

Topic Recommendation for GitHub Repositories: How Far Can Extreme Multi-Label Learning Go?

Ratnadira Widyasari, Zhipeng Zhao, Thanh Le Cong, Hong Jin Kang, David Lo
School of Computing and Information Systems, Singapore Management University

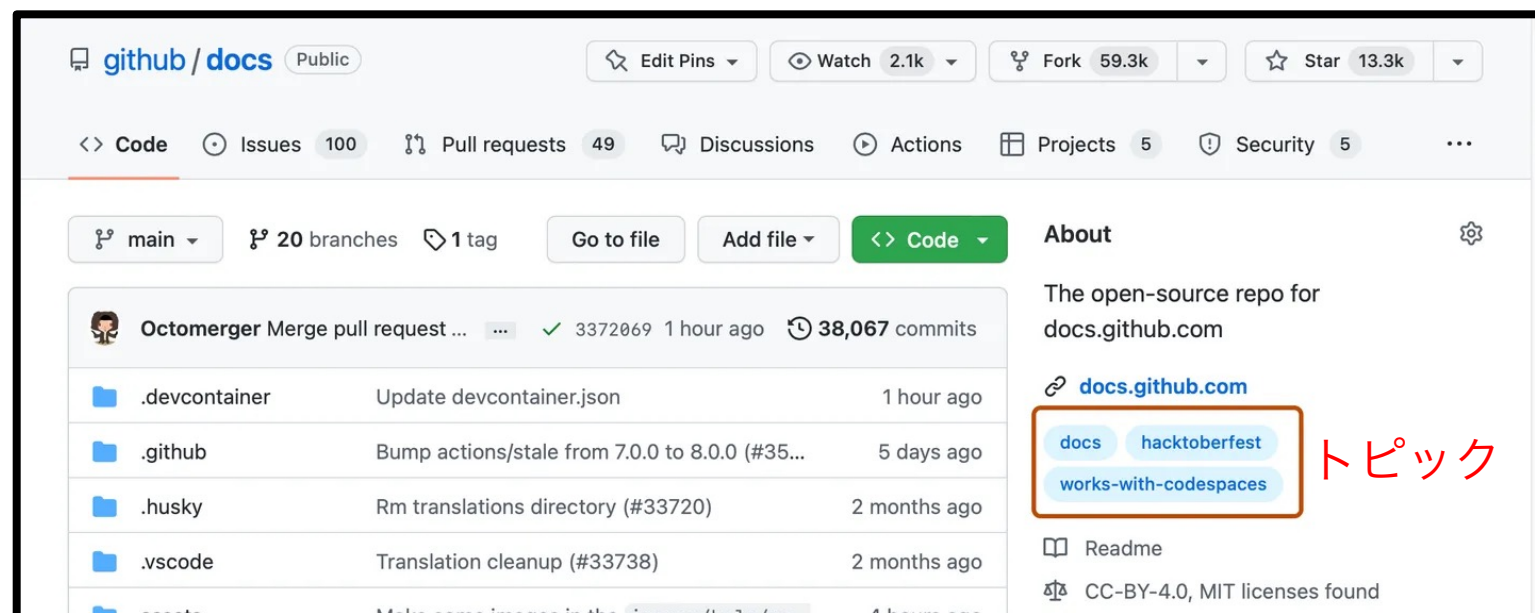
Proceedings of the 30th International Conference on Software Analysis,
Evolution and Reengineering (SANER '23), pp167-178

発表者：井町 瑠貴

背景 | GitHub Topic

GitHubでは、2017年からリポジトリにトピック（タグ）を付与することができる

- リポジトリの詳細を伝える役目
 - リポジトリの説明, 目標, 採用ライブラリ など



トピックの例^[1]

[1] <https://docs.github.com/ja/repositories/managing-your-repositorys-settings-and-features/customizing-your-repository/classifying-your-repository-with-topics>

背景 | トピックの割り当て

トピック数や、トピックを利用するリポジトリ数は日々増加している

- リポジトリ数：2020年から2022年で、200万→700万

適切にトピックを割り当てることは重要

- 開発者がリポジトリを発見しやすくなる
- しかし、トピック数（種類）が多いため、人手による適切な割り当ては困難

先行研究 | トピックの割り当て

既存研究では、リポジトリの説明や、READMEに基づいてトピック推薦を試みている

- マルチクラス分類
- マルチラベル分類

マルチクラス分類

- ナイーブベイズを用いて、トピックがリポジトリに関連する確率を予測^[11]
- 後述のマルチラベル分類手法を下回ることが実証されている

[11] C. Di Sipio, R. Rubei, D. Di Ruscio, and P. T. Nguyen, “A multinomial naive bayesian (mnb) network to automatically recommend topics for github repositories,” in Proceedings of the Evaluation and Assessment in Software Engineering, 2020, pp. 71–80.

マルチラベル分類

マルチラベル分類

- 1つのサンプルが、複数のラベルを取り得るという設定を扱う問題
- ラベル付けを行いたいサンプルをモデルに入力し、それぞれのラベルに対する確率を計算

マルチラベル分類の特徴

- 以下の場合、正しく分類できない場合がある
 - ラベル間に不均衡が生じている（データ量に差がある）
 - ラベルの確率がデータ量に影響を受けるため

先行研究 | マルチラベル分類

マルチラベル分類

- リポジトリのトピック推薦タスクをマルチラベル分類 (MLC) 問題として扱う^[6]
- 頻度の低い（登場回数が100回未満）トピックはデータセットから削除
 - 前述の特徴のため

頻度の低いトピックを考慮する
必要があるのではないのか？

[6] M. Izadi, A. Heydarnoori, and G. Gousios, “Topic recommendation for software repositories using multi-label classification algorithms,” Empirical Software Engineering, vol. 26, no. 5, pp. 1–33, 2021.

Extreme Multi-Label Classification

XML (Extreme Multi-Label Learning) 問題

- 膨大なラベルを用いて行うマルチラベル分類問題
- 以下のような特徴が挙げられる
 - データの疎密
 - 各ラベルのデータ量が少ない場合がある

トピック推薦タスクを
XML問題として解決できるのでは？

研究目的・アプローチ

研究目的

- GitHubリポジトリからトピックを推薦する

アプローチ

- 低頻度なトピックの削除がデータセットに与える影響の調査
- タスクをXML問題として扱う

RQ

RQ1：低頻度トピックの除去が与える影響とは
－ 先行研究で行われた前処理が与える影響の調査

RQ2：XML手法は有効か

RQ3：従来手法とXML手法のパフォーマンスに
どのような違いがあるのか

RQ4：低頻度トピックにおいて，先行研究と本研究
の手法にどのような違いがあるのか

利用するXMLモデル

教師ありXML

- Parabel_[29]
 - 木構造とone-vs-all の組み合わせ
- Bonsai_[30]
 - 基本的にはParabelと同じ, 浅く広い木構造になるよう改良
- LightXML_[22]
 - 深層学習モデル

ゼロショットXML

- ZestXML_[24]
 - TF-IDFで文書とトピックをベクトル化し学習

[29] Y. Prabhu, A. Kag, S. Harsola, R. Agrawal, and M. Varma, “Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising,” in Proceedings of the 2018 World Wide Web Conference, 2018, pp. 993–1002

[30] S. Khandagale, H. Xiao, and R. Babbar, “Bonsai: diverse and shallow trees for extreme multi-label classification,” Machine Learning, vol. 109, no. 11, pp. 2099–2119, 2020.

[22] T. Jiang, D. Wang, L. Sun, H. Yang, Z. Zhao, and F. Zhuang, “LightXML: Transformer with dynamic negative sampling for highperformance extreme multi-label text classification,” Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 7987–7994, May 2021.

[24] N. Gupta, S. Bohra, Y. Prabhu, S. Purohit, and M. Varma, “Generalized zero-shot extreme multi-label learning,” in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, ser. KDD '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 527–535.

調査方法 | データセット

スター数1000個以上のリポジトリを収集

- トピック
- 文書：リポジトリの説明, READMEで構成

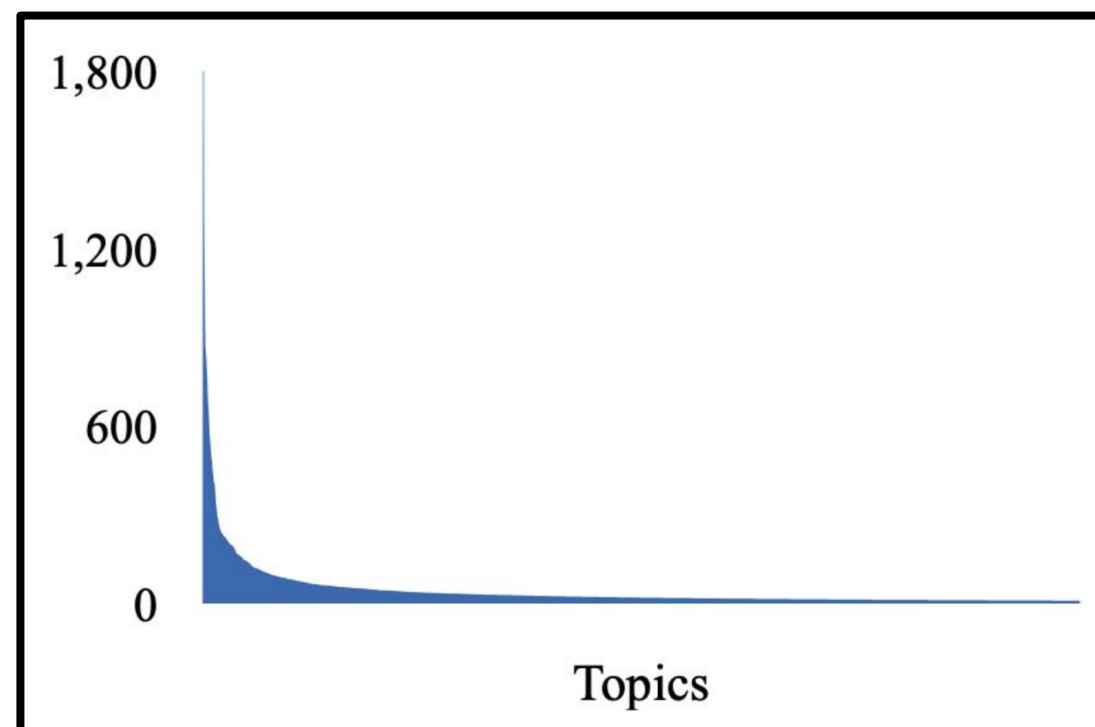
前処理を経て, 以下のデータセットを作成

- リポジトリ：21273件, トピック：37161件
- 80%：訓練用, 20%：テスト用に分割

Dataset	#GitHub Resipotories	#Topics
Training	17,018	31,973
Testing	4,255	12,912
Total	21,273	37,161

調査結果1 | 低頻度トピックによる影響

- トピックの頻度はロングテール分布に従う
 - 全トピックの内、低頻度トピックが占める割合は大きい
 - 20回未満のトピックを除去すると、3%のトピックしか残らない
 - 失われたトピックに重要なトピックが含まれている可能性がある



調査結果2 | XML手法は有効か？

ZestXMLが最も良い性能を発揮

- 平均F1値：0.39
- 平均SR：0.95
 - SR（成功率）：モデルの予測と一致するトピックが少なくとも1つあるリポジトリの割合

Model	Avg. F1	Avg. SR
Parabel	0.31	0.88
Bonsai	0.32	0.89
LightXML	0.31	0.89
ZestXML	0.39	0.95

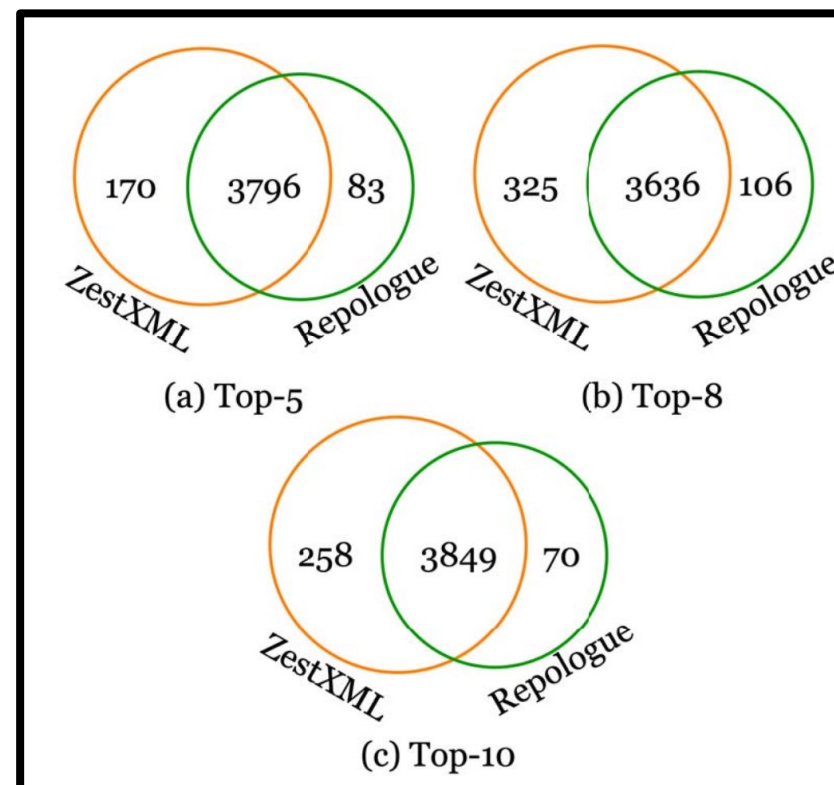
調査結果3 | 従来手法との比較

ZestXMLの方が良い性能を発揮

- 平均F1値：0.39
- 平均SR：0.95

Models	Avg. F1	Avg. SR
LR-TFIDF	0.33	0.90
ZestXML	0.39	0.95

ZestXMLの方がより多くのリポジトリの識別に成功



調査結果4 | 低頻度トピックにおける比較

低頻度トピックにおいて、ZestXMLは、ベースライン手法を上回った

- 平均F1値：3倍
- 平均SR：2倍

Model	Avg. F1	Avg. SR
LR-TFIDF	0.08	0.32
ZestXML	0.22	0.69

考察 | 予測結果の定性分析

従来手法では、高頻度のトピックに重点が置かれている

- “std”, “cmake”, など (c++に関連)

```
build hobbes you will need llvm 3 3 or later cmake 3 4 or later  
gnu gcc 4 8 or later and a version 2 5 or later linux kernel with  
llvm cmake and g installed after downloading this code you should  
be able to build and install hobbes just by running cmake make  
make install depending on where you ve installed llvm this might  
not work until you give cmake the path to llvm s cmake modules  
in particular you want the path to llvm s cmake file if you set  
the environment variable llvm dir to point to that directory they  
demonstrate many different aspects of the hobbes api embedding  
let s consider how to embed hobbes in a simple c program this code  
implements a very basic shell similar to hi c include include include  
int main hobbes cc c while std std cout std flush std string line  
std std line if line q break try c void print line catch std exception  
ex std cout ex what std cout std endl hobbes return 0 first to  
compile any expression we need to construct a hobbes cc object  
then in the context of an exception handler we can compile functions  
out of this hobbes cc object with the method giving it the type  
we expect back void in this case and also a string that we expect  
can compile to a value of that type if any stage of compilation  
fails parse error type mismatch etc then hi hi an interactive shell  
for hobbes type h for help on commands t t false boolean t c char  
t 0xff byte t short t 42 int t long t 42 0f float t 42 0 double  
and the types for record array variant and recursive values t 1  
10 int t foobar char t byte t name jimmy age 45 job programmer  
name char age int job char t jimmy 45 programmer char int char  
t cons function nil x char x also for qualified
```



考察 | 予測結果の定性分析

XML手法では、低頻度のトピックに重点を置くことができる

– “hobbes” に着目できている



まとめ

リポジトリのトピック推薦をXML問題として定式化し，有用性を調査

- 頻度の低いトピックを考慮できる
- 平均F1値を17.35%向上させることができた

今後の予定

- データの拡張
 - ソースコードやドキュメントの利用
- XMLとゼロショット学習の有用性を他のタスクでも調査
 - バグ管理 など
- GitHubにおけるタグ付け文化に関する実証実験

所感

なぜ提案手法でうまくいくのかの説明が不足しており、理由がいまいち理解できなかった

- 仕組みの説明が不足，自身の知識の不足
- zestXML以外のXML手法ではどうなる？

参考文献

マルチラベル分類モデル/多クラス分類モデルの閾値を最適な値に調整してみよう

- <https://nttdocomo-developers.jp/entry/202212081200>

数えきれないほどの分類を行うExtremeClassification

- <https://kamujun.hatenablog.com/entry/2018/06/22/180605>

The Extreme Classification Repository: Multi-label Datasets & Code

- <http://manikvarma.org/downloads/XC/XMLRepository.html>