

Recommending Code Improvements Based on Stack Overflow Answer Edits

Chaiyong Ragkhitwetsagul, Matheus Paixao

19TH International Conference On
Mining Software Repositories (MSR 2022)

発表者：井町 瑠貴

背景

- | Stack Overflow(SO)は、開発者が質問、コードスニペットの共有をするためのWebサイト
- | 開発者は、SOの回答からコードを再利用するのが一般的である

背景

- | SOには最適ではない回答が投稿され、最適化されるまで時間がかかることがある
- | その間、開発者は最適でないスニペットを使用する可能性がある

本研究では、SO上の回答編集に基づき、最適でないコードの改善を推奨する Macha を提案

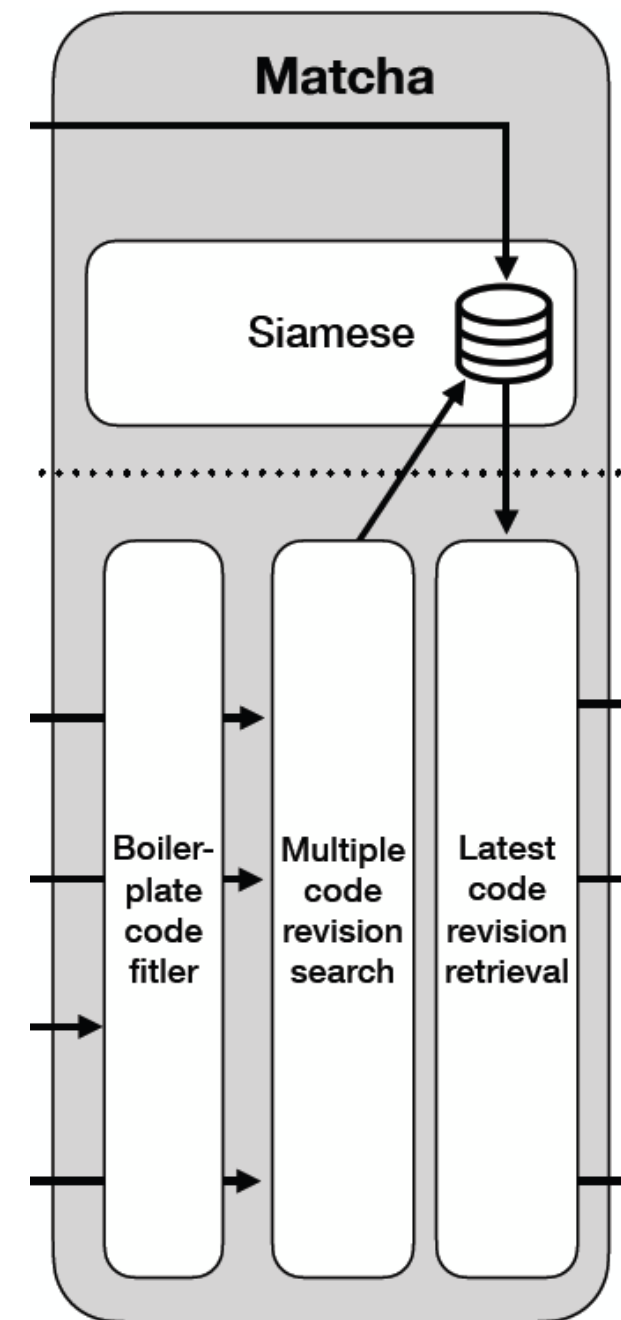
Matchaの設計

コード推薦システム

- 入力：Javaプロジェクト
- 出力：SOの回答から類似したスニペットの最新更新のもの

Siamese code clone search engine^[20]を拡張

- Boiler-plate Code Filter
- Multiple code revision search
- Latest code revision retrieval



[20] Chaiyong Ragkhitwetsagul and Jens Krinke. 2019. Siamese: scalable and incremental code clone search via multiple code representations. Empirical Software Engineering 24 (8 2019), 2236–2284. Issue 4.

Matcha | Boiler-plate Code Filter

boiler-plate code

- 複数の場所で繰り返される定型コード
- 開発者にとって有用ではない

作成方法

- 先行研究^[22]の分類に従って、パターンをリストアップ
- パターンにマッチする正規表現のリストを作成

[22] Chaiyong Ragkhitwetsagul, Jens Krinke, Matheus Paixao, Giuseppe Bianco, and Rocco Oliveto. 2021. Toxic Code Snippets on Stack Overflow. IEEE Transactions on Software Engineering 47, 3 (2021), 560–581.

Matcha | Latest code revision retrieval

Multiple code revision search

- SOの回答で同じコードスニペットの複数のリビジョンを検索

Latest code revision retrieval

- 各コードスニペットクエリに対して、SOの回答の最新バージョンが返されているかどうかを確認

リサーチクエストション

RQ1 (Phase 0)

- ソフトウェアプロジェクトとSOの回答間で類似したスニペットを正確に検出するか

RQ2 (Phase 2)

- Matchaが提示する推奨事項は、開発者にとってどの程度有用なのか

RQ3 (Phase 3)

- Matchaの推奨は、実際にはどの程度受け入れられているのか

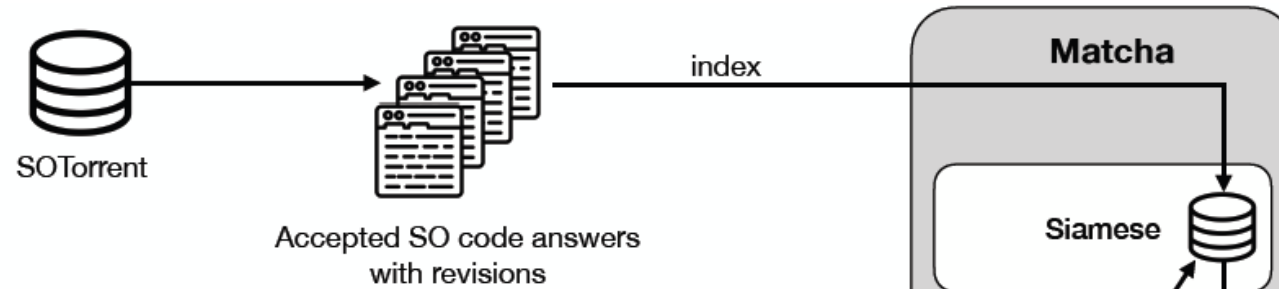
アプローチ | Phase 0

Matchaの健全性テスト

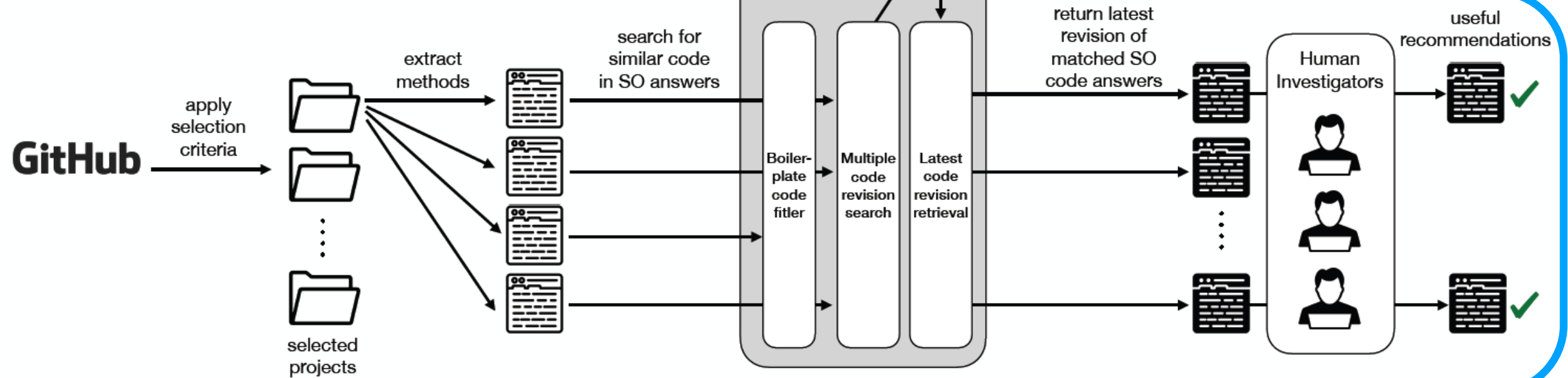
- Siamese と boiler plate code filter でどの程度の性能を発揮するかを評価
- SOTorrentデータセットのPostReferenceGHテーブルに含まれる情報を正解データとする
- データ抽出の粒度
 - GitHub : メソッド
 - SO : コードスニペット全体

アプローチ | 概要

1



2



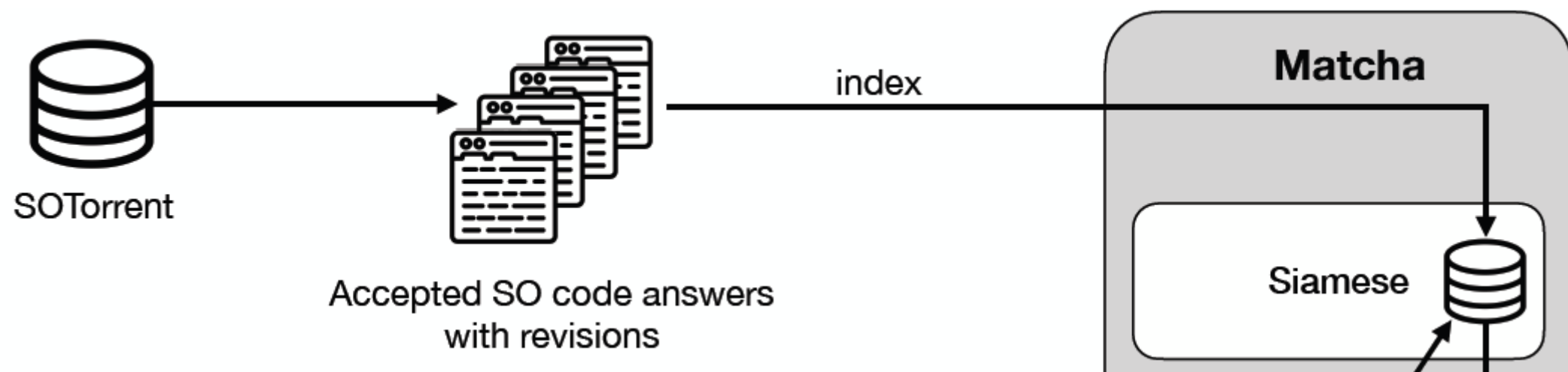
3



アプローチ | Phase 1

SOの回答からコードスニペットをSiameseでインデックス化

- SOTorrent データセットの最新版を利用
- Javaの回答から少なくとも1つのリビジョンがあるコードスニペットを抽出
- 抽出されたスニペットをSiameseにインポート



アプローチ | Phase 2

| SOの回答編集からコードの更新を推奨

- GitHubプロジェクトのセットを評価データセットとして選択
- 各プロジェクトに対してMatchaを実行し、どの推奨が開発者にとって有用であるかを評価

| プロジェクトの質に関する仮定

- 質が高いプロジェクト
 - 最適でないスニペットは少ない → Matchaによる影響が小さい
- 質の低いプロジェクト
 - 最適でないスニペットが多い → Matchaによる影響が大きい

アプローチ | Phase 2

GitHub Search^[5]

- GitHubリポジトリを検索するためのツール
- 各プロジェクトに対して25の特性データを提供
本研究では以下の条件でフィルタリング
 - 言語 : Java
 - 未解決のIssue, Pull Requestがある
 - フォークは除外

[5] Ozren Dabic, Emad Aghajani, and Gabriele Bavota. 2021. Sampling Projects in GitHub for MSR Studies. In Proceedings of the 18th International Conference on Mining Software Repositories (MSR '21). 560–564.

アプローチ | Phase 2

プロジェクトの選択基準

- プロジェクトの品質基準として3つの人気度指標を用いる

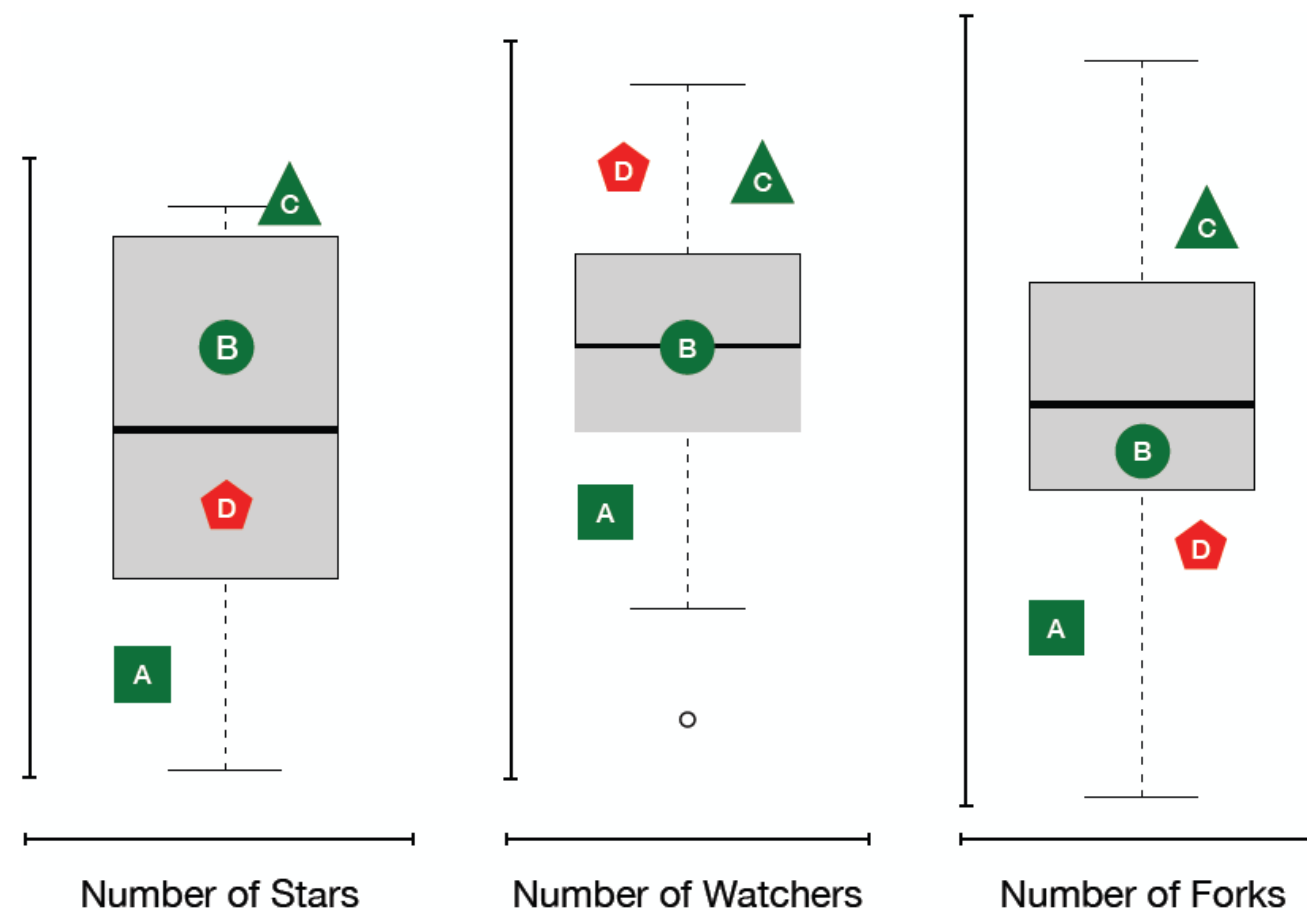
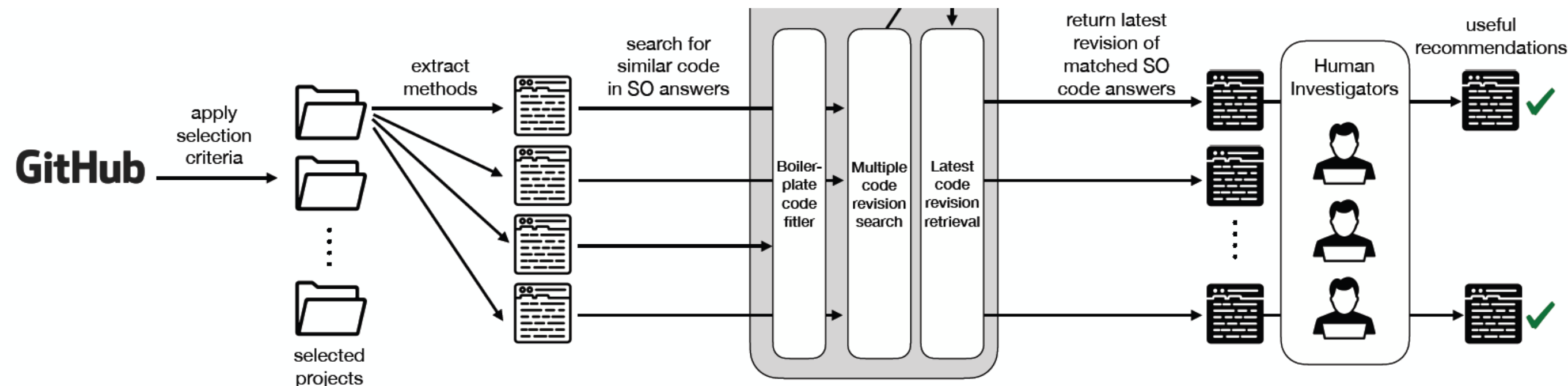


Figure 2: GitHub project selection criteria based on the distributions of number of stars, watchers, and forks.

アプローチ | Phase 2

Matchaの実行

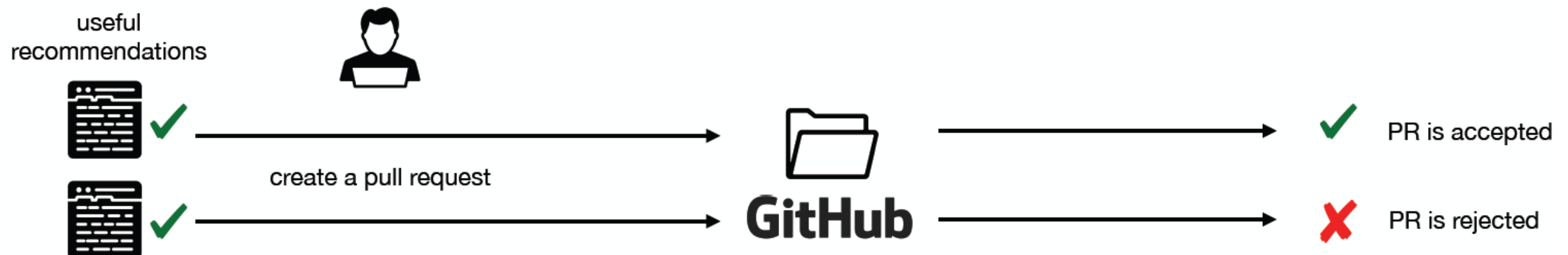
- 与えられたすべてのプロジェクトのコードを抽出しフィルタリング
- Siameseが最新でないスニペットを返した場合、推奨のためにマークする
- マークされたメソッドに対して最新の回答を返す



アプローチ | Phase 3

Matchaの推奨をプルリクエストとして提出

- 以下の条件を満たすプロジェクトに，推奨された変更点を反映したプルリクエストを提出
 - 自動化されたテストスイートを提供
 - 推奨された変更を加えた後，すべてのテストスイートで成功
- プルリクエストの説明文
 - SOの編集コメントから引用
 - 自分たちで作成



結論

| SOの回答更新に基づき、プロジェクトのコード変更を推奨できるMatchaを提案

| 今後の予定

- Matchaが提供するコード推奨の有用性を手動で検証
- 有用な推奨を含むプルリクエストをGitHubプロジェクトに提出することにより評価

所感

- この研究では、プロジェクトに対応した回答の最新バージョンを取得するまでが自動であるため、推奨の負担は改善の余地があると感じた
- SO内外の紐づけにコードクローン検出技術を用いていることが個人的になるほどと感じた