The ACM International Conference on the
Foundations of Software Engineering (FSE 2024)

北京大學
PEKING UNIVERSITY

PyRadar: Towards Automatically Retrieving and Validating Source Code Repository Information for PyPI Packages

Kai Gao (高恺)

gaokai19@pku.edu.cn

Weiwei Xu

xuww@stu.pku.edu.cn

Wenhao Yang

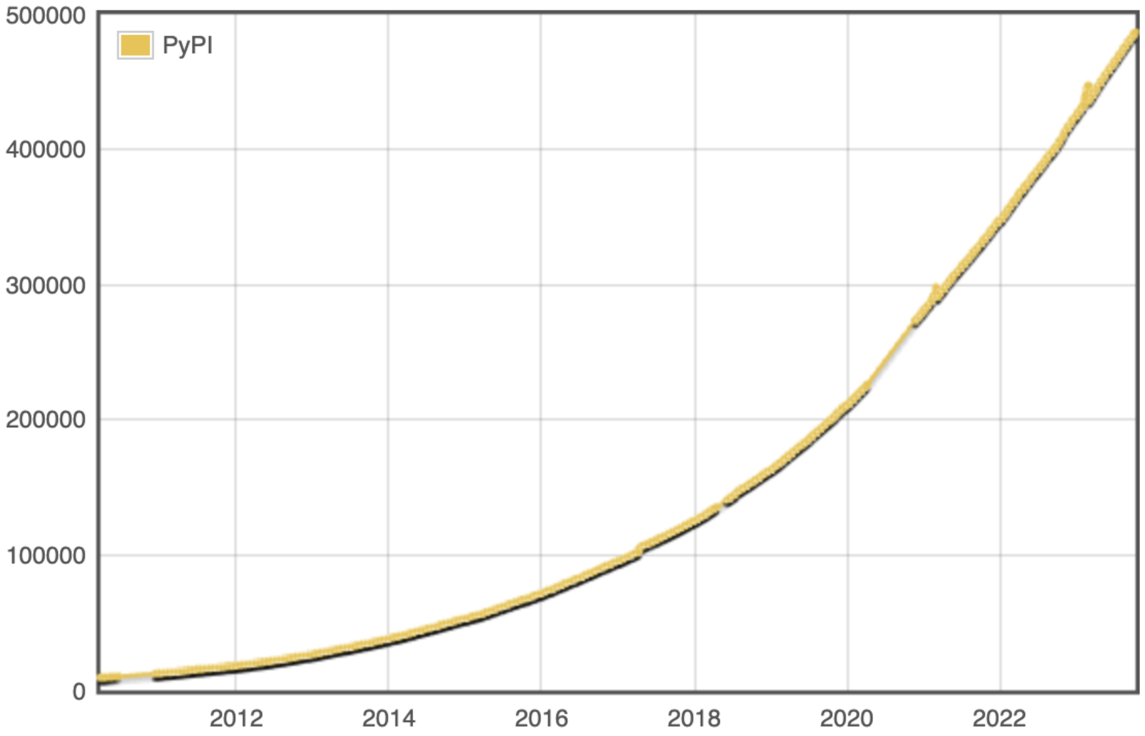
yangwh@stu.pku.edu.cn

Minghui Zhou

zhmh@pku.edu.cn

Porto de Galinhas, July 18, 2024

Rapid Growth and Wide Adoption of PyPI Packages



PyPI Package Count
(http://www.modulecounts.com/)

1.6 Billion

Downloads per day

9.2 Billion

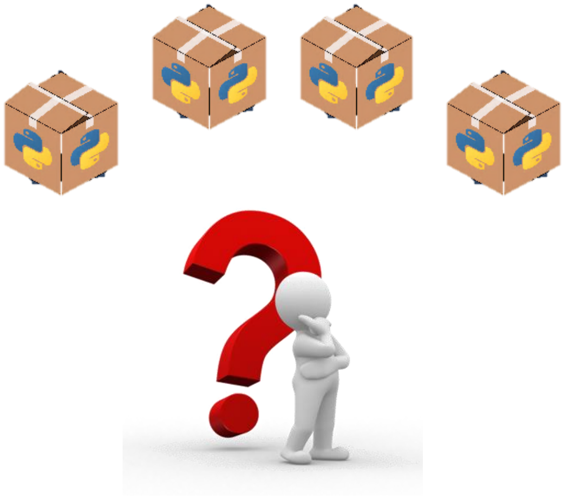
Downloads per week

34.2 Billion

Downloads per month

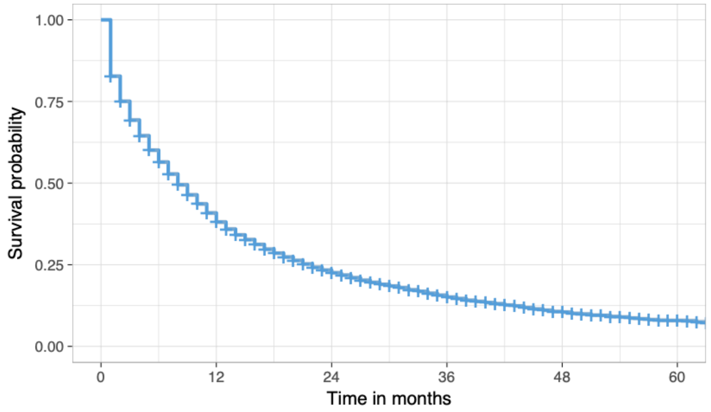
https://pypistats.org/packages/__all__

Critical Problems of Reusing Third-party Packages



Heartbleed

Log4Shell



[Valiev et al., 2018]

Which package to use?

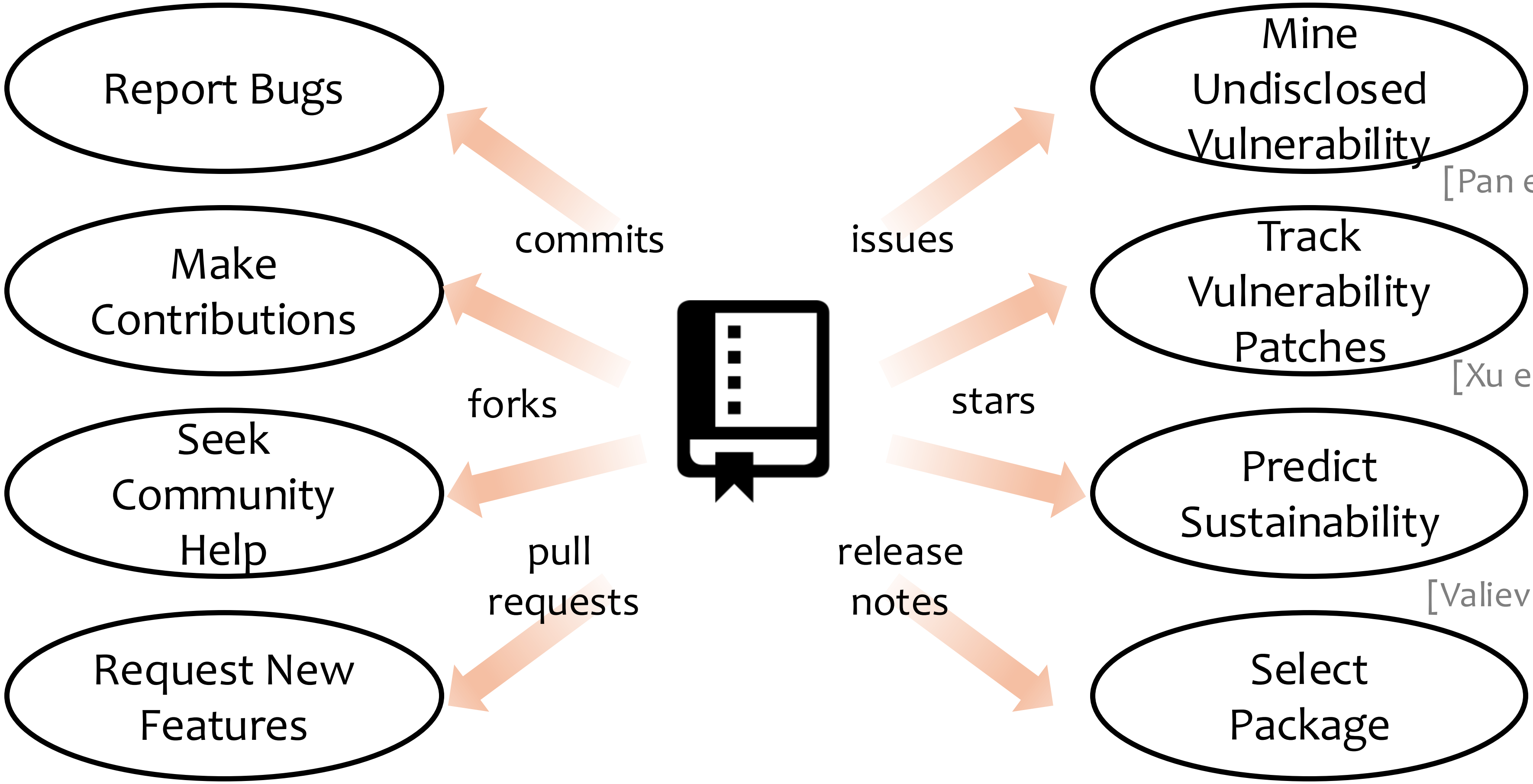
Security Vulnerabilities

Lack of Maintenance

Package Selection

Risk Monitoring

The Package's Source Code Repository Comes to the Rescue



Report Bugs

Make Contributions

Seek Community Help

Request New Features

commits

forks

pull requests

issues

stars

release notes

Mine Undisclosed Vulnerability

[Pan et al., 2022]

Track Vulnerability Patches

[Xu et al., 2022]

Predict Sustainability

[Valiev et al., 2018]

Select Package

[Larios Vargas et al. 2020]

snyk

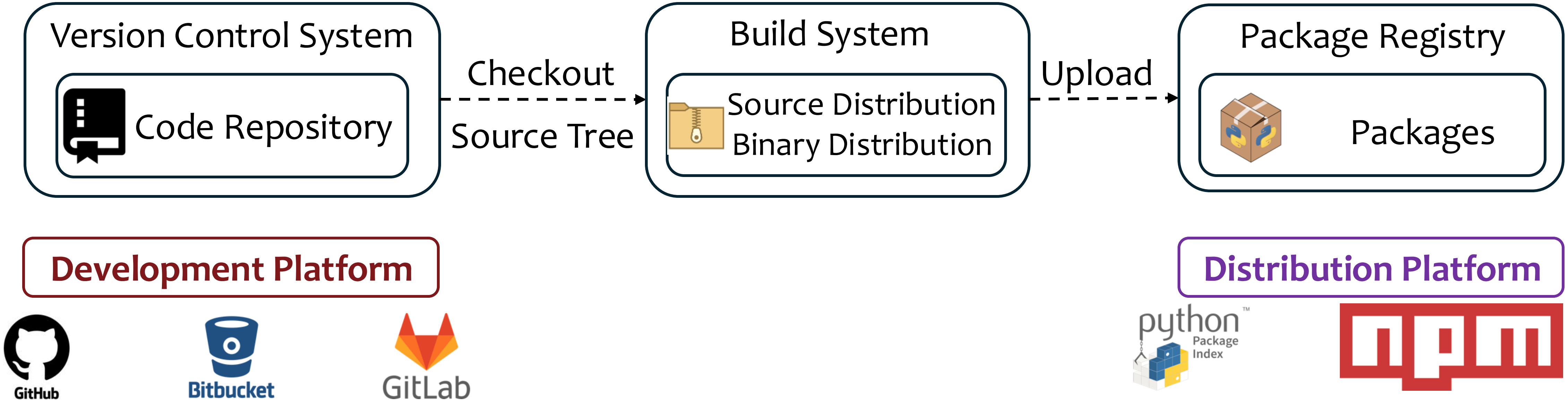
open/source/insights

Libraries.io

usage, risk identification and mitigation

Disconnections between Packages and Their Source Code Repositories

- Most programming languages communities adopt the **development-distribution separation** strategies to manage third-party packages



Version Control System

Code Repository

Checkout

Source Tree

Build System

Source Distribution

Binary Distribution

Upload

Package Registry

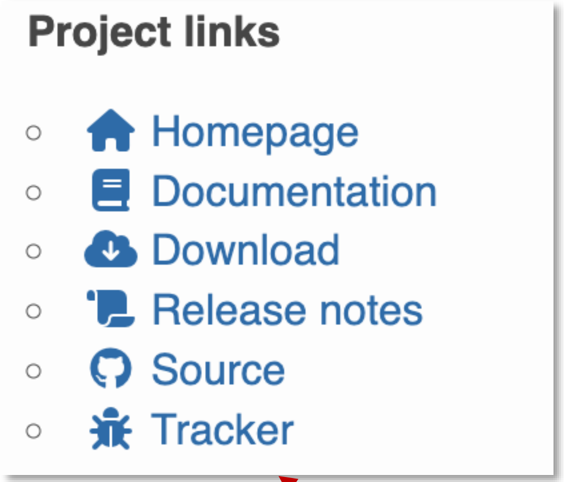
Packages

Development Platform

Distribution Platform

The typical workflow of publishing packages

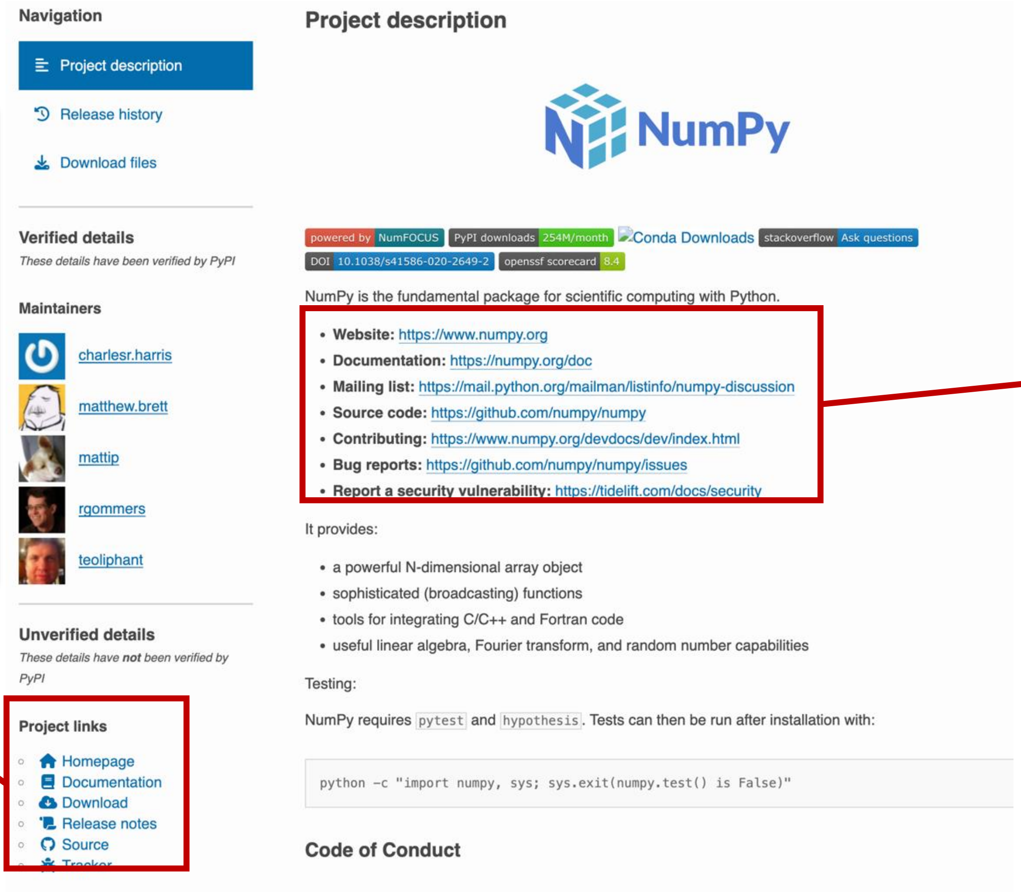
How to Manually Recover Links Between Packages and Their Source Code Repositories



Project links

- ○ Homepage
- ○ Documentation
- ○ Download
- ○ Release notes
- ○ Source
- ○ Tracker

Project Links



Navigation

- Project description
- Release history
- Download files

Verified details

These details have been verified by PyPI

Maintainers

- charlesr.harris
- matthew.brett
- mattip
- rgommers
- teoliphant

Unverified details

*These details have **not** been verified by PyPI*

Project links

- ○ Homepage
- ○ Documentation
- ○ Download
- ○ Release notes
- ○ Source
- ○ Tracker

Project description

powered by NumFOCUS | PyPI downloads 254M/month | Conda Downloads | stackoverflow Ask questions | DOI 10.1038/s41586-020-2649-2 | openssf scorecard 8.4

NumPy is the fundamental package for scientific computing with Python.

- **Website:** https://www.numpy.org
- **Documentation:** https://numpy.org/doc
- **Mailing list:** https://mail.python.org/mailman/listinfo/numpy-discussion
- **Source code:** https://github.com/numpy/numpy
- **Contributing:** https://www.numpy.org/devdocs/dev/index.html
- **Bug reports:** https://github.com/numpy/numpy/issues
- **Report a security vulnerability:** https://tidelift.com/docs/security

It provides:

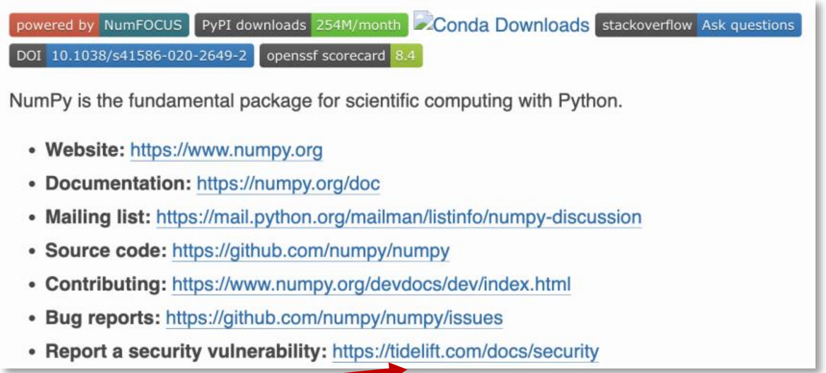
- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Testing:

NumPy requires `pytest` and `hypothesis`. Tests can then be run after installation with:

```
python -c "import numpy, sys; sys.exit(numpy.test() is False)"
```

Code of Conduct



powered by NumFOCUS | PyPI downloads 254M/month | Conda Downloads | stackoverflow Ask questions | DOI 10.1038/s41586-020-2649-2 | openssf scorecard 8.4

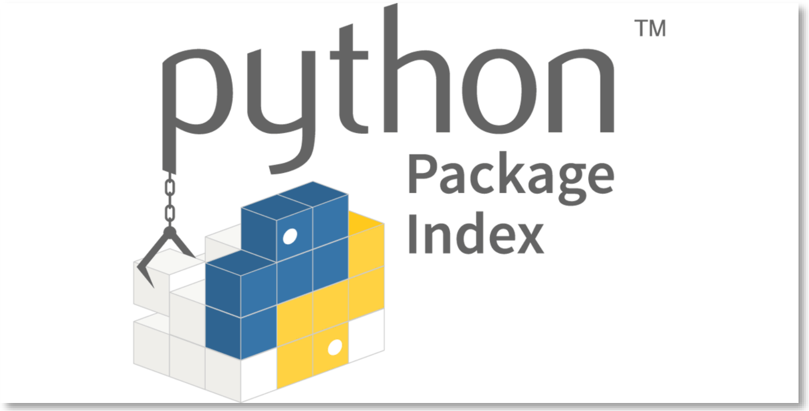
NumPy is the fundamental package for scientific computing with Python.

- **Website:** https://www.numpy.org
- **Documentation:** https://numpy.org/doc
- **Mailing list:** https://mail.python.org/mailman/listinfo/numpy-discussion
- **Source code:** https://github.com/numpy/numpy
- **Contributing:** https://www.numpy.org/devdocs/dev/index.html
- **Bug reports:** https://github.com/numpy/numpy/issues
- **Report a security vulnerability:** https://tidelift.com/docs/security

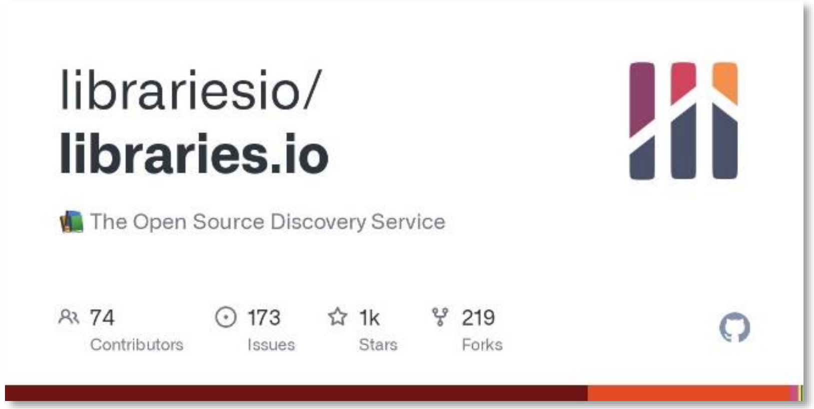
Project description

The PyPI project page of the numpy package, which are generated from the package’s **metadata.**

Existing Automated Tools: Metadata-based



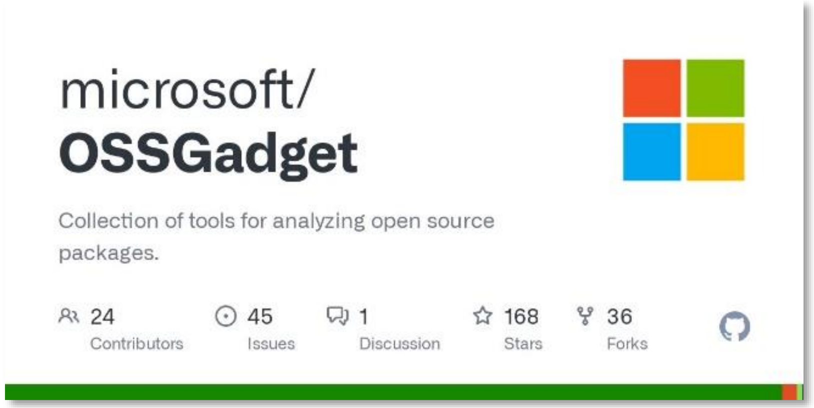
PyPI GitHub Statistics


librariesio/
libraries.io

The Open Source Discovery Service

74 Contributors · 173 Issues · 1k Stars · 219 Forks

Libraries.io


microsoft/
OSSGadget

Collection of tools for analyzing open source packages.

24 Contributors · 45 Issues · 1 Discussion · 168 Stars · 36 Forks

OSS Find Source

simonepirocca/
py2src

Research project about the verification of factors that influence the usage of Python dependencies

3 Contributors · 0 Issues · 4 Stars · 1 Fork

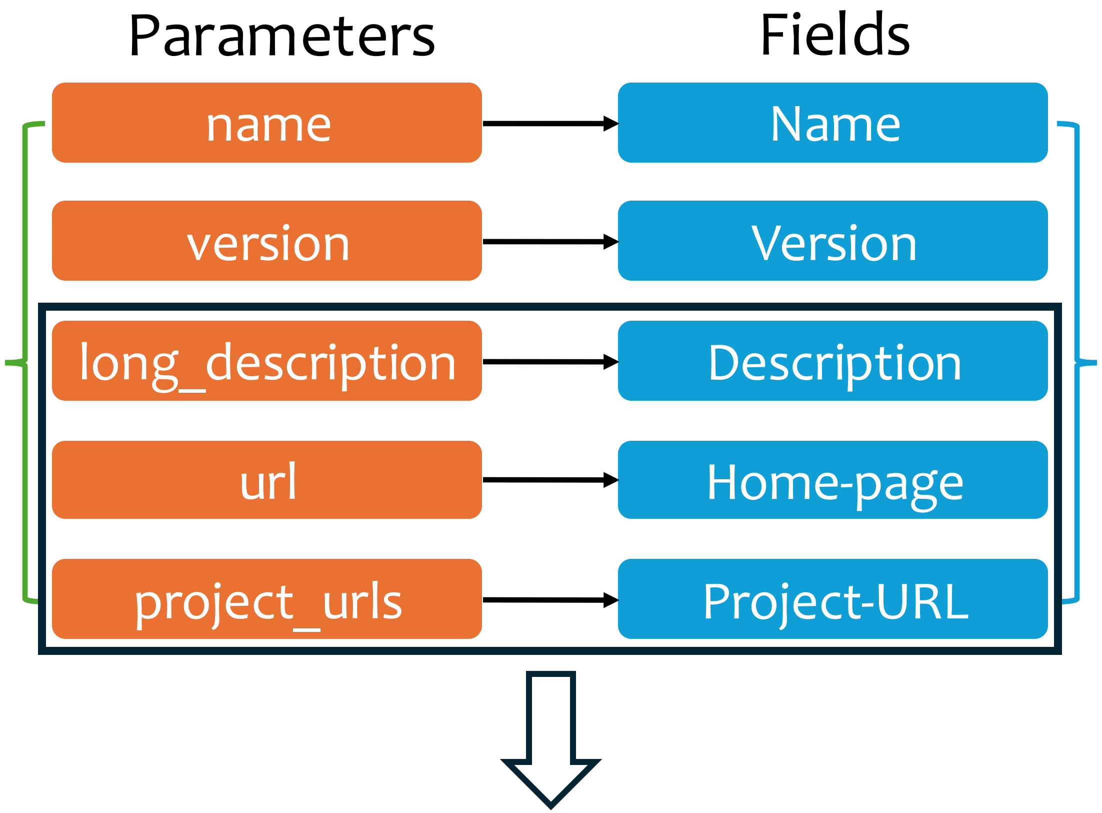
py2src [Vu D L, 2021]

Employ different heuristics to retrieve a source code repository URL from the package's **metadata.**

How are Metadata Generated?

Package Specification Files in Code Repository

- setup.py
- pyproject.toml
- setup.cfg



Parameters → Fields

name → Name

version → Version

long_description → Description

url → Home-page

project_urls → Project-URL

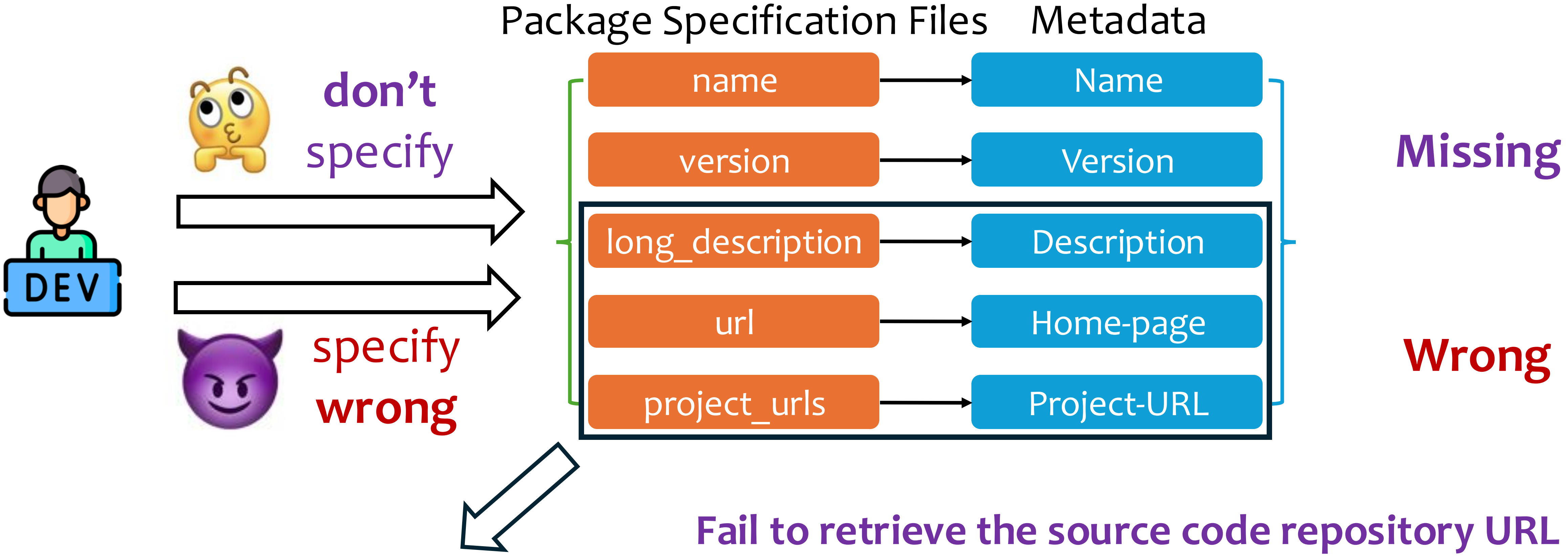
Metadata of PyPI Package

Manually specified by package developers

Used by metadata-based approaches

Automatically generated by build tools

Limitations of Metadata-based Approaches



don't specify

specify **wrong**

Package Specification Files	Metadata
name	Name
version	Version
long_description	Description
url	Home-page
project_urls	Project-URL

Missing

Wrong

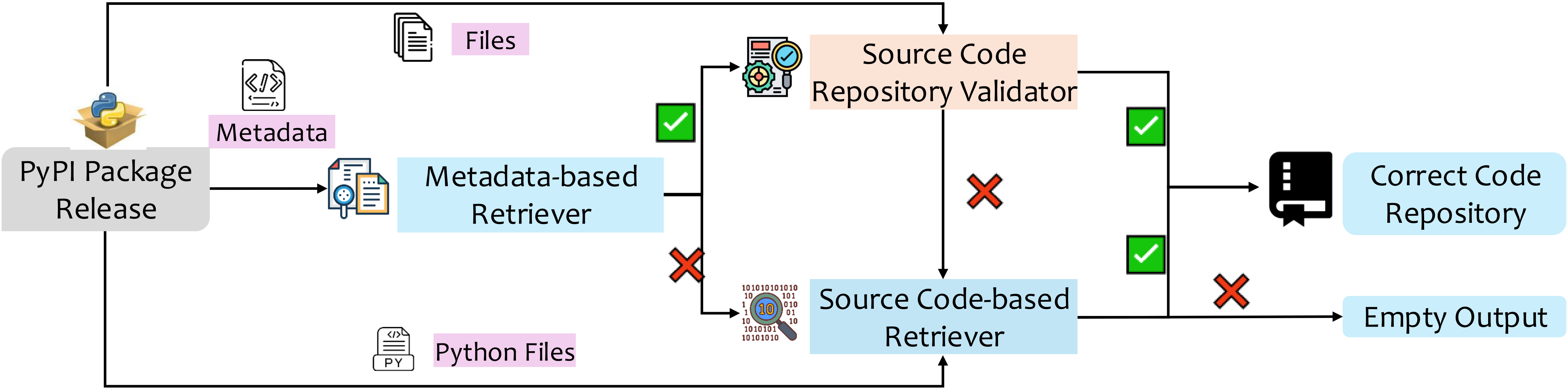
Used by metadata-based approaches

Fail to retrieve the source code repository URL

Retrieve the incorrect source code repository URL

Our Solution to Address the Two Limitations—— PyRadar

Intuitions: PyPI packages do not just have metadata, they also have **source code** in their distributions!



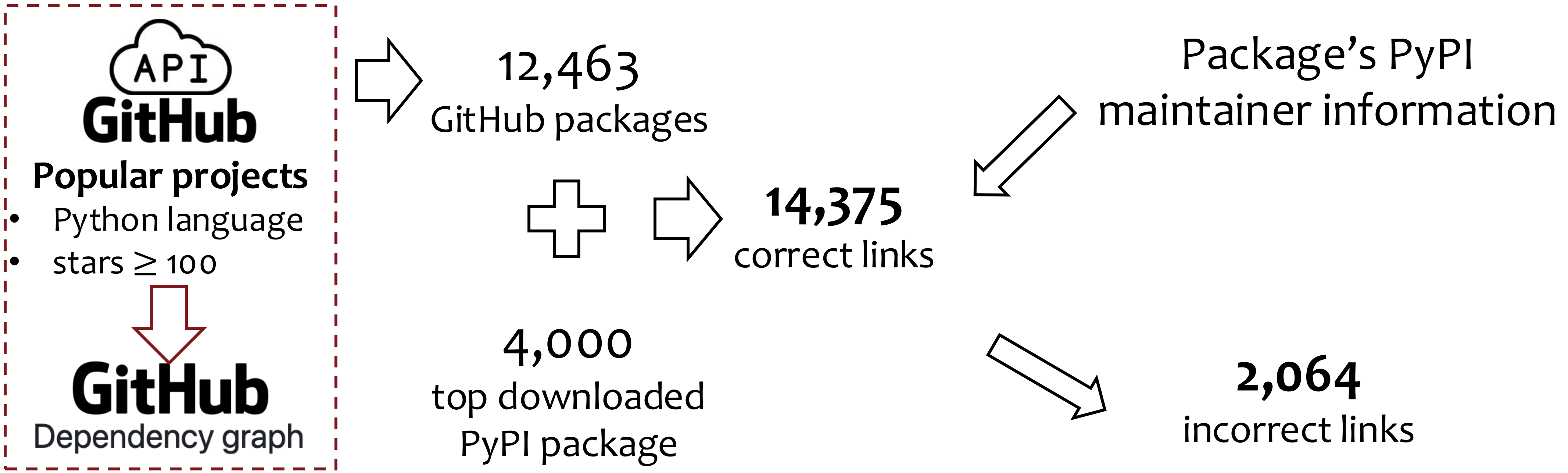
Overview of PyRadar

How to Collect the Correct and Incorrect Package-Repository Links?

A heuristic approach: collect correct links first, then incorrect links

Assumption 1: the linkage between **popular packages** and **popular source code repositories** should be correct.

Assumption 2: Packages published by the same source code repository should have **the same PyPI maintainers.**



API

GitHub

Popular projects

- Python language
- stars ≥ 100

GitHub

Dependency graph

12,463

GitHub packages

4,000

top downloaded PyPI package

14,375

correct links

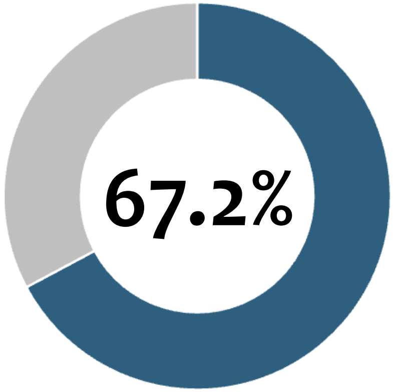
Package's PyPI maintainer information

2,064

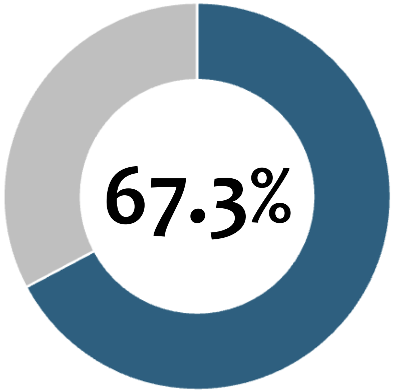
incorrect links

The Metadata-based Retriever: Evaluation of existing methods

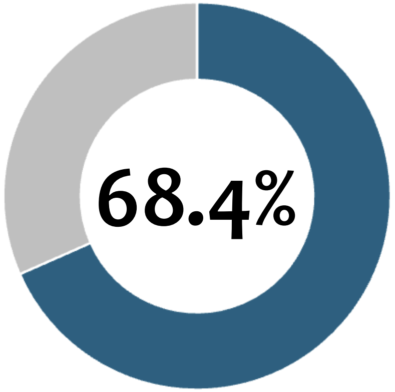
Collect metadata for **4,227,425** releases of 423,726 packages



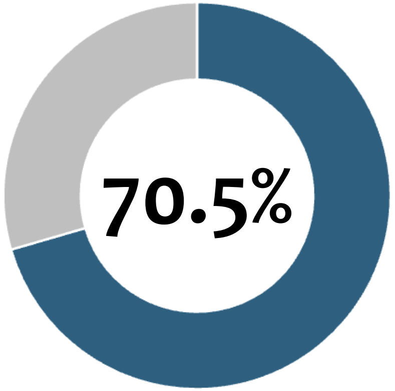
PyPI GitHub Statistics



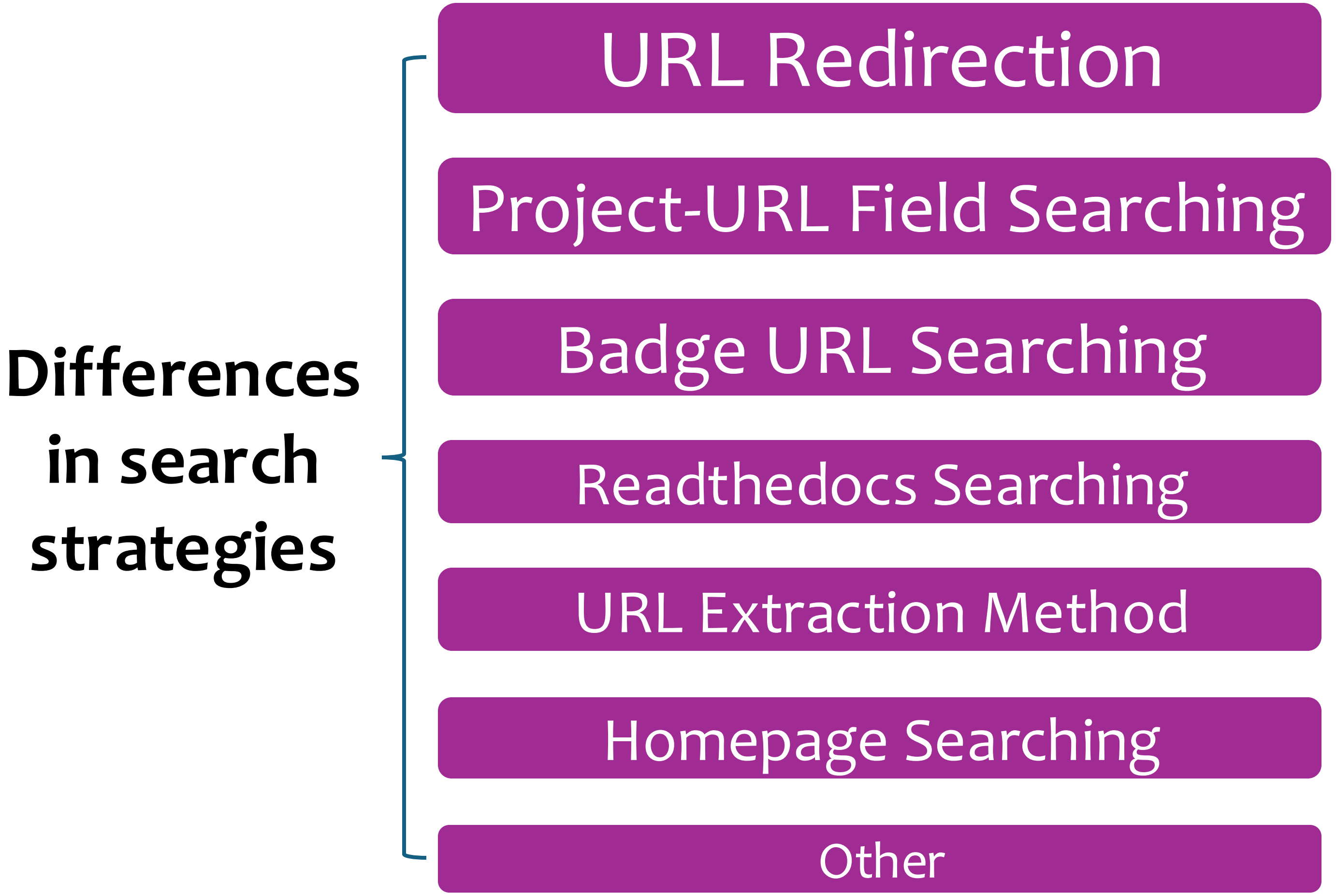
OSS Find Source



Libraries.io



py2src

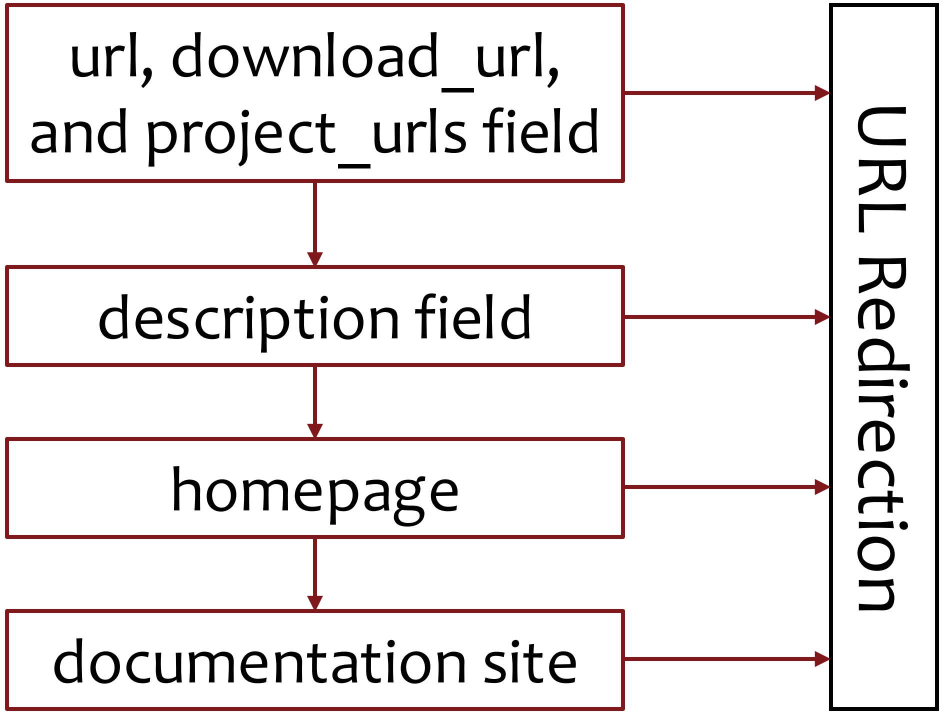


Differences in search strategies

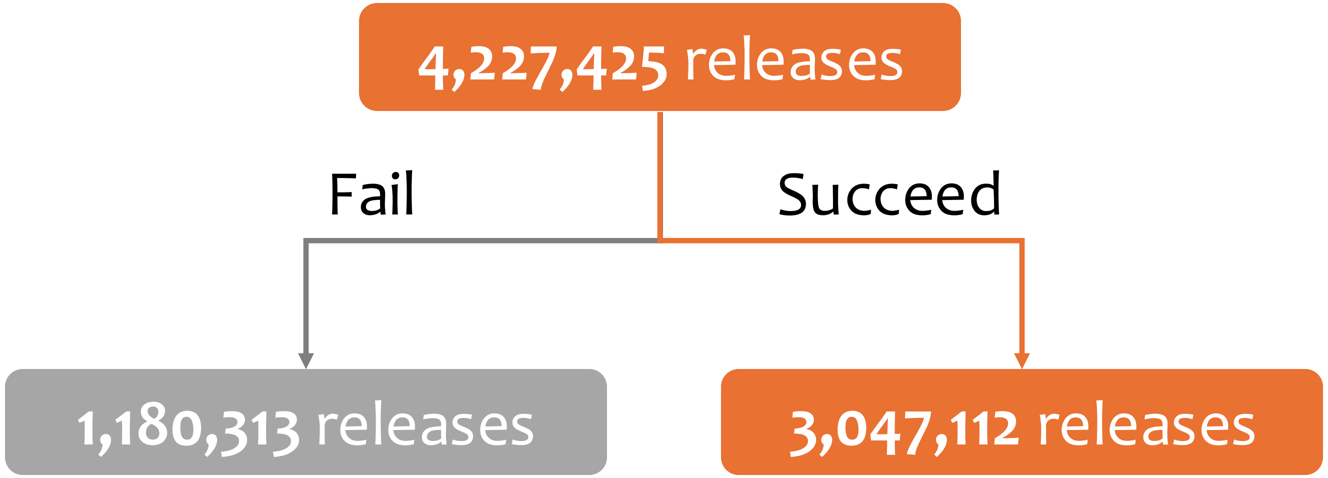
- URL Redirection
- Project-URL Field Searching
- Badge URL Searching
- Readthedocs Searching
- URL Extraction Method
- Homepage Searching
- Other

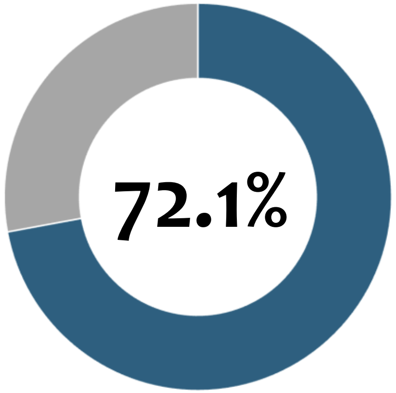
The Metadata-based Retriever: Design & Evaluation

Design



Evaluation



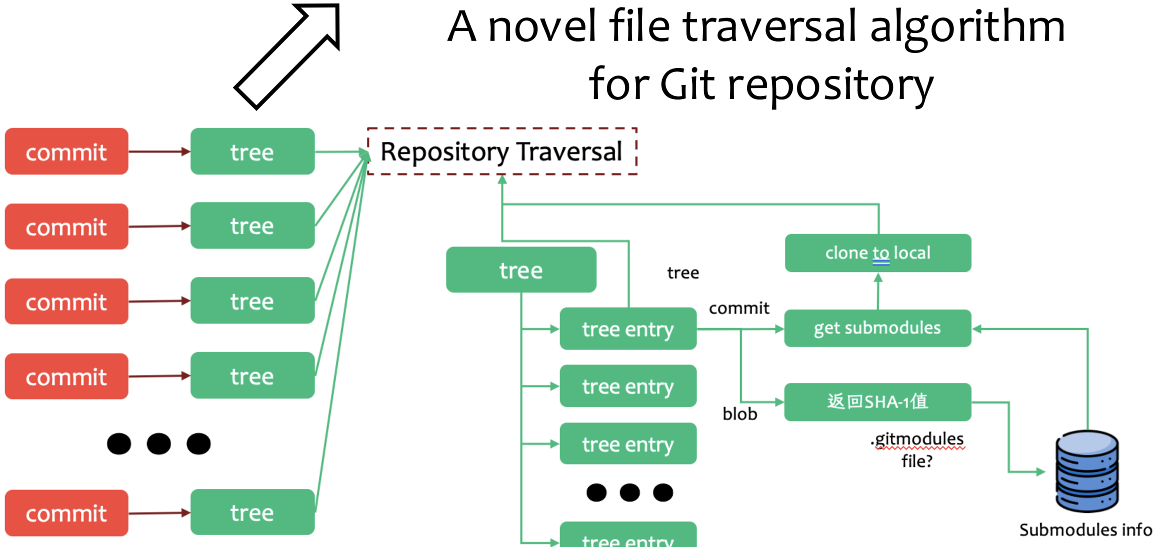


1.6% higher than py2src

The Source Code Repository Validator: Phantom File Analysis

Phantom files[1]: files appearing in the release’s distribution but not in the package’s source code repository

= hashes of files in the release’s distribution − hashes of files in the package’s source code repository



A novel file traversal algorithm for Git repository

Key Findings:

1. Incorrect links have **more phantom (Python) files** than correct links.

2. The package specification file **setup.py or pyproject.toml** is more likely to be a phantom file in the incorrect links.

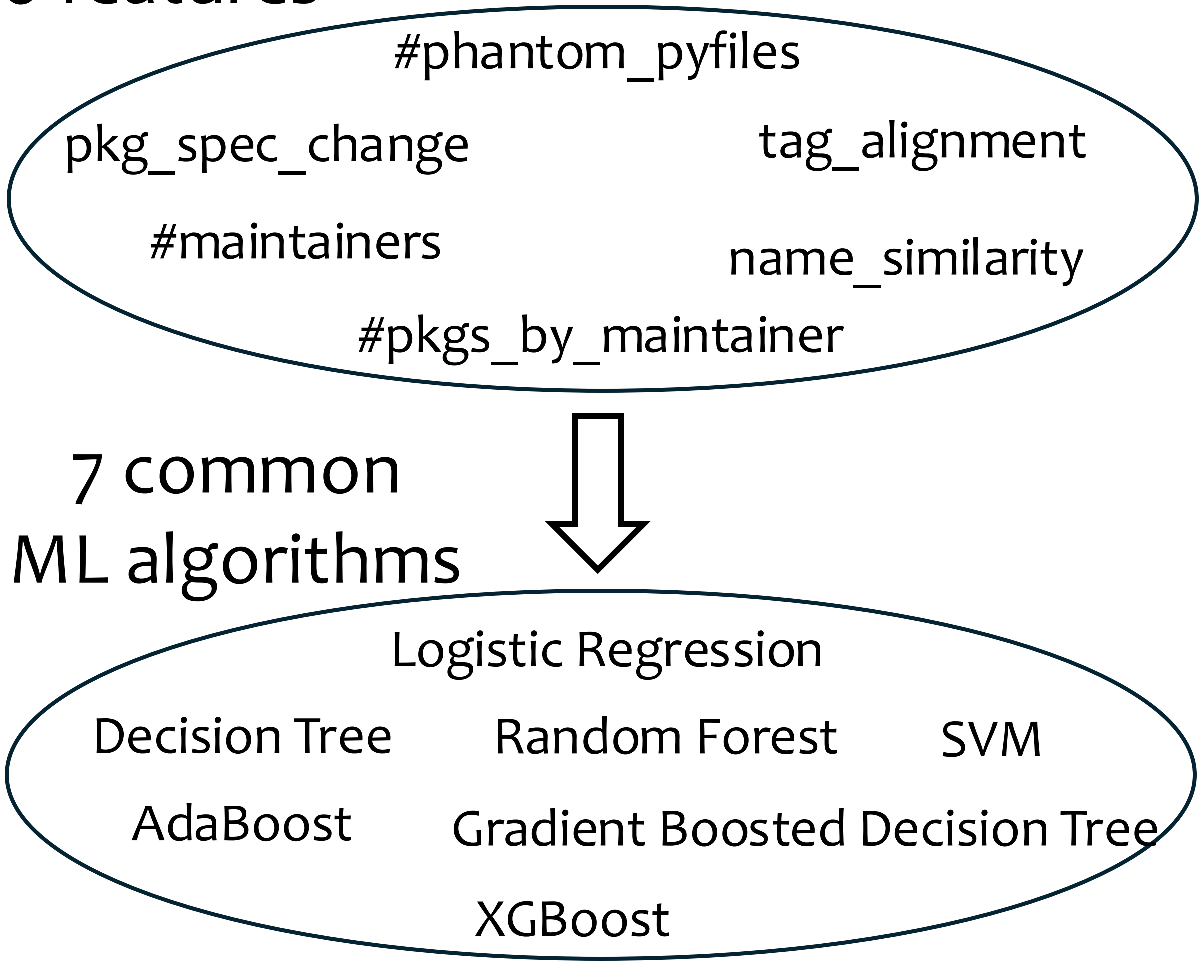
3. **Python files** typically remain the same in the correct links.

[1] Vu D L, Massacci F, Pashchenko I, et al. Lastpymile: identifying the discrepancy between sources and packages. ESEC/FSE 2021

The Source Code Repository Validator: Design & Evaluation

Design

6 features



#phantom_pyfiles

pkg_spec_change

tag_alignment

#maintainers

name_similarity

#pkgs_by_maintainer

7 common ML algorithms

Logistic Regression

Decision Tree

Random Forest

SVM

AdaBoost

Gradient Boosted Decision Tree

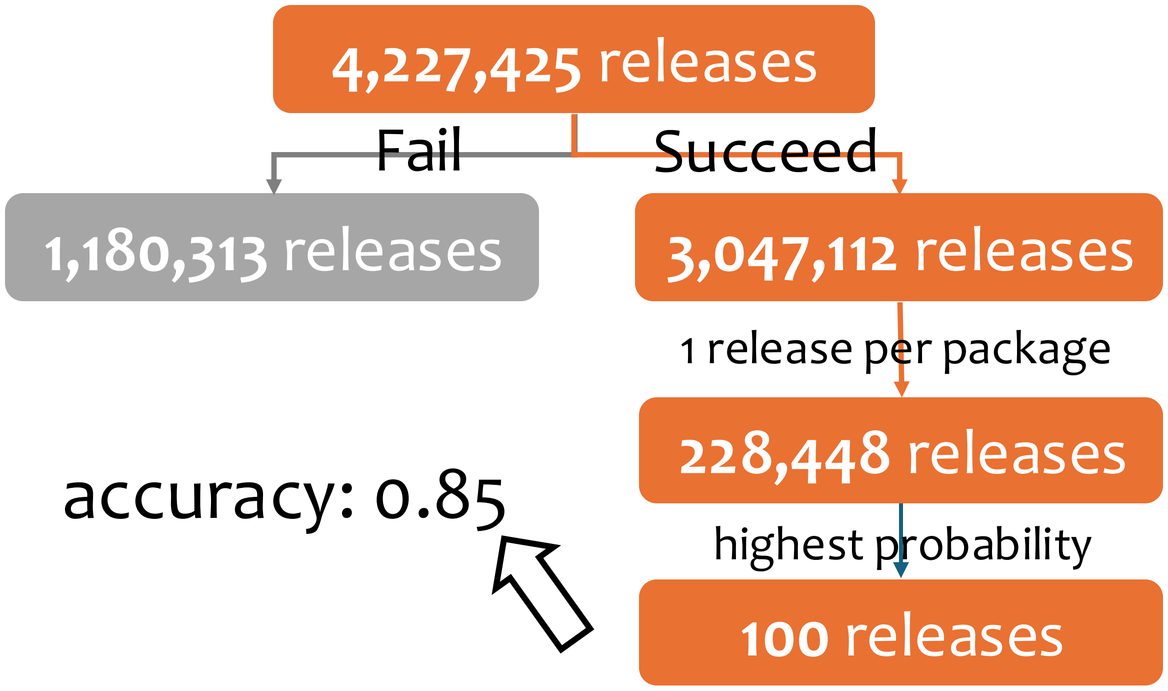
XGBoost

Evaluation

14,375 correct links

2,064 incorrect links

AUC: 0.995



4,227,425 releases

Fail

Succeed

1,180,313 releases

3,047,112 releases

1 release per package

228,448 releases

highest probability

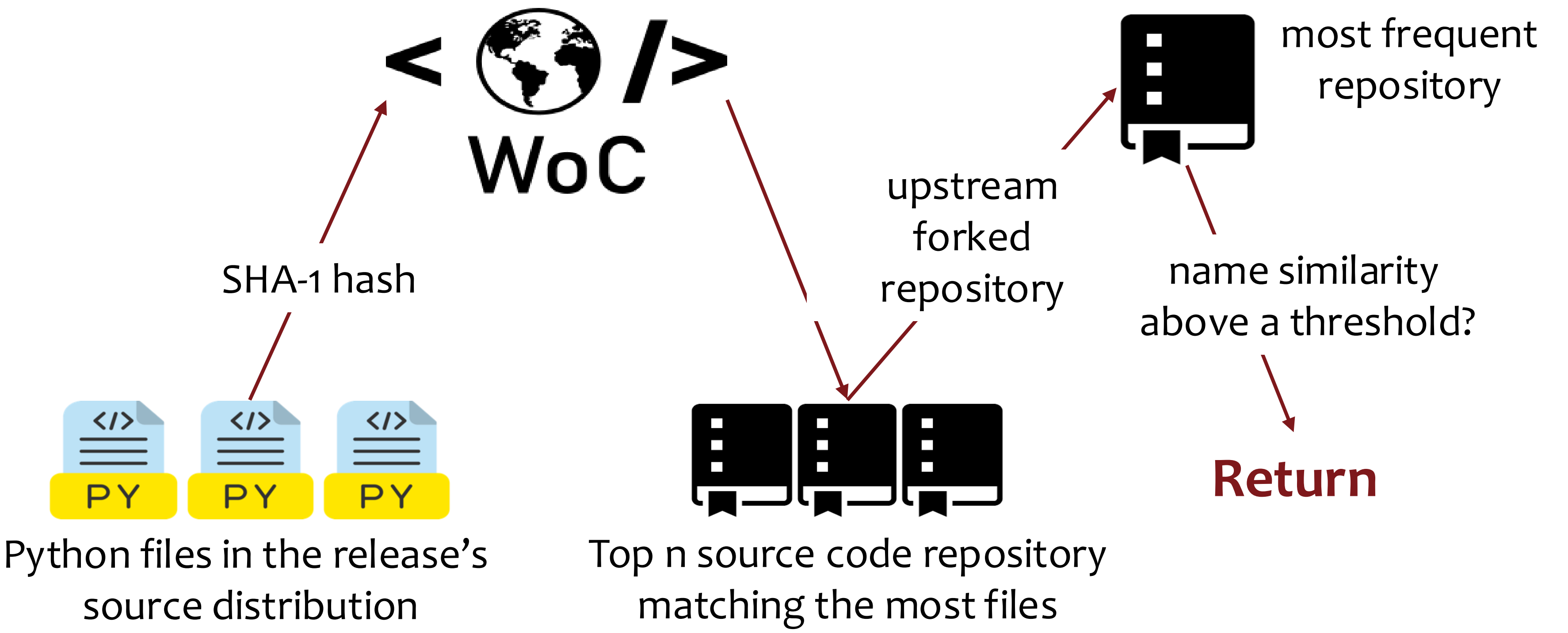
100 releases

accuracy: 0.85

The Source Code-based Retriever: Design

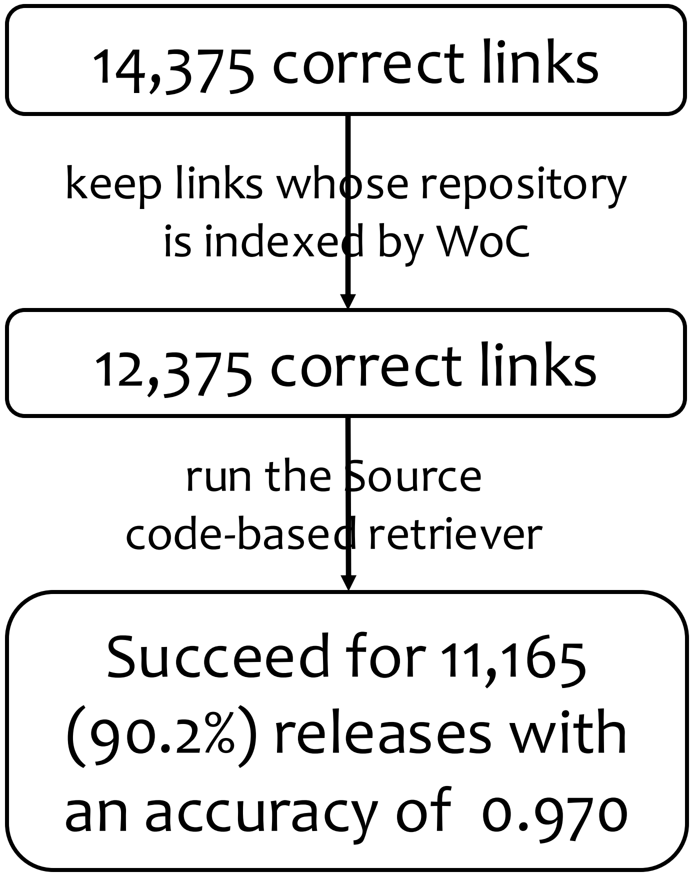
Key evidence: Python files are bridge between the package and its source code repository based on the results of phantom file analysis

Design: A **hash matching** and **name similarity**-based heuristic retrieval algorithm



WoC
SHA-1 hash
PY
PY
PY
Python files in the release's source distribution
Top n source code repository matching the most files
upstream forked repository
most frequent repository
name similarity above a threshold?
Return

The Source Code-based Retriever: Evaluation



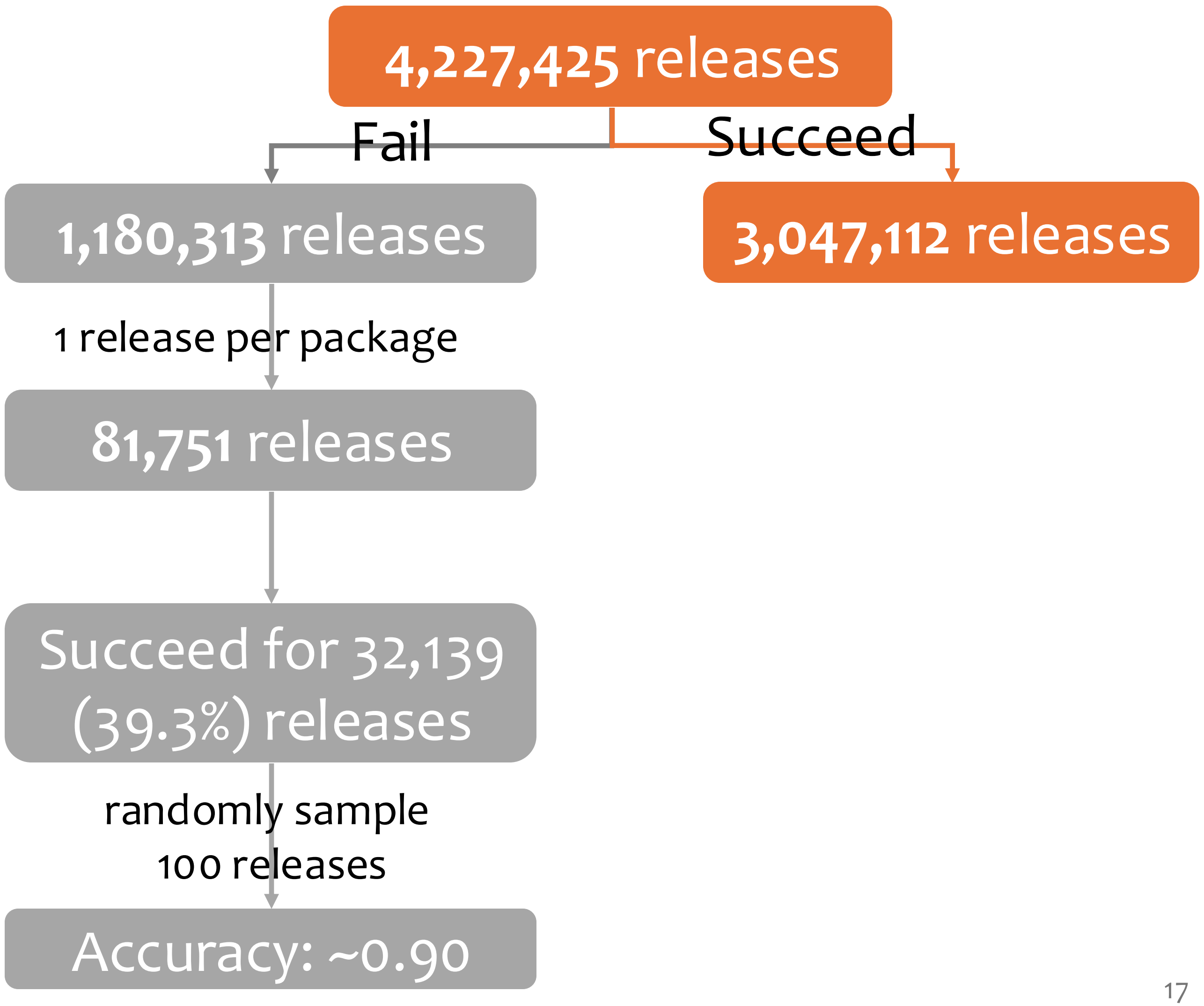
14,375 correct links

keep links whose repository is indexed by WoC

12,375 correct links

run the Source code-based retriever

Succeed for 11,165 (90.2%) releases with an accuracy of 0.970



4,227,425 releases

Fail

1,180,313 releases

1 release per package

81,751 releases

Succeed for 32,139 (39.3%) releases

randomly sample 100 releases

Accuracy: ~0.90

Succeed

3,047,112 releases

PyRadar: Overall Evaluation

14,375
correct links

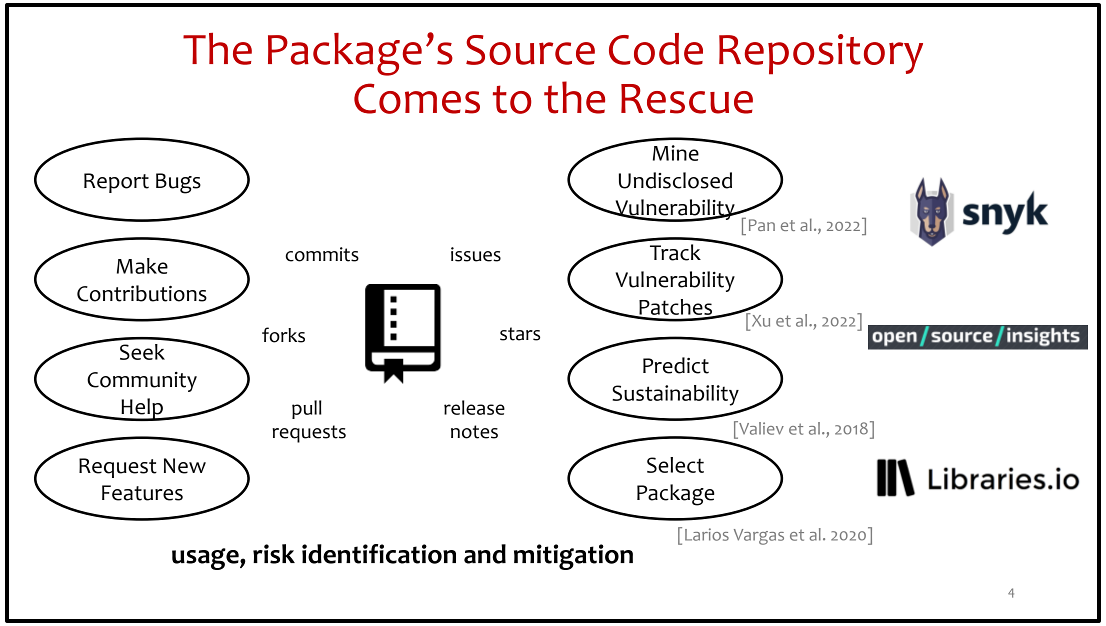
2,064
incorrect links

Accuracy: 0.88

Discussion

- Future Improvement
  - **Cross-link accounts** between code hosting platforms and package registries.
    - **Mechanisms:** account binding and account mutual authentication
    - **Automatic methods.**
  - Finer-grained **code analysis** to identify normal and abnormal changes in the build process.
  - Package registries (e.g., PyPI) and package management tools (e.g., pip) should **integrate validation mechanisms** to notify users if a package's repository information is problematic when searching or installing package.
- What about **other PL communities** that adopts the development-distribution separation strategies?
  - NPM, Maven, etc
  - Go

Summary



The Package's Source Code Repository Comes to the Rescue

Report Bugs

Make Contributions

Seek Community Help

Request New Features

commits

issues

forks

stars

pull requests

release notes

Mine Undisclosed Vulnerability

[Pan et al., 2022]

Track Vulnerability Patches

[Xu et al., 2022]

Predict Sustainability

[Valiev et al., 2018]

Select Package

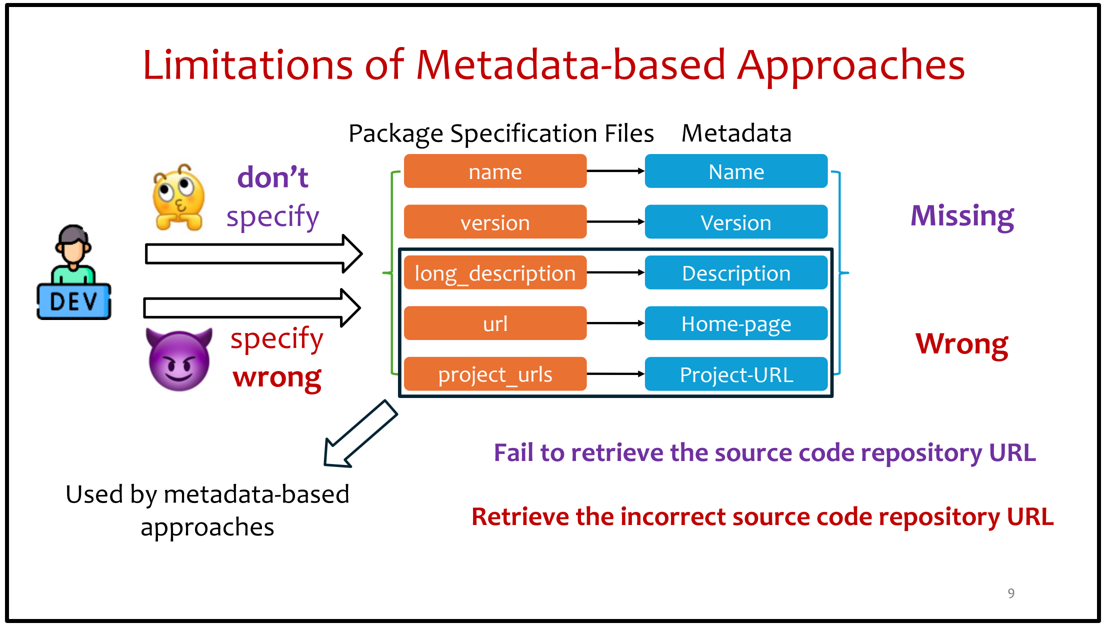
[Larios Vargas et al. 2020]

snyk

open/source/insights

Libraries.io

usage, risk identification and mitigation



Limitations of Metadata-based Approaches

DEV

don't specify

specify wrong

Package Specification Files

Metadata

Package Specification Files	Metadata
name	Name
version	Version
long_description	Description
url	Home-page
project_urls	Project-URL

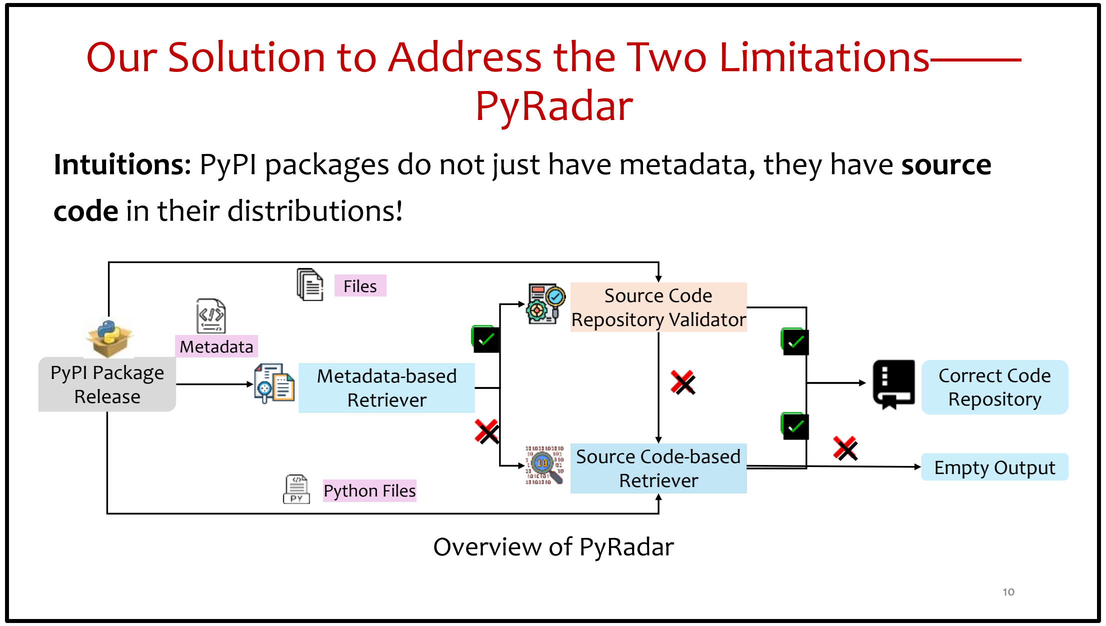
Missing

Wrong

Used by metadata-based approaches

Fail to retrieve the source code repository URL

Retrieve the incorrect source code repository URL



Our Solution to Address the Two Limitations——PyRadar

Intuitions: PyPI packages do not just have metadata, they have **source code** in their distributions!

PyPI Package Release

Files

Metadata

Python Files

Metadata-based Retriever

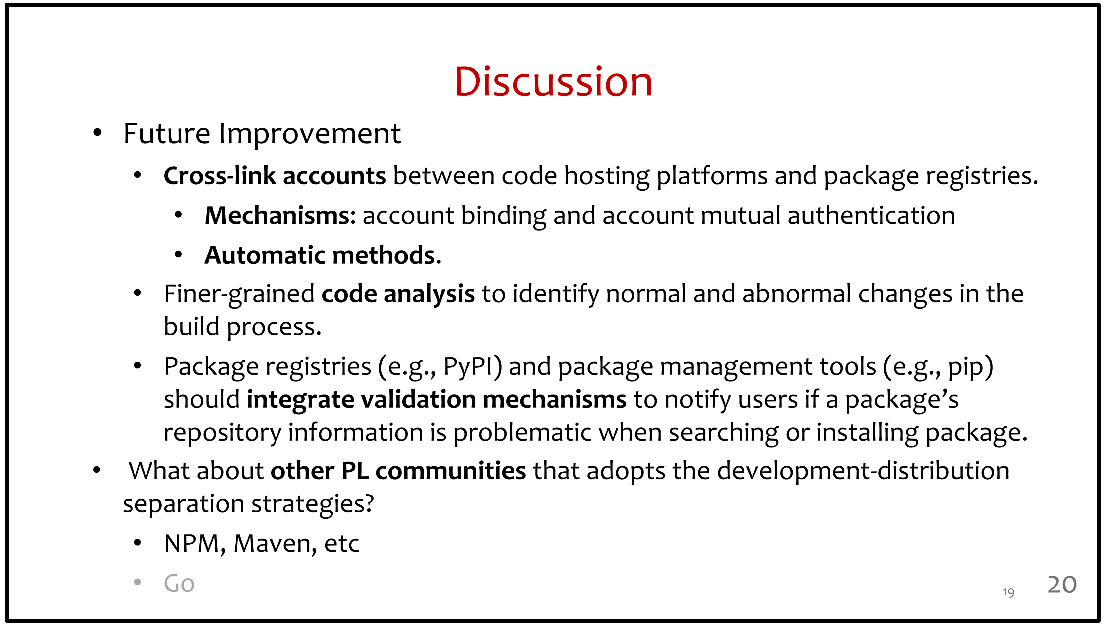
Source Code Repository Validator

Source Code-based Retriever

Correct Code Repository

Empty Output

Overview of PyRadar



Discussion

- Future Improvement
  - **Cross-link accounts** between code hosting platforms and package registries.
    - **Mechanisms:** account binding and account mutual authentication
    - **Automatic methods.**
  - Finer-grained **code analysis** to identify normal and abnormal changes in the build process.
  - Package registries (e.g., PyPI) and package management tools (e.g., pip) should **integrate validation mechanisms** to notify users if a package's repository information is problematic when searching or installing package.
- What about **other PL communities** that adopts the development-distribution separation strategies?
  - NPM, Maven, etc
  - Go

The ACM International Conference on the Foundations of Software Engineering (FSE 2024)

北京大學
PEKING UNIVERSITY

Thank You!

Paper

Code

Kai Gao (高恺)

gaokai19@pku.edu.cn

Weiwei Xu

xuww@stu.pku.edu.cn

Wenhao Yang

yangwh@stu.pku.edu.cn

Minghui Zhou

zhmh@pku.edu.cn

Acknowledgement: National Natural Science Foundation of China 61825201 and 62332001.