ソフトウェア開発における ドキュメントの更新必要性の 予測に向けた，ドキュメントと ソースコードの紐付け

オープンソースソフトウェア工学研究室
学籍番号　60230010
氏名　井町　瑠貴

背景｜ドキュメント

システムや関連するプロセスの有用な知識を提供

- ユーザはドキュメントを参照し利用している
- Markdownファイル（拡張子：md）に記述されることが多い
  - 本研究では，Markdownファイルをドキュメントとして扱う

Binomial logistic regression

For more background and more details about the implementation of binomial logistic regression, refer to the documentation of logistic regression in `spark.mllib`.

Examples

The following example shows how to train binomial and multinomial logistic regression models for binary classification with elastic net regularization. `elasticNetParam` corresponds to α and `regParam` corresponds to λ.

Python **Scala** **Java** **R**

More details on parameters can be found in the Python API documentation.

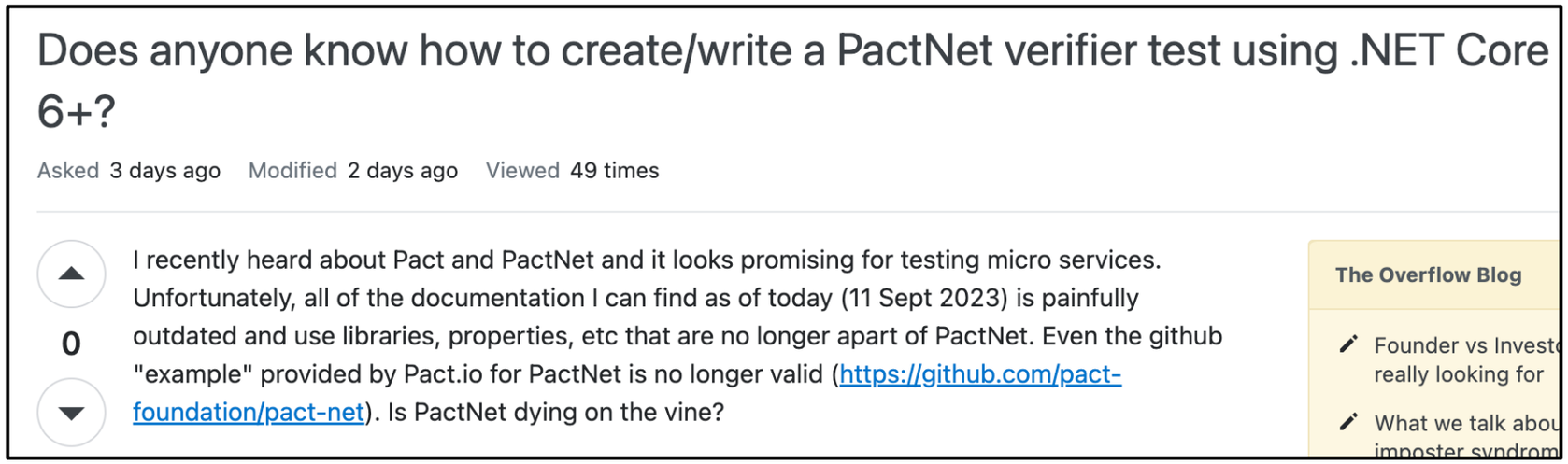
```
from pyspark.ml.classification import LogisticRegression

# Load training data
training = spark.read.format("libsvm").load("data/mllib/sample_libsvm_data.txt")
```

ドキュメントの例[1]

[1] https://spark.apache.org/docs/latest/ml-classification-regression.html

背景｜コードとドキュメントの関係

- ドキュメントの更新はソースコードの更新に追従して行われることが望ましい
  - しかし，ドキュメントの更新が行われないまま，放置されてしまうことがある
- 下の例では，Pactというテストツールのドキュメント更新の遅れが原因の質問が投稿されている



ドキュメントの更新遅れが原因のSO上での質問[2]

[2] https://stackoverflow.com/questions/77083190/does-anyone-know-how-to-create-write-a-pactnet-verifier-test-using-net-core-6

背景｜更新の遅れによる悪影響

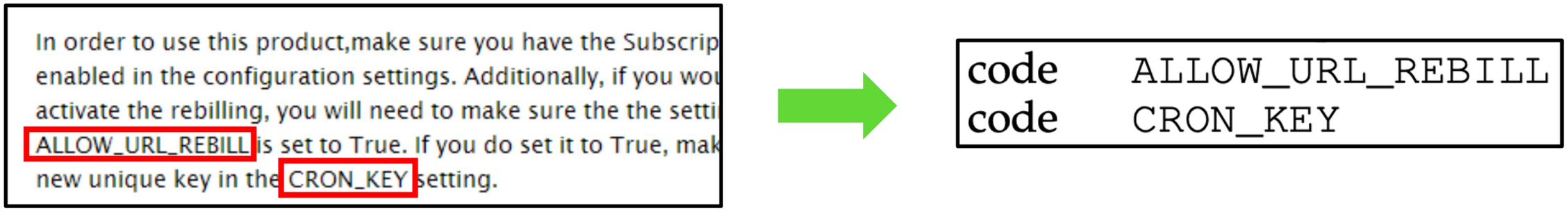
既存研究より，ドキュメントの更新が遅れると
以下のような問題が起こることが明らかになっている

- ドキュメント参照者の混乱を招く[4]
- ソフトウェアの老朽化を引き起こす[5]
- ユーザの学習コストが増加する[6]

[4] Mira Kajko-Mattsson. 2005. A survey of documentation practice within corrective maintenance. Empirical Software Engineering 10, 1 (2005), 31–55.
[5] David Lorge Parnas. 1994. Software aging. In Proceedings of International Conference on Software Engineering. 279–287.
[6] Igor Steinmacher, Christoph Treude, and Marco Aurélio Gerosa. 2018. Let me in: Guidelines for the successful onboarding of newcomers to open source projects. IEEE Software 36, 4 (2018), 41–49.

先行研究｜古くなったドキュメントの検出

- code elementを用いて，ドキュメントの古い参照箇所を検出[7]
  - code element：ドキュメントに含まれる変数，関数，クラス名などのコード要素



code elementの例[8]

[7] Wen Siang Tan and Markus Wagner and Christoph Treude. 2022. Detecting Outdated Code Element References in Software Repository Documentation. arXiv:2122.01479.
[8] C. Treude, M. P. Robillard and B. Dagenais, "Extracting Development Tasks to Navigate Software Documentation," in IEEE Transactions on Software Engineering, vol. 41, no. 6, pp. 565-581, 1 June 2015, doi: 10.1109/TSE.2014.2387172.

先行研究｜古くなったドキュメントの検出

アプローチ

- ドキュメントからcode elementを抽出
- プロジェクトからcode elementがある数をコミットごとにカウント
- n個（n>0）あった参照が0個になったとき，ドキュメントが古くなったと判断

出力例

code element	R37	R38	R39	R40
renderFiles(‘./files’)	3	3	0	0

ここでドキュメントが古くなっている

先行研究｜課題点

- 先行研究は，ソースコードから削除されたcode elementがドキュメントに存在することを検出
  - ソースコードに新たに追加されたcode elementに対する説明がドキュメントに存在するか検出できない

- 本研究では，ドキュメントの情報が不足している状態を検出することを目指す

コード	ドキュメント
○	○
○	×
×	○
×	×

本研究の検出可能範囲

先行研究の検出可能範囲

研究目的・アプローチ

研究目的

- コードとドキュメントの紐付け

アプローチ

- 以下の2つの観点から紐付けを検討
  - 空間的観点（テキスト情報）
    - 固有表現抽出
    - code element（正規表現によるコード関連要素の抽出）
  - 時間的観点
    - 更新の共起関係

データセット

- apache/sparkプロジェクトのコミットデータ
  - 2023年6月23日時点のデータ
  - データの概要は表で表記する
    - 総コミット数
    - ドキュメントのコミット数
    - 他に使う情報のデータをまとめる

アプローチ｜固有表現抽出

固有表現抽出（NER）

- ドキュメントから固有表現を抽出し，抽出した固有表現が他のファイルにどれほど存在するのか調査
- 仮説：関連するファイル間では共通の表現が多く存在する（根拠は？）
- いくつかの学習済みモデルがあるので，それぞれ試してみる？
  - spaCy
  - Hugging face（bert-base-NER）
- 入出力のイメージ
  - 入力：対象ドキュメントの全文（要：前処理）
  - 出力：単語のリスト（ラベルはとりあえず触らない？）

結果｜固有表現抽出

固有表現抽出の結果（表，実際のペアの例）

- そもそもどのような表現が抽出されたのか
- それぞれの表現の存在状況
  - 閾値を設定するなどして関連ファイルを特定できるのか
  - どのようなものがペアとして特定できるのか

アプローチ｜code element

code element（ce）

- ドキュメントからceを抽出し，抽出したceが他のファイルにどれほど存在するのか調査
  - ceは既存研究の正規表現リストを用いて抽出
- 仮説：関連するファイル間では共通のceが関連しないものに比べて多く存在する（根拠は？）
- どのファイルにも存在するようなceは除去したい
  - TF-IDFなどで重要度の高いものだけ扱うようにする？
  - イメージとしては，重要度の低いもの（どこにでもあるようなce）を取り除く

結果｜code element

- ceの結果（表，実際のペアの例）
  - そもそもどのような語句が抽出されたのか
    - どのような語句は除去する（した）のか
  - それぞれの語句の存在状況
    - 閾値を設定するなどして関連ファイルを特定できるのか
    - どのようなものがペアとして特定できるのか

結果｜空間的観点

空間的観点に着目した手法のまとめ

- 行った手法の比較
  - 抽出できた語句の違い
  - 特定できたペアの違い
  - 図を用いて具体的に説明

考察｜空間的観点

結果に対する考察

- なぜ？の説明
  - 抽出できた語句について
  - 特定できたペアについて
  - 図を用いて具体的に説明できればそれも

アプローチ｜時間的観点

更新の共起関係

- ある一定の期間内に変更が行われるコードとドキュメントのペア（とりあえず1対1関係）を探す
  - いくつか閾値を設けて結果を見比べる
- 仮説：関連するファイルは同時期に変更が加えられる
  - 閾値を短くすればするほど重大な変更（認識漏れが起こりにくい？）
- リリース直前にドキュメントをまとめて変更するプロジェクトが存在する可能性が考えられるため，以下の事前調査を実施
  - リリースタイミング（メジャー・マイナー）と，ドキュメントが含まれるコミットが行われたタイミングのプロット

アプローチ｜時間的観点

更新の共起関係

- 前スライドの内容を図示

結果｜時間的観点

結果の図示

- 行った手法の比較
  - 抽出できた語句の違い
  - 特定できたペアの違い
  - 図を用いて具体的に説明

今後の予定