

Predicting Merge Conflicts in Collaborative Software Development

Moein Owhadi-Kareshk

Sarah Nadi

Julia Rubin

背景

- 複数人の開発者が同じコードを変更するとマージでの競合が起こる
- 開発者はマージを行う前にコンフリクトを手動で解決する必要がある
 - エラーが発生しやすく時間のかかるタスク

先行研究

- 様々な特徴量と競合との相関関係を調査[13]
 - どの特徴量も強い相関がないことを発見
 - マージでの競合を予測できない可能性がある結論付ける
 - **非線形データにステップワイズ回帰を用いているため、問題がある**
- 各関数の編集回数などをカウントし予測[14]
 - Precision : 57.9% Recall : 82.67%
 - カウントに基づくため予測モデルは構築しない
 - **1つのデータセットを使用. 潜在的な特徴量を考慮していない**

[13] O. Leßenich, J. Siegmund, S. Apel, C. Kästner, and C. Hunsen, "Indicators for merge conflicts in the wild: survey and empirical study," Automated Software Engineering, vol. 25, no. 2, pp. 279–313, 2018.

[14] P. Accioly, P. Borba, L. Silva, and G. Cavalcanti, "Analyzing conflict predictors in open-source java projects," in Proceedings of the 15th International Conference on Mining Software Repositories. ACM, 2018, pp. 576–586.

研究の貢献とリサーチクエスチョン

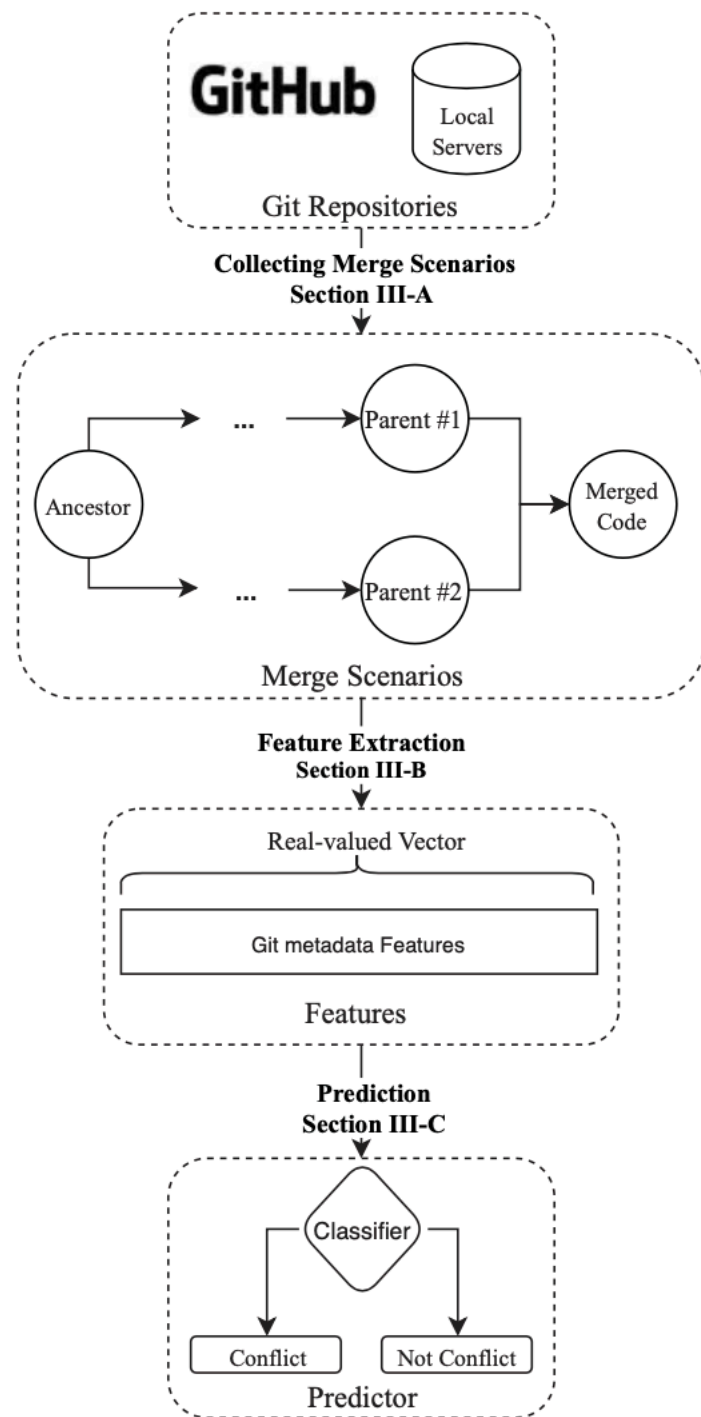
- 貢献

- Gitから抽出可能な特徴量から予測モデルが作成可能か調査
 - ▶ 744プロジェクトを対象とした，これまでで最大のコンフリクトに関する研究
 - ▶ 過去の文献（コードレビュー，コンフリクト）から潜在的な特徴量を作成

- リサーチクエスチョン

1. どのマージに関する特徴量がコンフリクトに影響を与えるか？
2. マージに関する特徴量を使用してコンフリクトを予測できるか？

研究の流れ

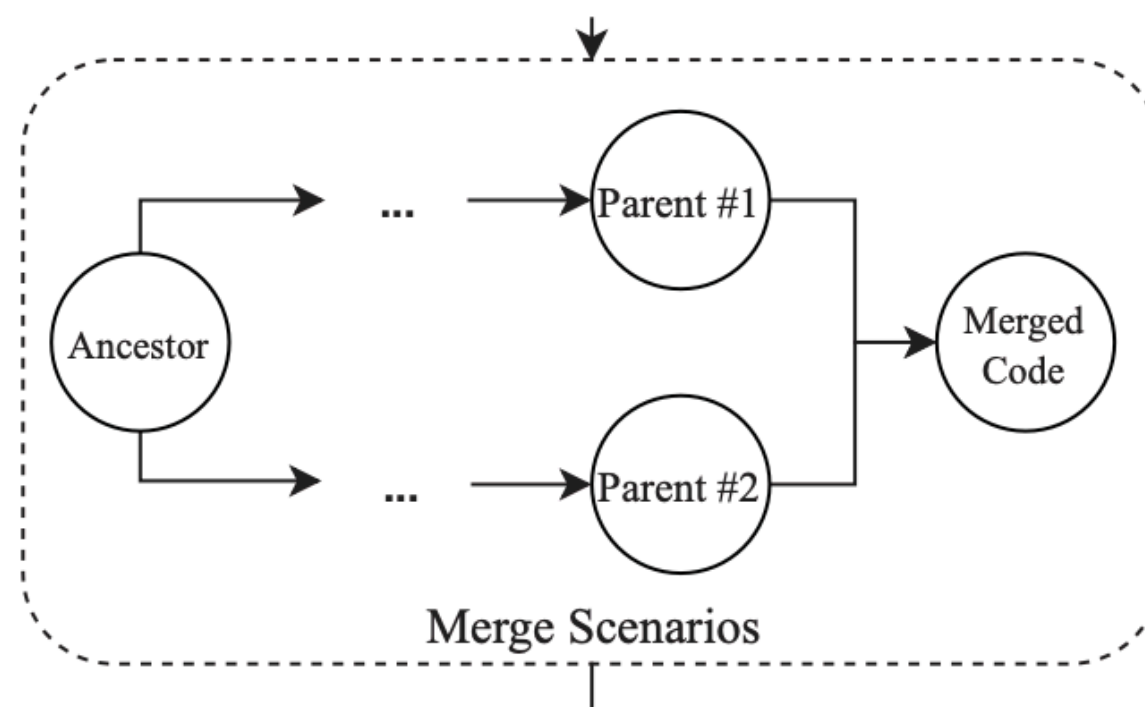


- Collecting Merge Scenarios
 - Gitリポジトリからマージに関するデータを取得する
- Feature Extraction
 - Gitの履歴から、マージに関する特徴量を抽出する
- Prediction
 - 機械学習を用いて予測モデルを構築
 - コンフリクトが起きるか、安全にマージできるかの2値分類

Fig. 1. Methodology for Creating the Proposed Conflict Predictor

Collecting Merge Scenarios

- Git履歴からMERGANSER[36]を用いて, マージが起こったポイントを抽出
 - 3way-mergeまで抽出可能
- コンフリクトが起きたマージをGitの定型フレーズより抽出
 - [Automatic merge failed; fix conflicts and then commit the result]



Feature Extraction

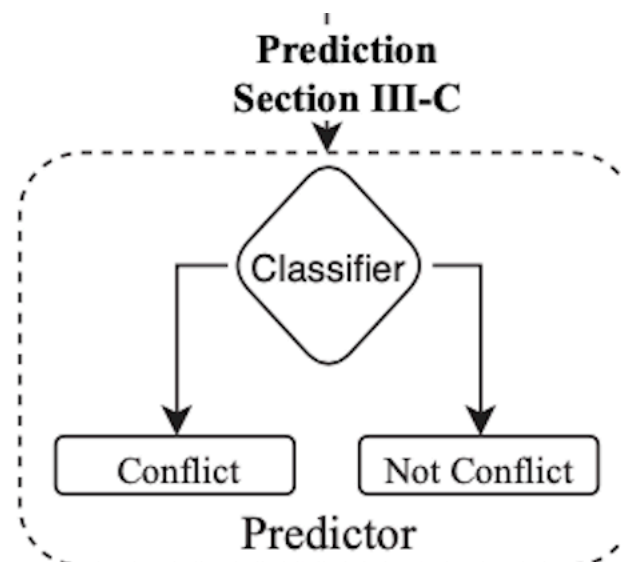
- 既存研究を参考に特徴量を作成

TABLE I
FEATURE SETS USED FOR TRAINING THE MERGE CONFLICT PREDICTOR

No.	Feature Set	References	Intuition for Including this Feature Set	Dimension
1	No. of simultaneously changed files in two branches	[5], [13]	The increase in simultaneously changed files increases the chance of conflicts. If the value of this feature is zero, no conflict can occur.	1
2	No. of commits between the ancestor and the last commit in a branch	[13], [34], [35]	Having more commits means more changes in a branch, which may increase conflicts	1
3	Commit density: No. of commits in the last week of development of a branch	[13]	Lots of recent activity may increase the chance of conflicting changes	1
4	No. added, deleted, renamed, modified, and copied files in a branch	[13], [34], [35]	More code changes may increase the chance of conflicts	5
5	No. added and deleted lines in a branch	[13], [35]	More code changes may increase the chance of conflicts	2
6	No. of active developers in a branch	[9], [34], [35]	Having more developers increases the chances of getting inconsistent changes	1
7	The frequency of predefined keywords in the commit messages in a branch. We use 12 key-words: fix, bug, feature, improve, document, refactor, update, add, remove, use, delete, and change.	[35]	These keywords can provide a high-level overview of the types of code changes and their purpose. Certain types of changes may be more prone to conflicts.	12
8	The minimum, maximum, average, and median length of commit messages in the branch	[35]	The length of a commit message can be an indicator of its quality	4
9	Duration of the development of the branch in hours	[9]	The longer a branch exists for, the more likely it is for inconsistent changes to happen in one of the other branches	1
Total number of features:				28

Prediction

- 分類器の選択
 - アンサンブル学習手法（ランダムフォレスト）
- 分類器の選択利用
 - マージのコンフリクト予測の対象データは不均衡データであるため
 - SMOTEは計算コストがかかる



データセット

- reaper dataset[15]より作成
 - reaper dataset : <https://reporeapers.github.io/>

TABLE II
DESCRIPTIVE STATISTICS OF OUR DATASET

Programming Languages	# Repositories	# Merges	# Conflicting Merges	Conflict Rate (%)
<i>C</i>	80	18,824	1,308	6.95
<i>C++</i>	109	42,420	3,621	8.54
<i>C#</i>	110	38,945	3,153	8.1
<i>Java</i>	120	36,853	2,190	5.94
<i>PHP</i>	112	50,342	4,737	9.41
<i>Python</i>	106	49,583	5,533	11.16
<i>Ruby</i>	106	40,690	3,192	7.84
<i>Sum</i>	744	267,657	21,734	-
<i>Weighted Average</i>	-	-	-	8.12

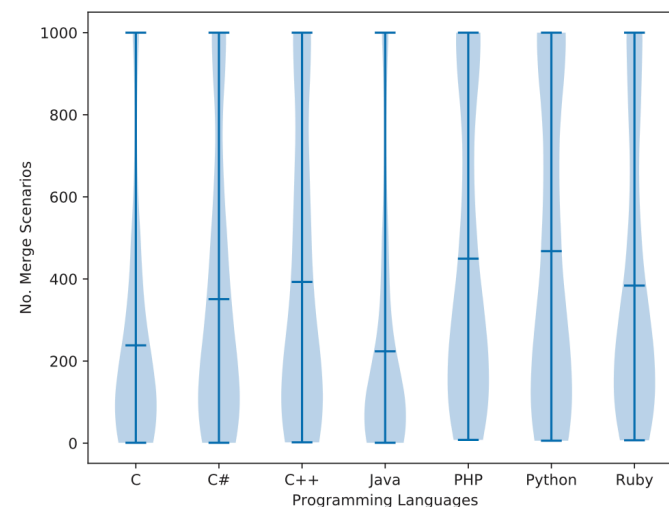


Fig. 2. The Distribution of Merge Scenarios

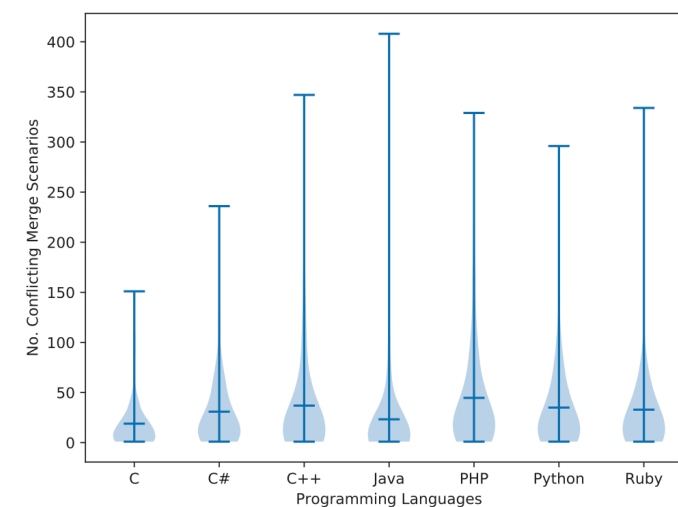


Fig. 3. The Distribution of Conflicting Merge Scenarios

RQ1の分析方法

- RQ1：どのマージに関する特徴量がコンフリクトに影響を与えるか？
- 方法
 1. スピアマンの順位相関により，各特徴量とコンフリクトの関係を分析
 2. Decision Treeを用いてモデル化し，各特徴量の重要度を分析

RQ1 : 1. スピアマンの順位相関の結果

TABLE III

SPEARMAN'S RANK-ORDER CORRELATION COEFFICIENTS (CC) AND THE CORRESPONDING P -VALUES (p) BETWEEN EACH FEATURE SET AND THE EXISTENCE OF CONFLICTS, SEPARATED BY LANGUAGE. THE CORRELATIONS THAT ARE EQUAL OR GREATER THAN 0.6 WITH P -VALUES (p) LESS THAN 0.05 ARE HIGHLIGHTED.

Feature Set	C		C++		C#		Java		PHP		Python		Ruby	
	CC	p	CC	p	CC	p	CC	p	CC	p	CC	p	CC	p
1	0.53	0.0	0.58	0.0	0.52	0.0	0.60	0.0	0.60	0.0	0.53	0.0	0.58	0.0
2	0.30	0.0	0.33	0.0	0.28	0.0	0.33	0.0	0.35	0.0	0.29	0.0	0.32	0.0
3	-0.17	0.0	-0.17	0.0	-0.07	0.0	-0.14	0.0	-0.19	0.0	-0.08	0.0	-0.01	0.0
4	-0.15	0.0	-0.14	0.0	-0.07	0.0	-0.13	0.0	-0.18	0.0	-0.08	0.0	-0.01	0.0
5	-0.14	0.0	-0.13	0.0	-0.07	0.0	-0.14	0.0	-0.19	0.0	-0.08	0.0	-0.10	0.0
6	-0.02	0.01	0.02	0.0	0.03	0.0	0.03	0.0	0.05	0.0	0.0	0.89	0.01	0.21
7	0.01	0.13	0.0	0.95	0.01	0.25	0.0	0.57	0.01	0.22	0.0	0.58	0.01	0.22
8	0.0	0.69	0.0	0.98	0.01	0.32	0.01	0.25	0.0	0.75	0.01	0.24	0.0	0.75
9	0.0	0.98	0.0	0.99	0.0	0.55	0.0	0.76	0.0	0.67	0.01	0.11	0.0	0.65

- Feature1（変更ファイル数）は強い相関
- Feature2（コミット数）は弱い相関

RQ1 : Decision Treeの重要度の結果

TABLE IV
FEATURE IMPORTANCE BASED ON A DECISION TREE CLASSIFIER

Feature Set	C	C++	C#	Java	PHP	Python	Ruby
1	0.93	0.98	0.95	0.93	0.95	0.94	0.94
2	0.01	0.01	0.0	0.02	0.01	0.01	0.01
3	0.01	0.0	0.01	0.01	0.0	0.01	0.0
4	0.01	0.0	0.01	0.02	0.01	0.01	0.01
5	0.01	0.0	0.01	0.01	0.01	0.01	0.0
6	0.01	0.01	0.01	0.01	0.01	0.02	0.02
7	0.01	0.0	0.01	0.0	0.01	0.01	0.01
8	0.01	0.0	0.01	0.01	0.01	0.0	0.01
9	0.01	0.0	0.0	0.01	0.0	0.0	0.01

- Feature1（変更ファイル数）は重要度が高い
- 一方で，Feature2（コミット数）は重要度が低い

RQ1の結論：

同時に変更されるファイル数と各ブランチのコミット数に相関関係あり。
しかし，重要度は同時に変更されるファイル数のみが高い

RQ2：予測方法

- RQ2：マージに関する特徴量を使用してコンフリクトを予測できるか？
- 作成する分類器
 - Decision Tree
 - 特徴量を全て使用
 - Random Forest
 - 特徴量を全て使用
 - Baseline1（パラメータ設定のための最低精度として使用）
 - ランダムに予測
 - Baseline2
 - Feature1（同時に変更されるファイル数）のみを用いて構築した Decision Tree

RQ2：予測方法

- RQ2：マージに関する特徴量を使用してコンフリクトを予測できるか？
- Hyper-parameters
 - グリッドサーチにより探索し，最も精度の高い値を設定
- Combination operators
 - [最小値,最大値, 平均, 中央値, Norm-1, Norm-2]を全て試行し，最も精度の高い値性よう
 - **すべてのデータでNorm-2が最も精度が高い**
- 評価方法
 - Precision, Recall, F1 (10-cross validation)

RQ2 : 結果

TABLE V
MERGE CONFLICT PREDICTION RESULTS. THE HIGHEST VALUES IN EACH CATEGORY ARE HIGHLIGHTED.

Programming Language	Classifier	Safe (Not Conflicting)			Conflicting		
		$Precision_S$	$Recall_S$	$f1-score_S$	$Precision_C$	$Recall_C$	$f1-score_C$
C	<i>Baseline #2</i>	1.00	0.81	0.90	0.28	1.00	0.44
	<i>Decision Tree</i>	0.99	0.89	0.94	0.37	0.88	0.52
	<i>Random Forest</i>	0.98	0.96	0.97	0.56	0.72	0.63
C++	<i>Baseline #2</i>	1.00	0.82	0.90	0.34	0.99	0.51
	<i>Decision Tree</i>	0.99	0.88	0.93	0.41	0.91	0.57
	<i>Random Forest</i>	0.97	0.96	0.97	0.63	0.68	0.66
C#	<i>Baseline #2</i>	0.99	0.83	0.90	0.32	0.92	0.48
	<i>Decision Tree</i>	0.99	0.85	0.92	0.35	0.90	0.51
	<i>Random Forest</i>	0.97	0.93	0.95	0.48	0.74	0.57
Java	<i>Baseline #2</i>	1.00	0.85	0.92	0.36	0.99	0.53
	<i>Decision Tree</i>	0.99	0.90	0.94	0.44	0.93	0.60
	<i>Random Forest</i>	0.98	0.95	0.97	0.58	0.83	0.68
PHP	<i>Baseline #2</i>	1.00	0.83	0.91	0.38	0.99	0.55
	<i>Decision Tree</i>	0.99	0.87	0.93	0.44	0.93	0.59
	<i>Random Forest</i>	0.98	0.93	0.95	0.54	0.82	0.65
Python	<i>Baseline #2</i>	1.00	0.82	0.90	0.29	1.00	0.45
	<i>Decision Tree</i>	1.00	0.87	0.93	0.36	0.95	0.52
	<i>Random Forest</i>	0.98	0.94	0.96	0.49	0.74	0.59
Ruby	<i>Baseline #2</i>	1.00	0.84	0.91	0.33	1.00	0.50
	<i>Decision Tree</i>	1.00	0.89	0.94	0.41	0.96	0.57
	<i>Random Forest</i>	0.98	0.96	0.97	0.59	0.72	0.65

RQ2の結論：

ランダムフォレストが安定して高い精度を示す。

Safeを予測することはConflictingを予測することよりも安易

考察：手法の有効性

- 有効性
 - 高いRecallでコンフリクトを予測することができる
- 欠点
 - Precisionが低い
 - ▶ Precisionが低いことは、頻繁にコンフリクトを予測し、早期にマージを促すこととなるため結果的にコンフリクトを防ぐことにつながる

まとめ・所感

- まとめ

- マージを行う際のコンフリクトを防止するための予測手法を構築
- 744プロジェクトを対象とした，これまでで最も大規模な研究

- 所感

- 既存研究を元に作成したメトリクスはまとまっていて参考になりそう
- 相関と効果量の両方を測定している点が参考になる