6ZK-01

開発者支援ツールのデータを利用したコードレビュー時間の推定

大山義人† 大場みち子‡

公立はこだて未来大学大学院 システム情報科研究科† 公立はこだて未来大学‡

1. 研究背景

システム開発プロセスにおいて，コードレビューが多くの開発現場で取り入れられている．コードレビューとは，IPA によると「開発者担当者が書いたソースコードを閲読してセキュリティ脆弱性あるいはそのきざしを読み取る作業」[1]とされている．その中でも GitHub などのバージョン管理サービスを使用したコードレビューが一般的である．GitHub はコードの変更内容を本番環境に反映する前にレビューできる PullRequest 機能を持っており，コードレビューに適したサービスである．コードレビューは『レビュワーのスキル』やレビュー対象の『ソースコードの属性』など多くの要因が関係しているため，レビューにかかる時間などの『レビュー結果』を推定しにくい．『レビュワーのスキル』や『ソースコードの属性』，『レビュー結果』の関係を分析することで，コードレビューの質や所要時間の見積もりができる可能性が高い．コードレビューに関する指標化と，それを用いた推定は多数研究でされている．そこで，GitHub のソースコードを分析することで，どのような情報がコードレビュー時間に影響があるのかを明らかにする．

2. 研究目的

本研究の目的は，ソースコードのレビュー時間とソースコードの属性を分析することである．

3. 関連研究

コードレビュー時間に関係する研究を述べる．

ソースコードの属性として，循環的複雑度[2]と呼ばれるソースコードの複雑さを測定する手法がある．1＋分岐（if，for，while など）の数で表すことができ，その数字の大小によってコードの構造がどれだけ複雑か推定できる．循環的複雑度はソースコードの可読性を数量的に表すことを表している．

GitHub と StackOverflow というエンジニアコミュニティサイトの活動実績から開発者のスキルをスコア化した研究[3]がある．開発者の GitHub での活動や StackOverflow での回答から，開発者がどの言語や技術ジャンルに詳しいか分析をしている．これにより，特定の専門分野に詳しい開発者を簡単に発見することができる．

コードレビュー分析のためにレビュー内容のカテゴリ分けした研究[4]がある．コードレビューのコメントをどのような種類のコメントか手動でタグ付けし，カテゴリ分けをしている．具体的な分類の種類には，バグや欠陥の発見，コードの改善，代わりのアプローチの提案などが考えられる．この手法を利用することにより，コードレビュー内容の分析をすることができる．

先行研究[5]では，コードレビュー時間とソースコード属性の関係性を重回帰分析を用いて分析した．先行研究[6]では，先行研究[5]では分析できなかった属性を含めた重回帰分析を行った．それらの結果，レビュー時間は属人的で，特にレビュイーに依存していた．具体的には，コードレビュー時間にはコードレビューを受ける人の過去の平均レビュー時間が最も関係が深いことがわかった．しかし，コードレビュー時間を十分な精度で推定できなかった．

4. 研究課題

先行研究[6]には，大きく分けて 2 つの問題点がある．1 つ目は，推定モデルの精度が低いため，推定の信頼性が低い点である．2 つ目はレビュイーとレビュー時間以外の関係が抽出出来ていない点である．

5. 解決アプローチ

課題を 2 つの手法によって改善する．モデルの精度が低く，推定の信頼性が低いという問題点は，機械学習を採用することで解決する．重回帰分析などの統計解析はデータの可視化に適しているが，機械学習は推定精度を高めるのに適しているからである．レビュイーとレビュー時間以外の関係が抽出できていないという問題点は，特徴量エンジニアリングを用いる事によって解決する．これは，適切に特徴量エンジニアリングすることで，説明変数の選択を効率的に行うことができるからである．

6. 機械学習を用いた実験

6.1 実験目的

Estimating Code Review Time Using Data from Developer Support Tools

†Yoshito Oyama ‡Michiko Oba

†School of System Information Science, Future University Hakodate

‡Future University Hakodate

本実験の目的はソースコードのどのような属性がコードレビュー時間と関係しているのかを明らかにすることである．

6.2 実験対象

GitHub での対象とするリポジトリの選定基準はスター数が多いこと，コードレビューが行われている OSS リポジトリであることである．スター数とは，GitHub 上で個人がリポジトリを覚えるため，コントリビューターに感謝を示すために使用されており，リポジトリの信頼度の目安として使用することができる．以上の条件を満たすリポジトリを 6 つ選択する．また，分析に利用する PullRequest の年代が大幅に異なると，使用されてるツールやオンライン上でのコードレビューの普及度合いが異なるため，2015～2020 年の 6 年間のデータを使用する．データ取得方法として，GitHub REST API v3 を使用する．API クライアント PyGitHub を使用して，GitHub REST API v3 を利用しデータを対象リポジトリから取得する．

6.3 実験方法

実験は GitHub からデータ取得，データの前処理，特徴量エンジニアリング，機械学習を使用してコードレビュー時間との関係性分析の順番で行う．

取得したデータは特徴量エンジニアリングや機械学習で分析しやすいよう，GitHubAPI で直接取得できないデータ・変数の取得や，ノイズデータを取り除くなどのデータの前処理をする．データの前処理で取り除くデータは主に，Bot によるプルリクエスト・コミットなどだが，人間によるコードレビューがされていないプルリクエストや，コードレビューとは関係のない議論によって merge まで大幅な時間がかかっているものをとり除く．

6.4 実験結果

実験の結果（表 1），決定係数が 0.353 となり，先行研究[6]に比べて，僅かに向上した．

表 1 リポジトリごとの推定精度

組織	リポジトリ	決定係数（学習）	決定係数（検証）
Facebook	React	0.271	0.315
	React Native	0.269	0.219
Microsoft	Vscode	0.615	0.41
	terminal	0.287	0.275
laravel	Laravel	0.511	0.252
vuejs	Vue	0.441	0.332
	合計	0.585	0.353

6.5 実験からの考察

実験がこのような結果になった理由は大きく分けて 3 つ考えられる．

1 つ目はデータ選択の問題である．一部の特徴量は，API から直接取ることができないので手動で抽出している．そのため，特徴量として抽出できていない変数が存在している可能性がある．また，指標の正しさに問題がある可能性がある．例えばコードの量を特徴量として手動で抽出する際，LOC として抽出する方法とサイズとして抽出する方法があり，正しい指標を選択できていない可能性がある．

2 つ目は特徴量エンジニアリングに問題がある可能性である．xfeat の利用方法に習熟していなかったため，利用方法が適切でない可能性である．また，特徴量エンジニアリングは xfeat 以外にも存在するため，今回のデータに適したライブラリが存在する可能性がある．

3 つ目は機械学習手法に問題がある可能性である．今回は機械学習の回帰モデルとして Elastic Net を使用したが，Elastic Net 以外の手法の検討も必要だと考える．

7. まとめ

本研究では GitHub の情報を利用して，コードレビュー時間の関係性を機械学習と特徴量エンジニアリングを用いて分析した．その結果，先行研究に比べて精度は僅かに向上した．今後の展望として，高い精度で推定できるように推定モデルの精度を向上させていきたいと考えている．

参考文献

[1] IPA:IPA ISEC セキュア・プログラミング講座：C/C++言語編 第 2 章 脆弱性回避策とソフトウェア開発工程：ソースコードレビュー，IPA(オンライン),入手先 <https://www.ipa.go.jp/security/awareness/vendor/programmingv2/contents/c103.html>(参照 2021-12)

[2] Yang, Cheng, Xun-hui Zhang, Ling-bin Zeng, et al.: RevRec: A Two-Layer Reviewer Recommendation Algorithm in Pull-Based Development Mode, Journal of Central South University, Vol.25, No.5, pp.1129-43 , (2018).

[3]Bacchelli, Alberto, and Christian Bird.: Expectations, Outcomes, and Challenges of Modern Code Review, Proc. Proceedings of the 2013 International Conference on Software Engineering(ICSE '13), USA: IEEE Press, pp.712-721, (2013).

[4] McCabe, T. J.: A Complexity Measure, IEEE Transactions on Software Engineering, Vol.2, No.4, pp.308-20, (1976).

[5] 大山義人，大場みち子：GitHub のデータを利用したコードレビュー時間の推定，第 82 回全国大会講演論文集，Vol.2020，No.1，pp.199-200（2020）．

[6] 大山義人，大場みち子：GitHub のデータを利用したコードレビュー時間に関連した属性の分析，第 83 回全国大会講演論文集，Vol.2021，AK-03（2021）．