G-13

社会的相互作用に着目した GitHub リポジトリへのスター付与数の見積もり手法

橋本 大輝†　伊原 彰紀†　小口 知希†
Daiki Hashimoto　Akinori Ihara　Tomoki Koguchi

1. はじめに

多くのソフトウェア開発プロジェクトは，ソースコードを含む，開発中に作成されるドキュメントのバージョン管理，および，開発者間の共有のために，Web サービス GitHub[1] を利用している．GitHub が持つ SNS 機能を使うことで，プロジェクトは，バージョン管理しているデータサーバ（リポジトリ）を容易に公開，共有することが可能である．さらに GitHub の普及に伴って，ソフトウェアの一部に GitHub で公開されたソースコードを再利用することも容易になり，効率的なソフトウェア開発を実現している．

GitHub で公開されているリポジトリは現在 4,200 万件[2]を超え，現在も増え続けている．膨大に登録されているリポジトリの中から，開発者にとって再利用できるソフトウェアの発見が困難であるため，Ouni らは，再利用可能と考えられるリポジトリを推薦する手法を提案している [6]．GitHub では，開発者が興味を持つリポジトリを記憶しておくために，リポジトリにスター (Star) を付与することが可能である．開発者が Star を付与したリポジトリは，GitHub における開発者のアカウントページにおいて，ブックマークのように管理することができる．このようにリポジトリに付与された Star は，人気の指標はもちろん，ソフトウェアの信頼性を示す指標としても用いられている [7]．リポジトリに付与される Star 数の増減は，ソフトウェアの継続的な保守を判断するための指標にもなり得る．

Sahin らは，リポジトリに蓄積された活動履歴に基づき，1 週間後にリポジトリに付与される Star 数を予測するための手法を提案している [3]．提案手法で使用する活動履歴には，プログラム変更提案に使用される Pull request の投稿数，不具合報告に使用される Issue の投稿数などが使用されている．これらの活動履歴は，プロジェクトの継続的な活動を捉え，Star 数を予測するための有効性を確認している．開発者が Star を付与する動機は，ソフトウェアの将来の安定性に限らず，開発者自身の興味や専門性にも依存すると考える．同じ専門性を有する開発者は，同じリポジトリに Star を付与する可能性が高く，開発者間の社会的相互作用が，リポジトリへの Star の付与数の予測に有用であると示唆する．

本論文では，GitHub の機能である開発者間の Follow 関係を社会的相互作用と捉え，多数の Follower を有する開発者の Star 付与が，リポジトリの Star 数の予測に有用であるか否かを明らかにする．GitHub の Follow 関係は， GitHub 利用者のアカウントを，別の利用者が Follow することで，利用者の GitHub 上での活動を追跡する機能である．従って，多数の開発者に Follow されている開発者は，特定のリポジトリへの Star 付与の行動が，他の開発者に通知されるため，当該リポジトリへの将来の Star 付与に影響すると考える．

続く 2 章では関連研究を述べ，本研究の立ち位置を説明する．3 章では，本研究で用いる分析手法について説明し，4 章で実際のデータを用いたケーススタディを行う．そして，5 章ではケーススタディで得た結果を分析し，6 章で考察を行う．最後に 7 章で，まとめと今後の課題を述べる．

2. 関連研究

2.1 リポジトリへの Star 付与

多くの研究では，多数の Star が付与されているリポジトリの特徴を調査している．Aggarwal らは，多数の Star を常に有するリポジトリは，当該リポジトリで管理されるソフトウェアのドキュメント整備に協力する利用者が多いことを明らかにしている[1] ．Borges らは，リポジトリに蓄積された活動履歴と Star 数との相関関係を分析している．分析の結果，特にコミットとプロジェクト年数に弱い相関を確認し，フォークとは強い相関関係を明らかにしている [2]．Borges らは，ソフトウェアのアップデートが遅いほど，リポジトリへの Star 付与が少ないことを明らかにしている [8]．

昨今では，リポジトリに蓄積された活動履歴に基づく Star 付与数の予測モデル構築手法が提案されている [3] [8] [9]．

2.2 社会的相互作用

GitHub のような透明性の高いコミュニティでは，多数の Follower 数を有する開発者はロックスター[10]と呼ばれ，GitHub コミュニティにおける人気や地位を表す指標となっている．Blincoe らは，800 人の GitHub 利用者にインタビューを行った結果，ロックスターと呼ばれる人気のある開発者のプロジェクトへの参加は，当該開発者に引き寄せられてプロジェクトに参加することを示している．本論文では，開発者間の Follower 関係を社会的相互作用と捉え，多数の follower を有する開発者のリポジトリへの Star 付与が，当該リポジトリにおける Star 数の予測に寄与するか否か評価実験を行う．

3. 分析方法

本章では，提案する予測モデルの構築に向けて，使用するメトリクスの概要，メトリクスの計測方法，予測モデルに用いるアルゴリズムの説明を行う．

3.1 概要

本論文では，リポジトリに付与されたスター数を予測するためのモデル構築に向けて，従来研究で使用されたリポジトリに記録された開発者の活動履歴に加えて，多数の follower を有する開発者によるスター付与を新たなメトリクスとして提案する．具体的には，過去にスターを付与し

[1] GitHub: https://github.com/
[2] https://github.com/search?q=is:public

†和歌山大学システム工学部，Faculty of Systems Engineering

表 1　説明変数リスト

従来研究メトリクス	Number of star	プロジェクトが期間内に取得したスター数
	Number of fork	プロジェクトが期間内に取得したフォーク数
	Number of commit	プロジェクトが期間内に取得したコミット数
	Number of release	プロジェクトが期間内に取得したリリース数
	Number of open issue	期間内にプロジェクトに投稿されたイシュー数
	Number of close issue	期間内にプロジェクトで閉じたイシュー数
提案メトリクス	Number of follower_25	期間内にスターをつけた開発者の中で，フォロワー数が全体の 25%以下の人数
	Number of follower_50	期間内にスターをつけた開発者の中で，フォロワー数が全体の 25%以上 50%以下の人数
	Number of follower_75	期間内にスターをつけた開発者の中で，フォロワー数が全体の 50%以上 75%以下の人数
	Number of follower_100	期間内にスターをつけた開発者の中で，フォロワー数が全体の 75%以下の人数

た開発者数を個々の開発者の follower 数に基づき分類することで，follower 数が多い開発者によるスター付与は，リポジトリへのスター数の増加に影響を与えるか否かを評価する．

予測モデル構築において，説明変数の計測期間を 4 パターン（予測日から過去 1 日以内，7 日以内，14 日以内，30 日以内），および，目的変数の計測期間を 4 パターン（予測日から 1 日以内，7 日以内，14 日以内，30 日以内）の組み合わせることで，説明変数がスター数の予測結果に与える効果を調査する．

3.2　目的変数，説明変数の計測

表 1 は，予測モデルに使用する説明変数のリストを示す．従来研究[3]で提案されたメトリクスのうち，fork，commit，issue は，それぞれ GHTorrent[1]から取得し，Release，Star は著者らが GitHub API[2] を用いて取得した．GHTorrent は，Gousios らが GitHub から収集したデータセットであり，GitHub に登録された 1 億件以上のリポジトリの活動履歴を収集しているデータセット[4][5] である．本論文では，GHTorrent（収集期間：2015 年 11 月 18 日から 2018 年 1 月 14 日まで）のデータを使用する．従来研究で提案されたメトリクスは，GitHub を使用したソフトウェア開発の主な活動である．具体的には，頻繁にコミットが行われているリポジトリは，機能追加，不具合修正が頻繁に行われているリポジトリとして，ソフトウェアの継続的な拡張が期待される．また，issue は，不具合報告要求，機能追加要求の数を示し，ソフトウェアの利用者の多さを表す．Fork は，Star 数と高い相関を持つことを従来研究で明らかにされている[1]．

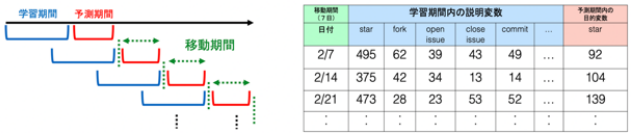

図 1　データセット作成の概略図

本論文では，従来研究のメトリクスに加えて，Star を付与した開発者が有する Follower 数によって 4 段階に分類し，それぞれの段階に該当する開発者数を新しいメトリクスとして提案する．Follower 数の区分は，学習期間内に Star を付与した開発者のうち，フォロワー数の四分位数を用いる．学習期間内にフォロワー数が全体の 25%以下の開発者の人数を Number of follower_25，学習期間内にフォロワー数が全体の 25%以下の開発者の人数を Number of follower_25，学習期間内にフォロワー数が全体の 25%以上 50%以下の開発者の人数を Number of follower_50，学習期間内にフォロワー数が全体の 50%以上 75%以下の開発者の人数を Number of follower_75，学習期間内にフォロワー数が全体の 75%以上の開発者の人数を Number of follower_100 とし，期間内のメトリクスとして用いる．Follower 数の分類基準については，本論文の分析結果に基づき，さらなる検討が必要と考える．

目的変数は，実験対象のリポジトリにおいて，予測時点から 1 日以内，7 日以内，14 日以内，30 日以内に付与された Star 数を予測する．ただし，GitHub API では，予測対象としている Star 数は，リポジトリに付与された始めの 40,000 件しか取得できないため，本論文でリポジトリに付与されたはじめの 40000 件の Star 数を対象に予測モデルを構築する．

説明変数，および，目的変数は，学習期間内に取得したメトリクスの偏りを抑えるために，予測時点の移動期間を 1 日，7 日とするデータセットを作成する．図 1 は，計測方法の概略図を示す．図 1 は移動期間 7 日（予測時点 2 月 7 日，2 月 14 日，2 月 21 日）を示す．移動期間 7 日では，学習期間 14 日，予測期間 7 日としているため，2 月 7 日を予測時点とした場合，学習期間は 1 月 24 日から 2 月 6 日まで表 1 に示す説明変数を計測し，予測期間は 2 月 7 日から 2 月 13 日までにリポジトリに付与される Star 数を予測する．

3.3　予測モデルの構築

予測モデルの構築において，説明変数間の相関が高くなると多重共線性と呼ばれる問題が生じる．多重共線性はモデルの精度に悪影響を及ぼすため，以下の式に定義される VIF（分散拡大係数）が 5 以上の説明変数は学習データから削除する．相関係数 r はピアソンの積率相関係数を用いる．

$$VIF = \frac{1}{1 - r^2}$$

本論文では Python 言語のパッケージ sklearn. ensemble. RandomForestClassifer [3]に含まれるランダムフォレストで予測モデルを構築する．

[1] GHTorrent: https://ghtorrent.org/
[2] GitHub API: https://developer.github.com/v3/
[3] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

表 2 －（a）モデルの評価

		予測期間			
		1 日	7 日	14 日	30 日
学習期間	1 日	0. 06	−0. 02	−0. 14	−0. 13
	7 日	0. 28	0. 72	0. 74	0. 74
	14 日	0. 26	0. 77	0. 84	0. 90
	30 日	0. 07	0. 86	0. 92	0. 97

表 2 －（b）モデルの評価

		予測期間			
		1 日	7 日	14 日	30 日
学習期間	1 日	−0. 17	−0. 07	−0. 19	−0. 17
	7 日	0. 15	0. 07	−0. 04	0. 01
	14 日	0. 04	0. 01	−0. 04	0. 24
	30 日	−0. 23	0. 30	0. 51	0. 74

表 2 －（c）モデルの評価

		予測期間			
		1 日	7 日	14 日	30 日
学習期間	1 日	0. 06	−0. 01	−0. 14	−0. 15
	7 日	0. 27	0. 76	0. 83	0. 87
	14 日	0. 29	0. 82	0. 94	0. 96
	30 日	0. 07	0. 86	0. 92	0. 97

表 2 －（d）モデルの評価

		予測期間			
		1 日	7 日	14 日	30 日
学習期間	1 日	−0. 19	−0. 05	−0. 19	−0. 80
	7 日	0. 14	0. 03	−0. 18	−0. 01
	14 日	−0. 02	0. 13	0. 29	0. 34
	30 日	−0. 28	0. 15	0. 32	0. 72

3.4 評価方法

本論文では，十分割交差検証を用いて予測モデルの構築，評価を行なう．学習データを 10 個に分割し，その中で 1 つをテストデータ，他の 9 つを学習データとして，予測モデルを構築する．学習データを分割した 10 個それぞれをテストデータとすることでモデルの評価を 10 回行う．さらに，十分割交差検証を 100 回行うことで，合計 1000 回モデルの評価を行なう．最終的に，1000 回分の評価結果の中央値をモデルの評価結果とする．

モデルの評価指標には，決定係数 （R2）を用いる．決定係数は，1 以下の値域を取り，値が 1 に近い値であるほど予測結果が実際の Star 数との誤差が小さく，モデルの評価として高いこと示す． y_iは実測値，$\bar{y_i}$は実際の値の平均値，Yは予測モデルが出力した値，Nはデータ数を示す．

$$R^2 = 1 - \frac{\sum_{i=0}^{n-1}(yi - Y)^2}{\sum_{i=0}^{n-1}(yi - \bar{y_i})^2}$$

表 3　メトリクスの特徴量

	提案モデル	従来モデル
fork	0.13	0.18
commit	0.15	0.21
openIssue	0.15	0.20
closeIssue	0.15	0.20
star	0.15	0.21
release	0	0
follower_25	0.14	
follower_50		
follower_75	0.13	
follower_100		

4. ケーススタディ

4.1 データセット

本論文では，Microsoft が開発するテキストエディタ Visual Studio Code[1] のリポジトリ[2] を分析対象とする．Visual Studio Code は，テキストエディタとしての機能はもちろん，拡張機能をインストールすることで，プログラムデバッグ，GitHub との連携，プログラム補完などをサポートする機能を有する．Visual Studio Code はオープンソースソフトウェアとしてプログラムを GitHub で 2015 年に公開し，2020 年 7 月 22 日現在では 10,070 個の Star が付与されている．本論文では，GHTorrent に記録されている 2015 年 11 月から 2018 年 1 月までの開発履歴を収集した．GitHub API で取得可能な Star 数は 40,000 件であり，分析対象とする Visual Studio Code のリポジトリでは， 2018 年 1 月に Star 数が 40,000 に達している．

4.2 分析結果

表 2 は，従来手法で使用されたメトリクスのみで構築した予測モデルの評価結果（表 2−(a)，表 2−(b)），および，本論文が提案する Follower 数のメトリクスを追加して構築した予測モデルの評価結果（表 2−(c)，表 2−(d)）を示す．表 2−(a)，表 2−(c)は，予測時点の移動期間が 1 日のデータセット，表 2−(b)，表 2−(d)は，予測時点の移動期間が 7 日のデータセットを用いている．それぞれの表には，学習期間 4 パターン（1 日，7 日，14 日，30 日）と予測期間 4 パターン（1 日，7 日，14 日，30 日）の組み合わせた 16 個の予測モデルの決定係数を示す．本評価結果から，予測結果に影響する 3 つの事象を確認した．

- 予測時点の移動期間が 7 日のデータセットに比べて，移動期間が 1 日のデータセットの方が，予測精度が高い
- 目的変数の Star 数を計測する期間が長いほど予測精度が高い
- 説明変数の計測期間が長いほど予測精度が高い

これら事象について次章の考察で検討する．

[1] https://azure.microsoft.com/ja-jp/products/visual-studio-code/

[2] https://github.com/microsoft/vscode

5. 考察

5.1 予測時点の移動期間

予測時点の移動期間が 1 日のデータセットを用いて構築した予測モデルは，予測時点の移動期間が 7 日の予測モデルに比べて，決定係数が極めて高い．機械学習技術において，学習データの特徴とテストデータの特徴が類似しているほど予測精度は向上する．本ケーススタディでは，予測時点の移動期間が 1 日のデータセットを 10 分割交差検定により評価したことにより，テストデータと類似する特徴を持つ学習データで予測モデルを構築し，高い予測精度の構築を達成することができたと考える．

本手法は，10 分割交差検証を行なっているため，予測時点よりも未来のデータが学習データに含まれていることが本論文の制約ではある．この制約があるとはいえ，予測時点の移動期間が 7 日のデータセットでは決定係数が低く，実用的な予測モデルを構築することはできないことを確認し，今後は予測日から 7 日よりも短いデータセットも含めた学習データを準備することが必要であると示唆する．

5.2 Star 数を計測する期間

目的変数の Star 数を計測する期間が長いほど予測精度が高い．特に，予測時点の移動期間が 1 日の結果を示す表 2-(a)， 表 2-(c)から，Star 数の計測期間が 1 日の予測モデルに比べて，7 日以上の予測モデルの決定係数が高い．その理由は，予測期間が長いほど Star 数の分散が大きい一方で，予測した Star 数の誤差は予測期間が 1 日と 7 日で大きな差がないからである．

5.3 学習データのメトリクス

表 3 は，データセット移動期間 1 日，学習期間 14 日，予測期間 7 日の予測モデルに貢献した説明変数の重要度を示す．重要度とは，予測結果を算出するために各説明変数が予測モデルに寄与する度合いを示す指標である．ランダムフォレストで構築される多くの決定木が，取得したデータから分類を行う際に，到達したノードの説明変数が重み付けされる．最も重要度が高い説明変数は，予測結果を出すまでに何度も効果的に働いたことを示す．

表 2-(c)における学習期間が 7 日，14 日においては，本論文が提案する follower_25 と follower_75 のメトリクスを追加した場合に予測精度が高くなることが多かった．follower_50 と follower_100 の説明変数は多重共線性を考慮した際に除かれた．follower_75 は学習期間内に follower が多い開発者が Star をつけた場合に，予測する Star 数が増える要因として，follower_25 は予測する Star 数が減少する要因として重要であったことが示唆される．

6. 制約

本論文では，多数の Star を付与された OSS プロジェクトである Visual Studio Code のみを対象に分析を実施した．Visual Studio Code は 2015 年に GitHub にリポジトリが作成され，GHTorrent に記録された約 2 年間の多数の開発履歴を使用したため，分析結果から得た知見は信頼できると考える．しかし，異なるプロジェクトを対象とした場合，分析結果が異なる可能性があるため，今後は他のプロジェクトも対象に同じ実験を実施する．

本論文で予測モデル構築において，予測対象としている Star 数は，リポジトリに付与された始めの 40,000 件しか取得できないため，40,000 件以降のスター履歴は分析の対象とすることができない．

本論文で提案する Follower メトリクスの閾値は，リポジトリに Star を付与した開発者の Follower 数の四分位数としているため，プロジェクトによって異なる値になる．複数プロジェクトを対象に分析を行う際は，共通の閾値を用いることを検討する必要がある．

本論文では，社会的相互作用を表す Follower メトリクスがスター数予測に有意であるか検証することが目的であるため，データの時系列を考慮していない．しかし，スター数予測を実際の開発現場で利用することを想定する場合は，時系列を考慮する必要がある．

7. おわりに

本論文では，プロジェクトの人気予測モデルの向上を目的とし，Star を付与した開発者の Follower 数をメトリクスに追加してモデルの構築，評価を行った．学習期間と予測期間を変化させて精度を比較することで，次の知見が得られた．

- 目的変数の Star 数を計測する期間が長いほど予測精度が高い．
- 学習データの計測期間が 7 日，14 日においては，follower_25 と follower_75 のメトリクスを追加した場合に予測精度が高い．

今後は，さらなる予測精度向上に向けて，今回考慮していない時系列を含めた分析の比較や，fork などの開発形態に即した説明変数の追加を行う．

謝辞

本研究は，JSPS 科研費 18KT0013 の助成を受けたものです．

参考文献

[1] Kran, A. , Abram, H. and Eleni, S. : Co-evolution of project documentation and popularity within github, Proceedings of the 11th Working Conference on Mining Software Repositories (MSR’14), pp. 360-363 (2014).

[2] Hudson, B. , Andre, H. , and Marco, T. V. : Understanding the Factors That Impact the Popularity of GitHub Repositories, Proceedings of the 32nd International Conference on Software Maintenance and Evolution (ICSME’16), pp. 334-344 (2016).

[3] Sefa, E. S. , Kubilay, K. and Ayse, T. : Predicting Popularity of Open Source Projects Using Recurrent Neural Networks, Proceedings of the 15th International Conference on Open Source Systems (OSS’19), pp. 80-90 (2019).

[4] Gousios, G. and Spinellis, D. : GHTorrent: GitHub’s data from a firehose, 9th Working Conference on Mining Software Repositories (MSR’12), pp. 12-21 (2012).

[5] Gousios, G. , Vasilescu, B. , Serebrenik, A. and Zaidman, A. : Lean GHTorrent: GitHub data on demand, Proceedings of the 11th Working Conference on Mining Software Repositories (MSR’14), pp. 384-387 (2014).

[6] Ali, O. , Raula, G. K. , Marouane, K. , Takashi, I. , Daniel, M. G. and Katsuro, I. : Search-based

software library recommendation using multi-objective optimization, Information and Software Technology, Vol. 83, pp. 55-75 (2017).
[7] Hudson, B. and Marco, T. V.: What's in a GitHub Star? Understanding Repository Starring Practices in a Social Coding Platform, Journal of Systems and Software, 146, pp. 112-129 (2018)
[8] Hudson, B., Andre, H. and Marco T. V.: Predicting the Popularity of GitHub Repositories, Proceedings of the 12th International Conference on Predictive Models and Data Analytics in Software Engineering, pp. 1-10 (2016).
[9] Neda, H. B., Gita, S., Heather, K. and Ivan, G.: A Cross-Repository Model for Predicting Popularity in GitHub, Proceedings of the 5th Intarnational Conference on Computational Science and Computational Intelligence (CSCI'18).
[10] Mihael, J. L., Bruce, F., Junghong, C., Jungpil, H., Jae, Y. M. and Jinwoo, K.: GitHub developers use rockstars to overcome overflow of news, Proceedings of the Extended Abstracts on Human Factors in Computing Systems (CHI'13), pp. 133-138 (2013).