとみなして検出する．
- AST: ソースコードを AST（Abstract Syntax Tree）に変換し，構文木を比較するこ

とによりクローンかどうかを判定する．

- 依存関係: ソースコードから PDG（Program Dependence Graph）を作成し，グラフを比較することによりクローンかどうかを判定する．

コードクローンメトリクスとは，各モジュールおけるコードクローン箇所の行数や，コードクローンの含有率（コードクローン箇所の行数÷総行数）などを指す．コードクローンメトリクスは保守性と関連があるため，欠陥モジュール予測モデルの説明変数として用いることにより精度が高まる可能性がある．コードクローンメトリクスを説明変数として用いた研究が少数ではあるが存在する．

馬場ら[1]は，説明変数としてコードクローン含有率とコードクローンとなっているコード片の個数を用い，その他の説明変数としてサイクロマチック数，コード行数などを用いて欠陥モジュール予測モデルを構築している．モデル構築に用いたデータは，日本のソフトウェア開発において得られた 40 個のモジュールであり，約 8 割のモジュールが欠陥を含んでいる．モデルの構築にはロジスティック回帰分析を用いている．さらに，RNR（Ratio of Non-Repeated code）という，コードクローンの非繰り返し度を表すメトリクスを用い，この値が 50 未満の場合はコードクローンとみなさない処理を行っている．実験の結果，予測精度が改善したことを示している．

亀井ら[5]は，予測モデルの説明変数として，馬場らのメトリクスに加え，クローンに含まれる条件分岐と繰り返し構文の数など，合計 5 つのコードクローンメトリクスを用いている．コードクローンメトリクス以外に，合計 15 個の複雑度メトリクスを説明変数とし，ロジスティック回帰分析により予測モデルを構築している．馬場らの研究で用いられている RNR は，亀井らの研究では用いられていない．実験では，Eclipse プロジェクトから得られた 8000 個以上のモジュールを用いてモデルを構築し，予測精度を評価している．なお，欠陥を含んでいたモジュールの割合は約 18%である．亀井らの研究では，コードクローンメトリクスが欠陥モジュール予測の精度改善に寄与しないことが示されている．ただし，規模の大きなモジュールの場合，精度が改善することを示唆している．

これらの先行研究では，コードクローンメトリクスを算出するために，トークンの並びに基づく検出方法である Kamiya らの検出方法[8]に基づくツールを用いている．ただし，検出方法によりコードクローンの定義が異なるため，別の検出方法を用いた場合，コードクローン含有率などのメトリクスに差異が生じる．このため，別の検出方法を用いてコードクローンメトリクスを算出し，説明変数として用いた場合，予測精度が高まる可能性がある．そこで本研究では，Kamiya らの検出方法以外によるコードクローンメトリクスを用いて予測することを試みる．

4 実験

4.1 概要

実験では，コードクローン検出方法の違いから生じる，コードクローンメトリクスの差異に着目し，欠陥モジュール予測モデルの精度に対し，この差異が影響するのかどうかを確かめた．実験では以下の 3 種類の欠陥モジュール予測モデルを構築し，それぞれの精度を比較した．

- モデル 1:単一のコードクローン検方法に基づくメトリクスを用いる
- モデル 2: 2 つのコードクローン検出方法に基づくメトリクスを組み合わせて用いる
- モデル 3:コードクローンメトリクスを用いない

モデル 2 は 2 つのコードクローン検出方法を適用し，得られたメトリクスを同時に用いた場合の精度を確かめるためのモデルである．モデル 3 は，コードクローンメトリクスが精度向上に寄与するかどうかを追実験するためのモデルである．

4.2 データセット

D'Ambros ら[2]は，欠陥モジュール予測モデルの精度評価に用いるためのデータセットを公開している[3]．このデータセットは，オープンソースソフトウェアのプロジェクトから収集した，リリース後の欠陥数や CK メトリクスなどを含んでいる．本研究では，比較的規模の小さな Lucene プロジェクトのデータを実験に用いた．実験対象のモジュール数は 311 個であり，うち 52 個（16.7%）が欠陥モジュールであった．

データセットには，CK メトリクスのようなソースコードメトリクスや，モジュールの変更履歴などを含んでいる．先行研究[1][5]では，クローンメトリクスの他にソースコードメトリクスを説明変数として用いている．そこで本研究も同様に，クローンメトリクス以外に，オブジェクト指向プログラムの複雑度を表すメトリクス（CK メトリクスなど）とソースコード行数の計 17 個を説明変数として用いた．

クローンメトリクスについては，ソースコードをダウンロードし，クローン検出ツールを適用することにより算出し，その後，クラス名をキーとしてマージをした．クローンメトリクス計測ツールとして，CCFinderX[7]と CloneDR[12]を用いた．CCFinderX はトークンの並び，Clone DR は AST に基づいてクローンを検出し，メトリクスを計算する．トークンの並びに基づく検出方法は先行研究で用いられており，また，クローンとして検出されるコードが比較的多い．逆に，AST に基づく検出方法はクローンとして検出されるコードが比較的少ないため，比較対象として採用した．これらのツールにより容易に計測可能なクローンメトリクスである，クローン含有率を予測モデルの説明変数として用いた．クローン含有率はコードクローン箇所の行数÷各モジュールの総行数と定義される．なお，各ツールで検出されるコードクローンは，必ずしもコピーアンドペーストで作成されたものとは限らず，従ってコードクローンとして検出された箇所全てが，保守性が低いとは限らない．

4.3 手順

実験では，データをランダムに 2 等分し，一方をラーニングテータ，他方をテストデータとする fold-out 法を 10 回適用してモデルを評価した．fold-out 法は予測モデルを評価する（交差妥当性を確認する）際に用いられる一般的な方法であり，例えば文献[10]などで用いられている．このとき，ラーニングデータとテストデータにおける欠陥モジュールの割合が偏らないように分割した．モデル 1，2，3 の構築には，線形判別分析，ロジスティック回帰分析，分類木の 3 つの方法を用いた．ロジスティック回帰分析適用時には AIC（Akaike's Information Criterion）に基づく変数選択を行ったが，クローンメトリクスが必ず説明変数に含まれるようにした．

予測精度の評価基準として，AUC (Area Under the Curve)を用いた．AUC とは ROC (Receiver Operating Characteristic)曲線の下の面積であり，値域は[0, 1]である．AUC が 1 に近いほど予測精度が高いことを示す．ROC 曲線とは，閾値を変化させて陽性率と偽陽性率を計算し，線によりつなげたものである．

4.4 結果

結果を表 1 に示す．表の最下行太字の平均は，3 つの予測モデル構築方法の AUC の平均値を示す．Clone DR を用いたモデル 1 と CC Finder X を用いたモデル 1 の AUC の平均値を比較すると，差は大きくはないが，Clone DR のほうが高い予測精度となっていた．また，線形判別分析，ロジスティック回帰分析，分類木それぞれの AUC を比較した場合も，Clone DR の予測精度が CC Finder X を下回ることがなかった．

表 1 各モデルの AUC

モデル	モデル 1 CC Finder X 使用	モデル 1 Clone DR 使用	モデル 2 CC Finder X, Clone DR 使用	モデル 3 クローンメトリクス不使用
分類木	0.586	0.587	0.601	0.588
線形判別	0.651	0.688	0.672	0.693
ロジスティック回帰	0.662	0.685	0.654	0.693
平均	**0.633**	**0.653**	**0.643**	**0.658**

Clone DR を用いたモデル 1 と，両方のツールのメトリクスを用いたモデル 2 の AUC の平均値を比較すると，前者の方が高い値となっていた．実験に用いたデータセットにおいては，複数のコードクローン検出ツールから得られたクローンメトリクスを同時に用いても，予測精度の向上は見られなかった．CC Finder X を用いたモデル 1 の予測精度が低かったことから，CC Finder X に基づくクローンメトリクスが悪影響を与えた可能性がある．

モデル 3 とその他のモデルの AUC の平均値を比較すると，クローンメトリクスを用いないモデル 3 の精度のほうが高かった．実験で用いたデータセットにおいては，クローンメトリクスは欠陥モジュールの予測精度を改善する効果は見られなかった．これは，先行研究[5]と類似の結果である．

ロジスティック回帰分析時の変数選択の結果は，fold-out 法を 10 回適用したそれぞれの場合においてかなり異なっており，半数以下の変数しか説明変数として採用されなかった場合もあれば，ほとんどの変数が採用された場合もあった．説明変数としては public メソッドの数などが選ばれる場合が多く，プログラム行数は選ばれない場合が多かった．CC Finder X と Clone DR を用いたモデル同士を比較すると，採用された説明変数は異なっていたことから，それぞれのメトリクスのモデルに対する影響も異なると推測される．

4.5 考察

CC Finder X と Clone DR の精度の相対的な差は最大で 6%（(0.688-0.651) / 0.651）であり，差が大きいとはいえないが無視できるほど小さいともいえない．よって，CC Finder X 以外のツール，例えば Clone DR などを用いてクローンメトリクスを算出してモデルを構築すべきかどうかの結論を下すためには，さらなる分析が必要となる．まずは，文字/行の並びに基づくクローン検出方法や依存関係に基づく検出方法など，さらに多くの検出方法を適用するとともに，複数のデータセットを用いて分析する必要がある．

どのデータセットにおいても，特定の検出手法によるクローンメトリクスを用いた場合の予測精度が高い場合，その検出手法を用いることが最も適していることとなる．データセットにより精度の高い検出手法が異なる場合，精度が平均的に高く，かつ精度の分散が小さい手法を用いることが適切といえる．あまり精度に差がない場合は，適用時のスケーラビリティ（規模の大きなソフトウェアに適用できるか）や適用時のコストを考慮して手法を選ぶとよいといえる．

さらに結論の信頼性を高めるためには，各クローン検出方法により検出されるクローンにどのような違いがあるのかなど，定性的な分析も行う必要がある．例えば文字/行の並びに基づく検出方法は，単純なコピーアンドペースト部分しか見つけられないが，トークンの並びに基づく検出手法では変数名の違いは無視し，構造が似ている部分を検出する．類似のコードを共通ルーチンに集約するという面では後者が適しているが，保守性低下の原因となるコピーアンドペースト部分を発見するためには前者が適している可

能性がある．

コードクローンには無計画なコピーアンドペーストなどにより作られたものと，何らかの理由で計画的に作成されたクローンが存在する可能性がある．その場合，前者が多いデータセットではコードクローンメトリクスを用いることにより予測精度が高まり，後者が多いデータセットでは予測精度が変化しないと考えられる．それらを考慮した分析は今後の課題のひとつである．

5 おわりに

本研究では，欠陥モジュール予測モデルにおいて，コードクローンメトリクスを説明変数として用いる場合に，コードクローン検出方法の差異が精度に影響を与えるかどうかを確かめた．実験に用いたデータセットにおいては，クローンメトリクスを算出したクローン検出ツールが異なる場合，若干ではあるが予測精度に違いが生じた．ただし，結果に一般性があり，かつ予測精度に明確な差があるかどうかまではいえなかった．また，実験では CC Finder X と Clone DR しか適用しておらず，結果の信頼性を高めるためには，多数存在しているその他のコードクローン検出ツールを適用する必要がある．今後の課題は，適用するデータセットとコードクローン検出ツールを増やし，どのコードクローン検出ツールを用いれば最も予測精度が高まるのかを明らかにすることである．これにより，予測モデル構築時に，最適なコードクローン検出ツールを調査，検討する労力を省くことができると期待される．

謝辞 本研究の一部は，文部科学省科学研究補助費（基盤 C：課題番号 25330090），2014 年度南山大学パッヘ奨励金 I-A-2 による助成を受けた．

参考文献

[1] 馬場慎太郎，吉田則裕，楠本真二，井上克郎：Fault-Prone モジュール予測へのコードクローン情報の適用，電子情報通信学会論文誌 D，Vol. J91-D，No. 10，pp. 2559-2561, 2008.
[2] D'Ambros, M., Lanza, M. and Robbes, R.: An extensive comparison of bug prediction approaches, *Proc. international working conference on mining software repositories (MSR)*, pp. 31-41, 2010.
[3] D'Ambros, M., Lanza, M. and Robbes, R.: http://bug.inf.usi.ch/
[4] Göde, N. and Koschke, R.: Incremental Clone Detection, *Proc. European Conference on Software Maintenance and Reengineering (CSMR)*, pp. 219-228, 2009.
[5] 亀井靖高，左藤裕紀，門田暁人，川口真司，上野秀剛，名倉正剛，松本健一：クローンメトリクスを用いた Fault-Prone モジュール判別の追実験，電子情報通信学会論文誌，Vol. J93-D，No. 4，pp. 544 -547, 2010.
[6] 神谷年洋，肥後芳樹，吉田則裕：コードクローン検出技術の展開，コンピュータソフトウェア，Vol. 28，No. 3，pp. 29-42, 2011.
[7] Kamiya, T.: http://www.ccfinder.net/
[8] Kamiya, T., Kusumoto, S. and Inoue, K.: CCFinder: a multilinguistic token-based code clone detection system for large scale source code, *IEEE Transactions on Software Engineering*, Vol. 28, No. 7, pp. 654-670, 2002.
[9] 金明哲：R と樹木モデル（1），ESTRELA，5 月号，pp70-76，統計情報研究開発センター，2005.
[10] T. Khoshgoftaar, and P. Rebours: Improving Software Quality Prediction by Noise Filtering Techniques, *Journal of Computer Science and Technology*, Vol. 22, Issue 3, pp 387-396, 2007.
[11] D. Rattana, R. Bhatiab, and M. Singhc: Software clone detection: A systematic review, *Information and Software Technology*, Vol. 55, Issue 7, pp. 1165-1199, 2013.
[12] Semantic Designs: nchttp://www.semdesigns.com/Products/Clone/