

A Large-Scale Empirical Study of Open Source License Usage: Practices and Challenges

Jiaqi Wu
The State Key Laboratory of
Blockchain and Data Security,
Zhejiang University
Hangzhou, Zhejiang, China
jiaqiwu@zju.edu.cn

mailto:jiaqiwu@zju.edu.cn
Jia
Huawei
China
xin.xia@acm.org

Lingfeng Bao*
The State Key Laboratory of
Blockchain and Data Security,
Zhejiang University
Hangzhou, Zhejiang, China
lingfengbao@zju.edu.cn

Xiaohu Yang
The State Key Laboratory of
Blockchain and Data Security,
Zhejiang University
Hangzhou, Zhejiang, China
yangxh@zju.edu.cn

Xing Hu
The State Key Laboratory of
Blockchain and Data Security,
Zhejiang University
Hangzhou, Zhejiang, China
xinghu@zju.edu.cn

OSSの普及とOSSライセンス

- | OSSライセンス（以下ラインセンス）の種類は常に増え続けている
 - 現在450以上のライセンスが存在しておりそれぞれが独自の条件を持っている
- | OSSの活用は企業における開発費用を低く抑える手段として非常に有効
 - 複数のサードパーティOSSコンポーネントを統合する際はそれぞれのライセンスを遵守する必要がある
 - 一方で、OSSコンポーネントを商用ソフトウェアに組み込むことは法的問題を招きやすい

先行研究

- | パッケージ管理プラットフォームにおけるOSI非承認ライセンスの使用
 - NPM, RUBYGEMS, CRANを対象に調査
 - OSI非承認ライセンスに限定された研究
- | 非互換ライセンスの使用状況に関するアンケート
 - NPMだけを対象に調査
 - アンケート内容も範囲も狭過ぎて統計的に有意ではない

研究目的

パッケージ管理プラットフォームのライセンス使用状況を
大規模かつ多角的に分析する

データセット

I Maven, NPM, PyPI, RubyGems, Cargo

- 各プラットフォームからメタデータを収集
- 提供されているデータダンプを使用 or APIで収集

Platform	#Package	#Versioned Package
MAVEN	513,939	10,288,841
NPM	2,182,959	29,750,552
PyPI	485,223	4,413,827
RUBYGEMS	188,807	1,427,793
CARGO	103,850	709,755

ライセンス抽出

- 5つのプラットフォームのメタデータにはライセンスフィールドが存在
 - ライセンス情報は必ずしも統一されたフォーマットで記述されるわけではない

一般的な記述方法

```
"name": "be-plugin-antd-theme",
"version": "1.0.0",
"license": "MIT",
"time": "2022-09-15T03:35:20.759Z",
"dependencies": {
  "antd-pro-merge-less": "^3.0.11",
  "rimraf": "^3.0.0",
  "serve-static": "^1.14.1",
  "slash2": "^2.0.0"
}
```

```
"name": "django-modeltranslation",
"version": "0.7.3"
"license": "New BSD",
"upload_time": "2014-01-04 23: 46: 04.877961 UTC",
"classifiers": [
  "Framework :: Django",
  "License :: OSI Approved :: BSD License",
  "Operating System :: OS Independent",
],
"requires": [
  "django(>=1.3)"
]
```

*classifiers*フィールドに記述

不要な文字列も含まれる

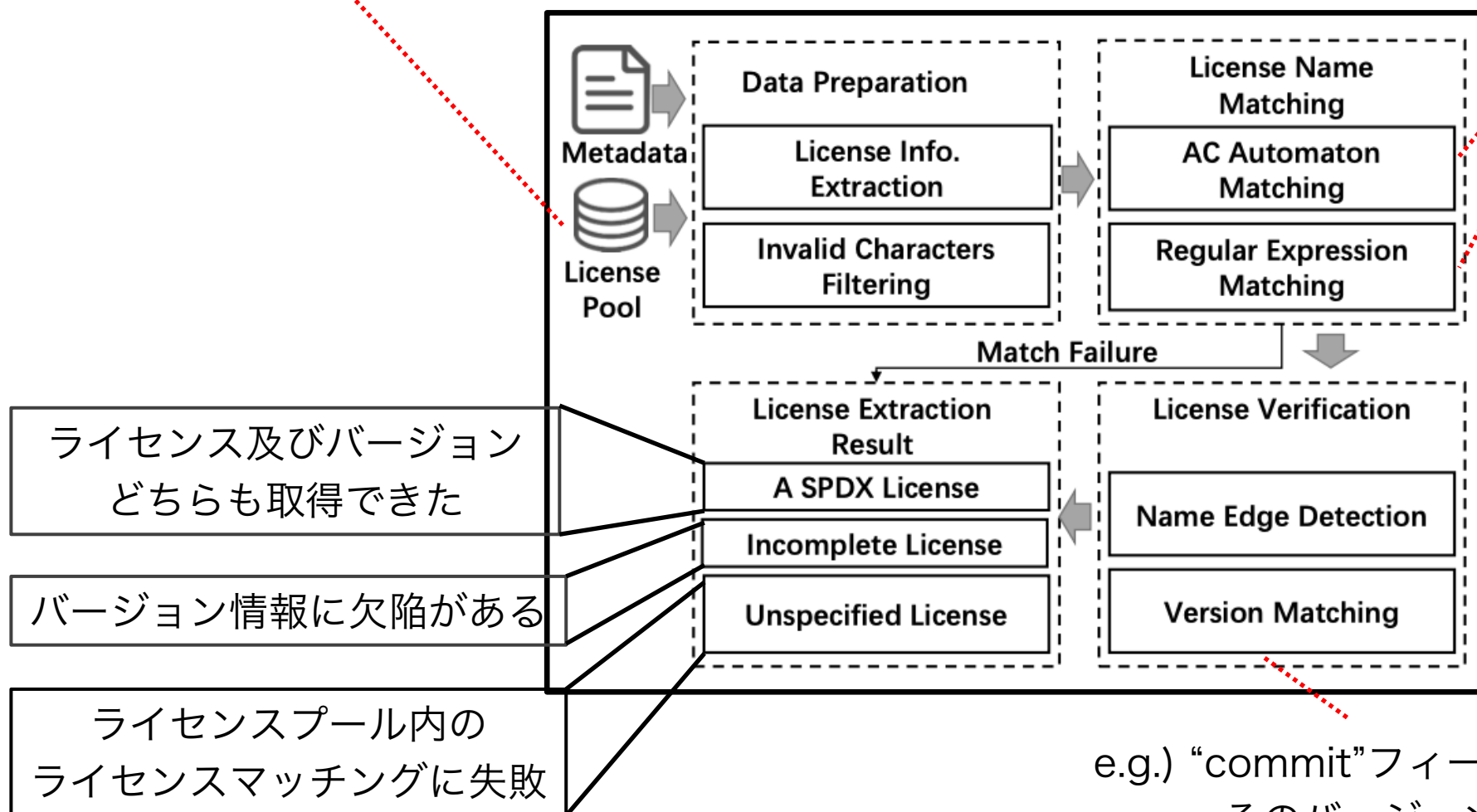
- ライセンス抽出のための既存ツールはメタデータにライセンス情報が不足している場合には使用できない

ライセンス情報を抽出するためのアルゴリズムを提案

ライセンス抽出方法

SPDXライセンスリストに含まれている
469ライセンスの名前と識別子

AC(Aho-Corasick)オートマトンで入力文字列中で
複数のマッチする文字列を同時に取得可能



ライセンス情報を正確に
抽出するために正規表現を使用
e.g.) LGPLv3+は、ACオートマトンは
"+"をマッチできない

ライセンス及びバージョン
どちらも取得できた

バージョン情報に欠陥がある

ライセンスプール内の
ライセンスマッチングに失敗

e.g.) "commit"フィールドから"MIT"を検出した場合、
そのバージョンは後続の文でマッチング

非互換検出方法

| LiDetector

- 学習ベース手法で任意のライセンス文から意味のあるライセンス条項を識別
- 確率的文脈自由文法を適用してソフトウェアの使用条件を推測
- 他のライセンスの使用条件と比較することで非互換を検出

| License-compatibility

- Libraries.ioが提供
- ライセンスの互換性マトリックスを使用して非互換検出

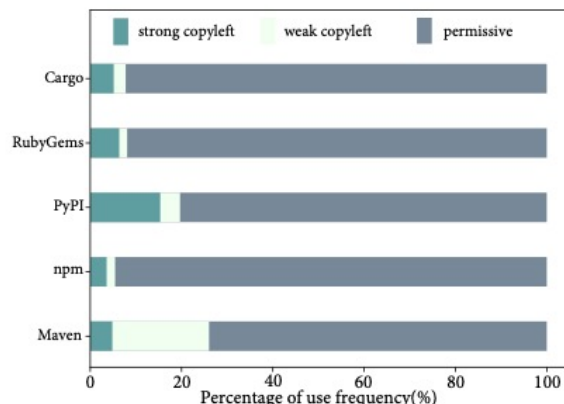
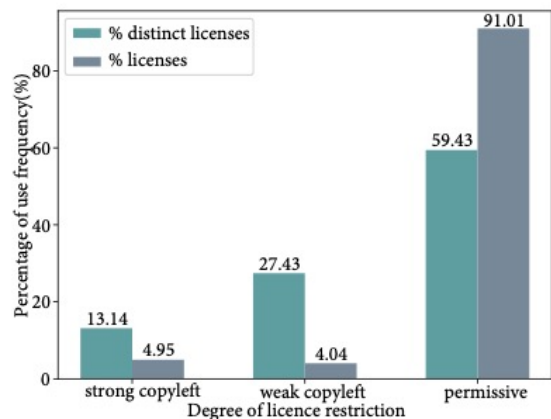
- |RQ1：5つのパッケージ管理プラットフォームにおけるライセンスの使用の現状とそれらの共通点/相違点は何か
- |RQ2：パッケージのバージョンアップでライセンスはどのくらいの頻度で変更されるか
- |RQ3：バージョン管理されたパッケージとその依存関係にあるパッケージの間でライセンス非互換が引き起こす主な原因は何か
- |RQ4：ライセンス使用の時間的な経過による変化はあるか
- |RQ5：全てのパッケージと人気のあるパッケージの間/ライセンスの使用量にどのような違いがあるか

RQ1 | ライセンスの使用状況とその共通点/相違点

	MAVEN	NPM	PyPI	RUBYGEMS	CARGO
%SPDX License	28.71	85.73	71.16	64.34	97.27
%Unspecified License	56.38	13.8	15.41	34.61	2.73
%Incomplete License	14.91	0.47	13.43	1.05	0.0009
%Multi-licenses	1.15	0.81	4.14	0.46	28.39

EPL-2.0とGPL-2.0の組み合わせが64.84%

MITとApache-2.0の組み合わせが96.03%



どのプラットフォームでも寛容なライセンスの使用率が顕著

GPL系は強力なコピーレフト条項に基づいて
一定数存在している

Rank	MAVEN	NPM	PyPI	RUBYGEMS	CARGO
1	Apache-2.0 (59.42%)	MIT (60.27%)	MIT (54.64%)	MIT (70.51%)	MIT (51.22%)
2	MIT (23.78%)	ISC (18.66%)	Apache-2.0 (20.03%)	Apache-2.0 (18.25%)	Apache-2.0 (36.10%)
3	LGPL-2.1-only (5.66%)	Apache-2.0 (13.22%)	GPL-3.0-only (6.14%)	GPL-2.0-only (2.72%)	GPL-3.0-only (2.45%)
4	GPL-3.0-only (2.97%)	BSD-3-Clause (1.41%)	AGPL-3.0-only (3.79%)	GPL-3.0-only (1.78%)	MPL-2.0 (1.69%)
5	BSD-3-Clause (1.33%)	GPL-3.0-only (1.14%)	GPL-3.0-or-later (2.34%)	BSD-3-Clause (1.09%)	BSD-3-Clause (1.20%)

- ライセンスの互換性は法的問題やプロジェクト間の対立を引き起こす
 - パッケージ間の潜在的な依存関係は追跡困難なため直接的な依存関係を調査

Tools	MAVEN	NPM	PyPI	RUBYGEMS	CARGO
LiDetector	7.33%	4.14%	3.29%	7.17%	7.81%
License-Compatibility	6.33%	3.60%	3.04%	6.71%	7.72%

- LiDetectorだけで十分検出可能
 - 辞書ベースのLicense-Compatibilityでは新たなライセンスへの対応は不可能
 - LiDetectorの方が厳しくライセンス非互換を定義

RQ3 | ライセンスの互換性

| tldrlegalに基づいてライセンス条項を分類

- 23の条項を含む（11の権利条項, 12の義務条項）

Terms	MAVEN	NPM	PyPI	RUBYGEMS	CARGO
%Sublicense	82.95	56.21	44.31	84.37	40.56
%Hold Liable	4.16	11.61	15.46	6.85	16.83
%Distribute	3.87	11.19	15.21	5.26	16.29
%Place Warranty	2.91	6.16	9.89	2.13	14.77
%Commercial Use	1.74	4.21	4.54	0.66	4.57

- 全てのプラットフォームでライセンス非互換となる原因はsublicense
- 寛容なライセンスとGPLファミリーライセンスが混在していることが原因

| あるコアパッケージに他のパッケージが依存することが多い

| アプローチ

- パッケージ間の依存関係を重みつき有向グラフを構築
- パッケージの影響力をPageRankアルゴリズムで評価
 - PageRankの値が上位5%に属するパッケージをコアパッケージとみなす

Tools	MAVEN	NPM	PyPI	RUBYGEMS	CARGO
LiDetector	8.44%	9.00%	3.99%	15.83%	0.25%
License-compatibility	7.48%	7.35%	3.48%	10.73%	0.23%

pom.xmlファイルにライセンスフィールドが存在していないため
記述がない場合が多い

	MAVEN	NPM	PyPI	RUBYGEMS	CARGO
%Unspecified License	87.84	10.38	14.21	36.70	0.82
%Incomplete License	6.91	4.40	9.85	0.48	0.01
%Multi-License	0.90	4.74	1.45	0.85	48.42
%Strong Copyleft License	1.66	8.02	11.28	30.82	0.02
%Weak Copyleft License	24.12	4.92	9.10	5.37	0.94
%Permissive license	74.21	87.06	79.62	63.81	99.04

NPMとCargoはプラットフォーム内での
ライセンスの標準化のレベルが高い

RUBYGEEMSを除いて、影響力の大きいパッケージは強力なコピーレフトライセンスの使用を避ける傾向にある

Tools	MAVEN	NPM	PyPI	RUBYGEMS	CARGO
LiDetector	8.44%	9.00%	3.99%	15.83%	0.25%
License-compatibility	7.48%	7.35%	3.48%	10.73%	0.23%

コアパッケージに着目した場合、依存パッケージとの非互換の割合は増加している（RQ3参照）

- | ライセンス抽出方法はなぜ既存ツールを使わなかったのか？
 - Ninkaなどはすでにメンテナンスされていない？
- | Mavenは結果を見る感じデータを取れていないだけなのでこの論文で議論するのは好ましくない
 - Mavenはどこにライセンス情報が記述されてる？
- | このレベルの分析研究が今までされてなかったのがびっくり

補 足 資 料

RQ2 | ライセンスの変更

ライセンス変更は依存関係パッケージと非互換を引き起こす可能性

Platform	#Modification	#Addition	#Deletion	%Change
MAVEN	38,161	14,611	10,091	0.64%
NPM	156,315	45,015	16,151	0.79%
PyPI	44,063	10,616	4,217	1.50%
RUBYGEMS	14,544	10,765	962	2.12%
CARGO	7,688	844	31	1.41%

パッケージのアップデート
でライセンスを変更することは稀

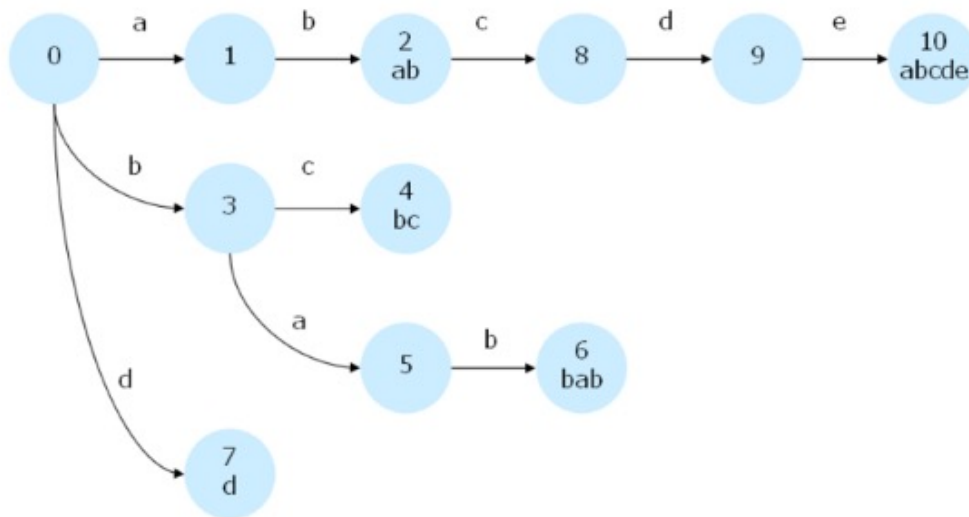
Platform	# S→M	# M→S	# M→M	# S→S
MAVEN	596 (1.7%)	404 (1.2%)	234 (0.7%)	33,934 (96.5%)
NPM	8,388 (5.9%)	8,047 (5.6%)	75 (0.1%)	126,913 (88.5%)
PyPI	3,940 (11.5%)	3,849 (11.2%)	633 (1.8%)	25,978 (75.5%)
RUBYGEMS	287 (2.1%)	61 (0.4%)	53 (0.4%)	13,445 (97.1%)
CARGO	2,249 (37.8%)	784 (13.2%)	104 (1.8%)	2,810 (47.3%)

Apache-2.0とGPL-2.0とEPL-2.0の
間の移行がほとんど

MITとApache-2.0の組み合わせへの変更がほとんど

任意のテキストから辞書に含まれるキーワードを全て抽出する方法

- 辞書中に含まれる各文字をエッジとしたトライ木を構築
- 各ノードには辞書中のキーワードがあれば保持
- キーワード集合{ab, bc, bab, d, abcde}からトライ木を作成すると



これは各ノードが状態，エッジを状態遷移であるオートマトンとみなすことができる

“babcdex”が先ほどのオートマトンに入力された場合

- $0 \rightarrow 3 \rightarrow 5 \rightarrow 6 \rightarrow$ (遷移先なし) と遷移する
 - ここから, “bab”が見つかるが, “ab”, “abcde”なども同時に抽出したい

Aho-Corasick法は遷移先が見つからなかった場合はfailure関数を用いて効率よく遷移先を見つける

- スタート値にリスポーンしたら計算量が多くなる

