

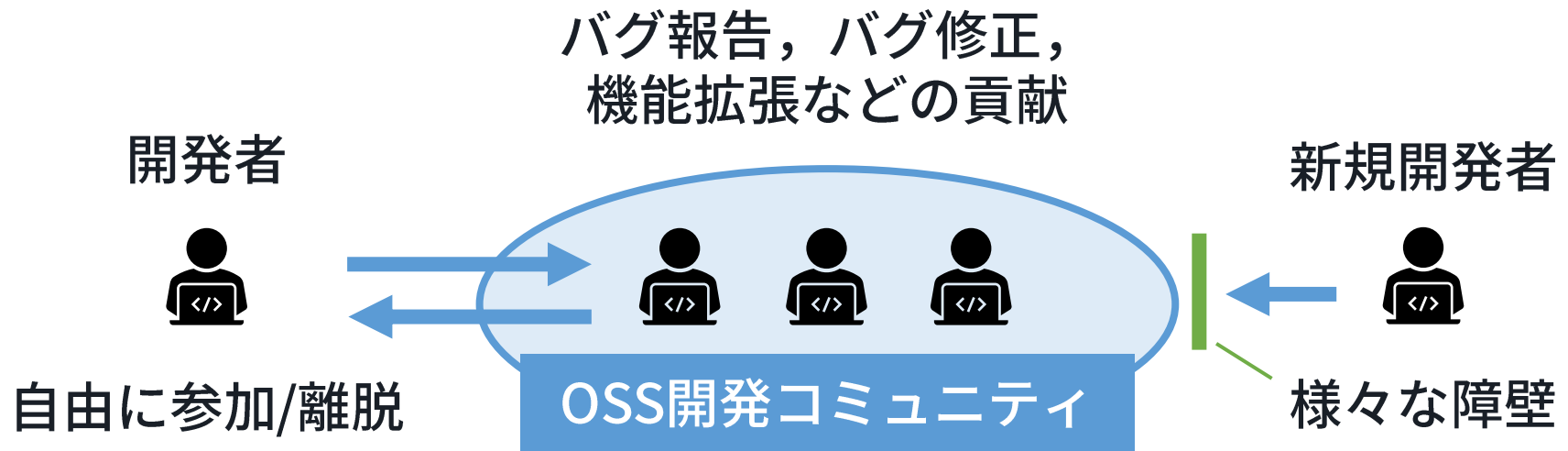
OSSプロジェクトへの オンボーディング支援のための Good First Issue自動分類

M2 堀口 日向

研究発表

2021/10/6

1. 背景



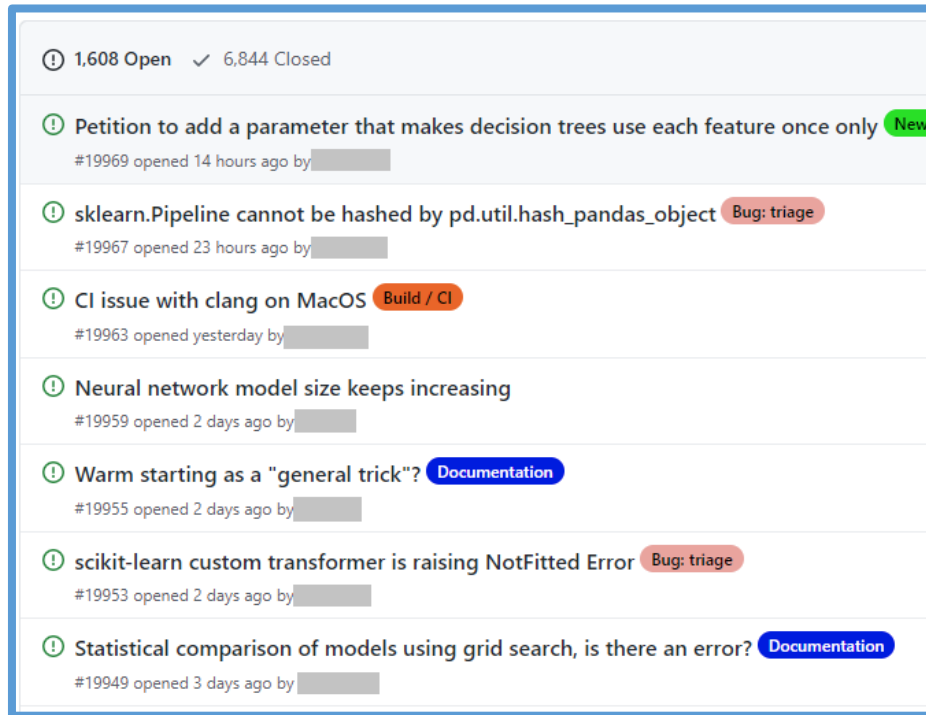
- プロジェクトを持続可能にするため、新規開発者からの貢献を求めている^[1, 2]
- 新規開発者がコミュニティに参加する際、様々な障壁に直面^[3, 4]

[1] Israr Qureshi and Yulin Fang. Socialization in Open Source Software Projects: A Growth Mixture Modeling Approach. Organizational Research Methods, pp. 208–238, 2011.
[2] Forte, A. and Lampe, C. Defining, Understanding, and Supporting Open Collaboration: Lessons From the Literature. American Behavioral Scientist 57, 5 2013, pp. 535–547.
[3] I. Steinmacher, T. Conte, M. A. Gerosa, and D. Redmiles, “Social barriers faced by newcomers placing their first contribution in open source software projects,” in CSCW ’15, 2015, pp. 1379–1392.
[4] C. Hannebauer and V. Gruhn, “On the relationship between newcomer motivations and contribution barriers in open source projects,” in OpenSym ’17, 2017, pp. 1–10.

1. 背景 | 課題選択の障壁

貢献を行う際、プロジェクトに存在するIssue(課題)の中から
選択して取り組む

Issue一覧の例 (Scikit-learn)



新規開発者



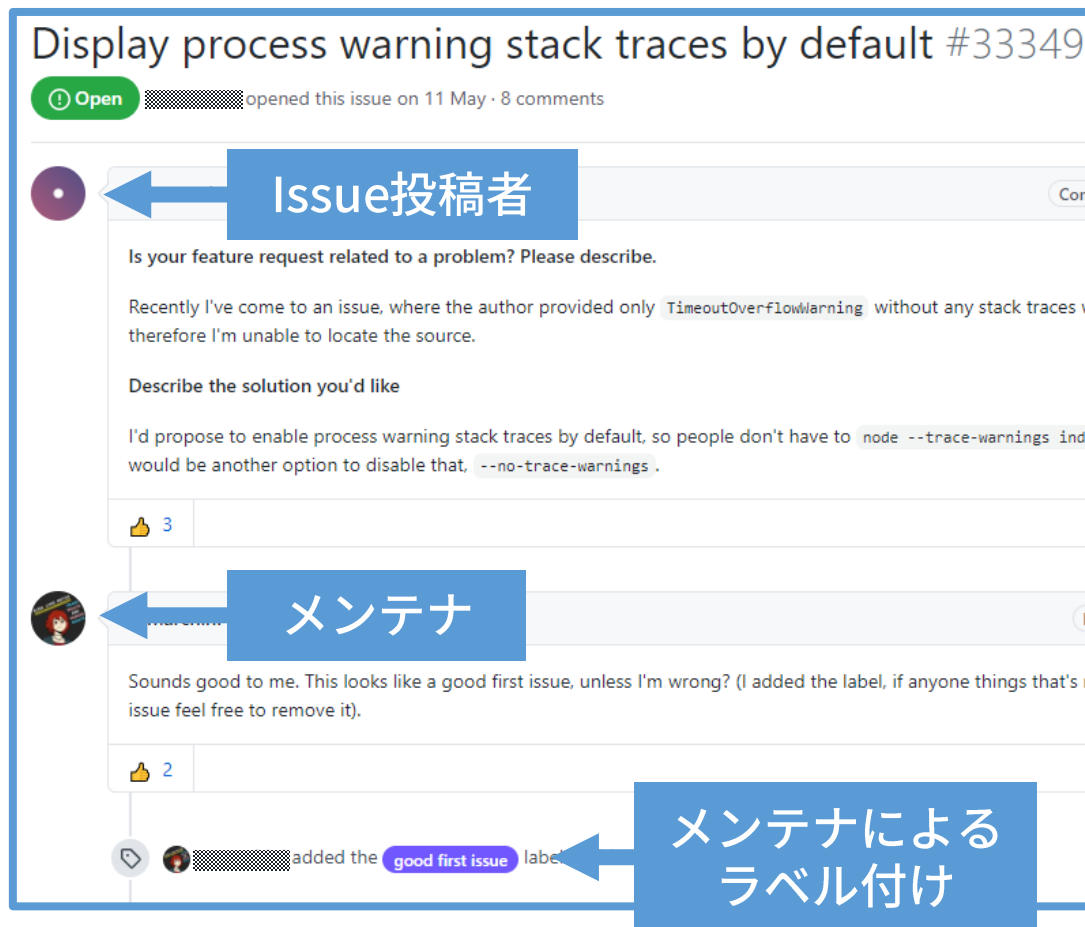
Issueの理解，対処に
時間がかかる^[5]

⋮

[5] I. Steinmacher, T. U. Conte and M. A. Gerosa, Understanding and Supporting the Choice of an Appropriate Task to Start with in Open Source Software Communities, 48th Hawaii International Conference on System Sciences, pp. 5299-5308, 2015

1. 背景 | Good First Issue (GFI)

GFIの例 (Node.js)



一部プロジェクトでは
新規開発者向けのIssueに
「**Good First Issue (GFI)**」

ラベルを付与



障壁を軽減

メンテナにとって負担^[6]

[6] <https://github.blog/2020-01-22-how-we-built-good-first-issues/>

2. 目的とアプローチ

目的

新規開発者向けのIssueを分類する機械学習モデルを構築

アプローチ

1. データセットの作成

- 特定のOSSプロジェクトからGFIと通常のIssueを収集
- Issue解決者の経験に基づいたフィルタリング
- データ拡張 (Data Argumentation)

2. 深層学習

- Attention-based LSTM (単語レベル)
- CNN (文字レベル)

試行錯誤中…

3. データセットの作成

特定のOSSプロジェクトからGFIと通常のIssueを収集

- **単一プロジェクトに限定**して学習

理由：プロジェクトごとにGFIの基準が異なる可能性

- 対象プロジェクトの選定方法

1. GFIの保有数が多い順にソート

2. プロジェクトを目視

- GFI数 > 通常Issue数 になっていないか
- 整合性のあるデータを収集可能か

→ Material-UI※（WebUIフレームワーク）を利用

	データ件数[件]
通常Issue (負例)	12,164
GFI (正例)	1,220

3. データセットの作成

前処理

Issue投稿時に含まれる情報のみを利用

→ タイトル, 本文, 投稿者

- タイトル, 本文
 - 単語レベルに分割 (One-Hotベクトルに変換)
 - 単語数 (数値)
 - 説明の丁寧さに関連
- 投稿者
 - プロジェクトへの権限レベル (One-Hotベクトルに変換)
 - コアメンバーの場合, 新規開発者を惹きつけるために多くのGFIを投稿する可能性

3. データセットの作成

前処理

Issueは
Markdown形式で記述

コード片

インラインコード
問題の詳細な説明

Issueの例 (<https://github.com/pandas-dev/pandas/issues/39701>)

commented on 10 Feb • edited ▾

☒ I have checked that this issue has not already been reported.
☒ I have confirmed this bug exists on the latest version of pandas.
☐ (optional) I have confirmed this bug exists on the master branch of pandas.

Note: Please read [this guide](#) detailing how to provide the necessary information for us to reproduce your bug.

Code Sample, a copy-pastable example

```
import contextlib, io, json
import pandas as pd

redirect = io.StringIO()
with contextlib.redirect_stdout(redirect):
    pd.show_versions(as_json=True) # produces invalid json
pd_versions = redirect.getvalue()
pd_versions = json.loads(pd_versions) # raises JSONDecodeError
```

Problem description

`pd.show_versions(as_json=True)` is supposed to print valid JSON, but it really doesn't. It is supposed to work as advertised but it doesn't. It prints invalid JSON that is not parsable. The following error is printed when attempting to parse it:

```
json.decoder.JSONDecodeError: Expecting property name enclosed in double quotes: line 1 column 2 (char 1)
```

This functionality is important because when running a job in the cloud, it is not feasible to print a multiline output. This is because every printed line has significant attached metadata. It is better to have valid JSON output instead in this case. The implementation of the `show_versions` function looks to be very hacky and poorly reviewed, given that it fails in this very basic way.

- コード片，インラインコード，ハイパーリンクなどの置換
- ストップワードや記号の除去，数字や単語の正規化など

4. 深層学習モデルの構築

Attention-based LSTM

各単語を入力したときのLSTMの出力を重みづけして、
最終的な出力を計算

→ LSTMよりも長期的な依存関係（文脈）を扱える

- 入力部分

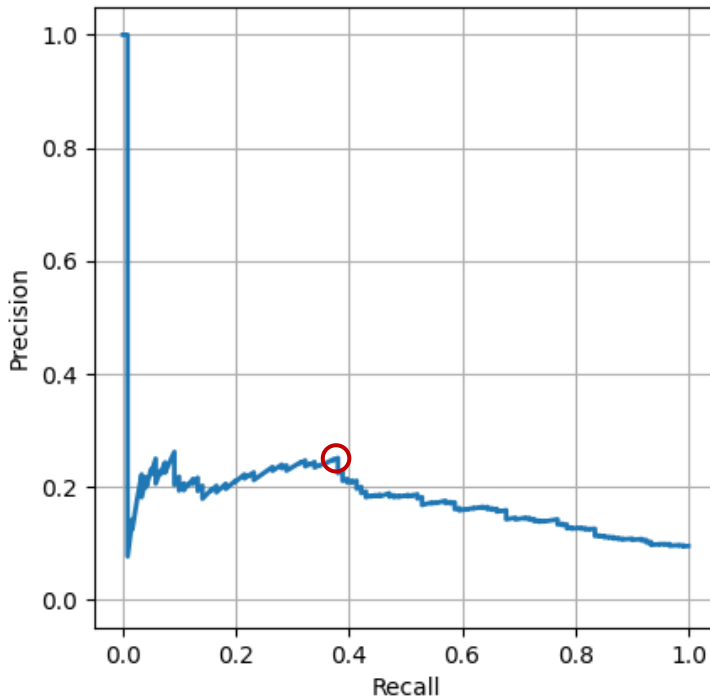
- 埋め込み層を利用して、単語のOne-Hotベクトルを128次元のベクトルに埋め込み、LSTMに渡す

- 出力部分

- Attention-based LSTMの出力を1024次元の全結合層に渡す
- 上の出力を1次元の全結合層に入力し、シグモイド関数で0~1の確率として出力（0：通常Issue，1：GFI）

5. 結果

- 不均衡データの分類モデルのため、Precision-Recall曲線で評価
- 各データポイントは、ある閾値の時のRecallとPrecision
右に行くほど閾値が低くなり、Recall $\uparrow$ Precision $\downarrow$



Precision : 0.23
Recall : 0.39

2. 目的とアプローチ

目的

新規開発者向けのIssueを分類する機械学習モデルを構築

アプローチ

1. データセットの作成

- 特定のOSSプロジェクトからGFIと通常のIssueを収集
- Issue解決者の経験に基づいたフィルタリング
- データ拡張 (Data Argumentation)

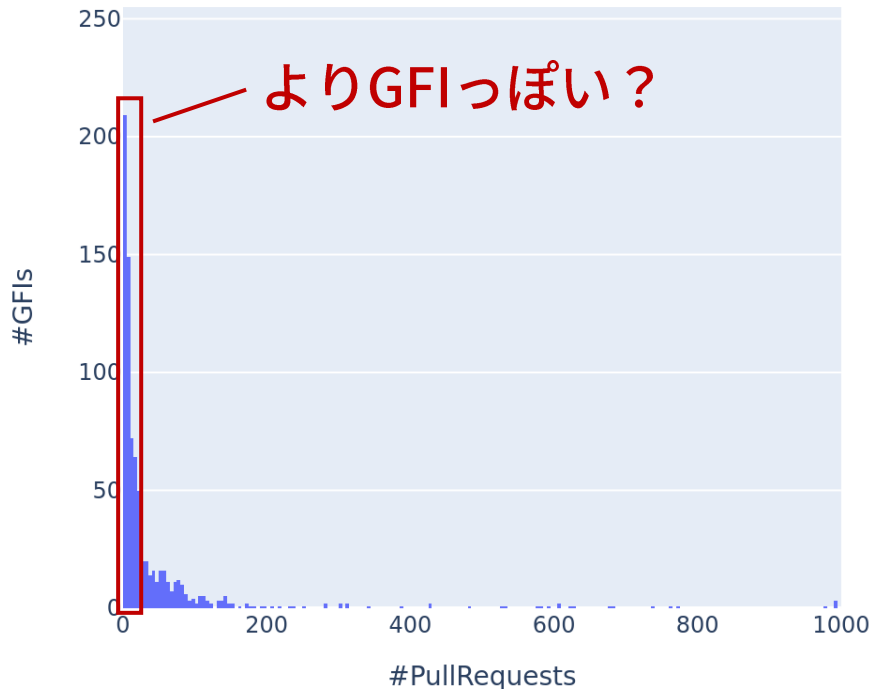
2. 深層学習

- Attention-based LSTM (単語レベル)
- CNN (文字レベル)

試行錯誤中…

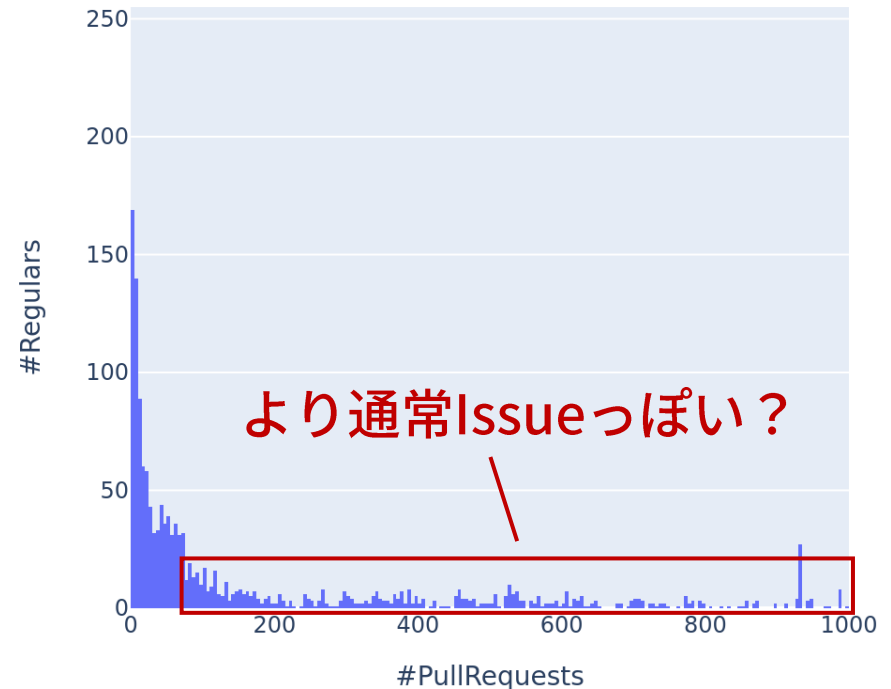
6. 現在取り組んでいること

Issue解決者の経験に基づいたフィルタリング



←
経験の少ない
開発者が
解決したGFI

→
経験豊富な
開発者が
解決したGFI



←
経験の少ない
開発者が解決した
通常Issue

→
経験豊富な
開発者が解決した
通常Issue

7. 今後試してみたいこと（アイデア段階）

- データ拡張※¹（Data Argumentation）
 - 少ないデータ数で高い精度が得られる
 - データセットをフィルタするとデータ数が減るので有効かも
- 文字レベルの深層学習モデル※²
 - 前処理でのコード片置換
 - モジュール名など，Issueの難易度に関連する情報が欠落
 - コード片をトークン化するのは手間…
 - 文字レベルの深層学習モデルを使えば解決できそう

※¹ EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks, <https://arxiv.org/abs/1901.11196>

※² Character-level Convolutional Networks for Text Classification, <https://arxiv.org/abs/1509.01626>