SoftwareX 10 (2019) 100263

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

pybliometrics: Scriptable bibliometrics using a Python interface to Scopus

Michael E. Rose [a,*], John R. Kitchin [b]

[a] Max Planck Institute for Innovation and Competition, Germany
[b] Carnegie Mellon University, Department of Chemical Engineering, United States

ARTICLE INFO

Article history:
Received 19 March 2019
Received in revised form 19 June 2019
Accepted 19 June 2019

Keywords:
Scopus
Software
Python
Bibliometrics
Scientometrics

ABSTRACT

We present a wrapper for the Scopus RESTful API written for Python 3. The wrapper allows users to access the Scopus database via user-friendly interfaces and can be used without prior knowledge of RESTful APIs. The package provides classes to interact with different Scopus APIs to retrieve information as diverse as citation counts, author information or document abstracts. Files are cached to speed up subsequent analysis. The package addresses all users of Scopus data, such as researchers working in Science of Science or evaluators. It facilitates reproducibility of research projects and enhances data integrity for researchers using Scopus data.

Code metadata

Current code version	2.1
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX_2019_55
Legal Code License	MIT License
Code versioning system used	git
Software code languages, tools, and services used	python
Compilation requirements, operating environments & dependencies	requests, pbr
If available Link to developer documentation/manual	https://pybliometrics.readthedocs.io/en/stable/
Support email for questions	Michael.Rose@ip.mpg.de

1. Motivation and significance

Scopus is one of the citations, bibliometrics and abstracts databases that have become standard in the field of scientometrics and bibliometrics [1–4]. A search on Google Scholar in January 2019 for "Scopus" returned about 1,990,000 results, about 24,200 of which date from 2018 alone. For example, a number of recent studies in the field of Economics of Science exemplify the importance of Scopus [5–11].

To ease the use of the Scopus data, we develop `pybliometrics`, a Python package to access the RESTful APIs that Scopus provides. The alternative to our solution is either to download information from Scopus manually, or to use one of the few solutions in languages less prominent than Python. The latter includes solutions in PHP and bash [12], DSpace [13] or JavaScript on Apache [14]. The functionality in Python however puts `pybliometrics` in the hands of scientists who already use Python (e.g. for data analysis, visualization, etc.). The manual download on the other hand has its own limitations. For example, only 2000 records can be downloaded from a search. Manual work is also error prone while an automated process avoids idiosyncratic errors, is more efficient, allows reproducibility and acts as "authoritative definition" of the final dataset [15].

`pybliometrics` comes with a consistent, simple interface, does not require a server, and can easily be integrated with everything in the data science ecosystem of Python, including visualization tools and Machine Learning.

* Corresponding author.
E-mail address: Michael.Rose@ip.mpg.de (M.E. Rose).

https://doi.org/10.1016/j.softx.2019.100263

Table 1
Overview of classes in pybliometrics.

API	Type	Rest.	Class in pybliometrics	Purpose
Affiliation Search	Search		AffiliationSearch	Search for affiliation entities
Author Search	Search		AuthorSearch	Search for authors
Scopus Search	Search		ScopusSearch	Search for documents
Abstract Retrieval	Retrieval		AbstractRetrieval	Retrieve information on documents
Affiliation Retrieval	Retrieval		ContentAffiliation-Retrieval	Retrieve information on affiliations
Author Retrieval	Retrieval		AuthorRetrieval	Retrieve information on authors
Citations Count Metadata	Metadata	x		Retrieve document citation counts
Citations Overview	Metadata	x	CitationsOverview	Retrieve document citation counts by year
PlumX Metrics	Metadata			PlumX metrics (social media mentions) for documents
Serial Title Metadata	Metadata	x	SerialTitle	Retrieve information on serial titles
Subject Classifications	Metadata			Search for and retrieve Subject Classifications
Author Feedback	Other	x		Provide author feedback (i.e. corrections)

Notes: Table lists currently (January 2019) available RESTful APIs of the Scopus database. "Rest." indicates that the subscriber either needs additional permission from Elsevier for this API (i.e. the used key needs more privileges) or all views of the API are restricted.

2. Software description

`pybliometrics` builds on the requests package [16] and is available from the Python Project Inventory (PyPI). It can be used cross-platform inside a Python interpreter or in script mode.

2.1. Software architecture

`pybliometrics` implements eight RESTful APIs of the Scopus database (Table 1). These cover the three search APIs (Abstract Search, Author Search, Affiliation Search), the three retrieval APIs (Abstract Retrieval, Affiliation Retrieval, Author Retrieval) and two out of five metadata APIs (Citation Overview API, Serial Title API).

Each class is designed in a consistent way. The required parameter is a search query string (for search classes) or the identifier for Scopus entities (for retrieval and metadata classes) such as authors, documents or affiliations. Only the Citation Overview class requires an additional parameter, namely the start year. Query and retrieval results are cached for subsequent analysis.

2.2. pybliometrics.scopus exceptions

`pybliometrics` defines a number of exceptions depending on the type of error received. Though they all inherit from exceptions provided by the requests package, they allow error-specific handling.

2.3. Configuration

A configuration file stores two important pieces of information: the authentication credentials and the paths to cache files. Upon first usage, and if the configuration file does not exist, `pybliometrics` guides through the process of creating it. The file is stored in a hidden folder named ".scopus" in the user's home directory (`~/`).

2.4. Software functionality

In general, information the user is interested in is stored in properties. Most properties are of object type string or list of namedtuples (e.g. in case multiple results with multiple fields of information are returned). The names of properties are aligned to the field names in the Scopus documentation.

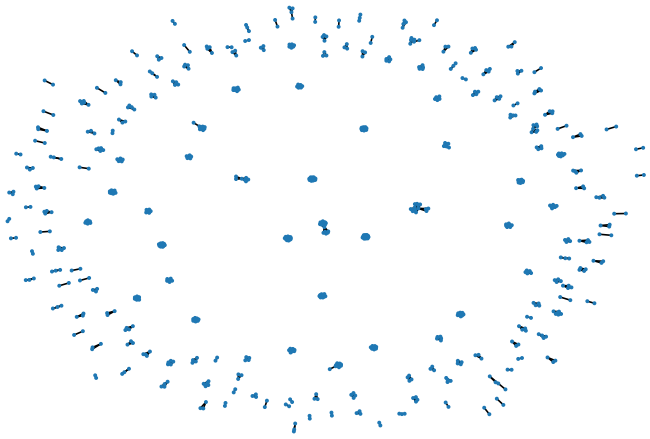

Fig. 1. Collaboration network for the SoftwareX journal. *Notes:* Figure shows a co-author network using publications in the journal SoftwareX until 2019. Each blue dot represents an author. Two authors are connected by an undirected and unweighted black link. Graph layout according to an adopted Fruchtermann–Reingold algorithm using NetworkX.

3. Illustrative examples

3.1. Automated download of information

Often one is only interested in getting a list of publications and the information Scopus provides on them. The following code provides the time necessary to download all publications in the journal Science in 2010.

The total number of results (i.e. the number of publications) equals 2260, which is already more then a manual download allows. Depending on download speed, the time to download the information takes less than 5 min. However, since `pybliometrics` caches the information, reusing it takes less than second only.

3.2. Creating a collaboration network

A common task in bibliometric analysis is to create a social network of co-authors. The following example uses NetworkX [17] (Version 2.3) to create the co-author network for the SoftwareX journal:

The resulting graph is depicted in Fig. 1. The network consists of 734 unique authors that appear on 228 papers. The nodes in the graph are connected through 1636 edges. Because we use author profile IDs provided by Scopus, we do not need to disambiguate author names.

Source Code 1: Example code to time the download of retrieving an article list.

```
from datetime import datetime

import pandas as pd
from pybliometrics.scopus import ScopusSearch

# Download
start = datetime.now().replace(microsecond=0)
s = ScopusSearch("ISSN(0036-8075) AND PUBYEAR IS 2010")
end = datetime.now().replace(microsecond=0)
print(end-start)
# 0:04:29
print(len(s.results))  # Number of papers
# 2260

# Reusing
start = datetime.now().replace(microsecond=0)
s = ScopusSearch("ISSN(0036-8075) AND PUBYEAR IS 2010")
end = datetime.now().replace(microsecond=0)
print(end-start)
# 0:00:00
```

Source Code 2: Example code to produce a co-author network using scopus.

```
from itertools import combinations

import networkx as nx  # Version 2.3
import matplotlib.pyplot as plt  # Version 3.0.3
from pybliometrics.scopus import ScopusSearch

# Obtain authors
s = ScopusSearch('SOURCE-ID(21100422153)')
print(len(s.results))  # Number of papers
# 228
authors = [i.author_ids.split(';') for i in s.results]
# Create list of pairwise combinations
combs = [list(combinations(i, 2)) for i in authors]
edges = [i for j in combs for i in j]
# Create network
G = nx.Graph()
G.add_edges_from(edges)
print(nx.info(G))
# Name:
# Type: Graph
# Number of nodes: 734
# Number of edges: 1636
# Average degree:   4.4578
# Draw and save network
nx.draw(G, node_size=2)
plt.savefig('network.pdf', bbox_inches='tight', figsize=(50, 50))
```

3.3. Analyzing citation distributions

Citations to papers are the foundation of many evaluation exercises. The class `CitationOverview()` provides access to the (restricted) Citation Overview API. The following exercise gathers yearly citation counts for three publications until early 2018 cited in the Nobel decision of Economist Richard H. Thaler. We use pandas [18] (Version 0.24.2) and seaborn [19] (Version 0.9.0). Additionally, we perform a two-sided Kolmogorov–Smirnov test for the similarity of citation distributions using scipy [20] (Version 1.2.1).

The resulting citation distribution for the three papers is shown in Fig. 2. With a p-value $< 2.9 \times 10^{-16}$, the test rejects the null hypothesis that the citation distributions for the first and the second paper are the same.

Source Code 3: Example code to analyze citation distributions using scopus.

```
import matplotlib.pyplot as plt  # Version 3.0.3
import pandas as pd  # Version 0.24.2
import scipy  # Version 1.2.1
import seaborn as sns  # Version 0.9.0
from pybliometrics.scopus import AbstractRetrieval, CitationOverview

# Save yearly citation counts in DataFrame
data = pd.DataFrame()
for eid in ("2-s2.0-84900013243", "2-s2.0-84977703147", "2-s2.0-84977736029"):
    ab = AbstractRetrieval(eid)
    year = ab.coverDate[:4]  # Year from format YYYY-MM-DD
    co = CitationOverview(eid, start=year)
    data = data.append(pd.Series(dict(co.cc), name=ab.title))
# Prepare data for plotting
data = data.T.astype(float)
data.index.name = "year"
data = data.reset_index()
# Plot and save
fig, ax = plt.subplots(figsize=(15, 7))
melted = data.melt(id_vars="year", var_name="paper", value_name="citations")
sns.barplot(data=melted, x="year", y="citations", hue="paper", ax=ax)
plt.savefig("citations.pdf", bbox_inches="tight")
# KL-similarity of distributions
scipy.stats.ks_2samp(data.iloc[:,0], data.iloc[:,1])
# Ks_2sampResult(statistic=1.0, pvalue=2.922351375337816e-16)
```

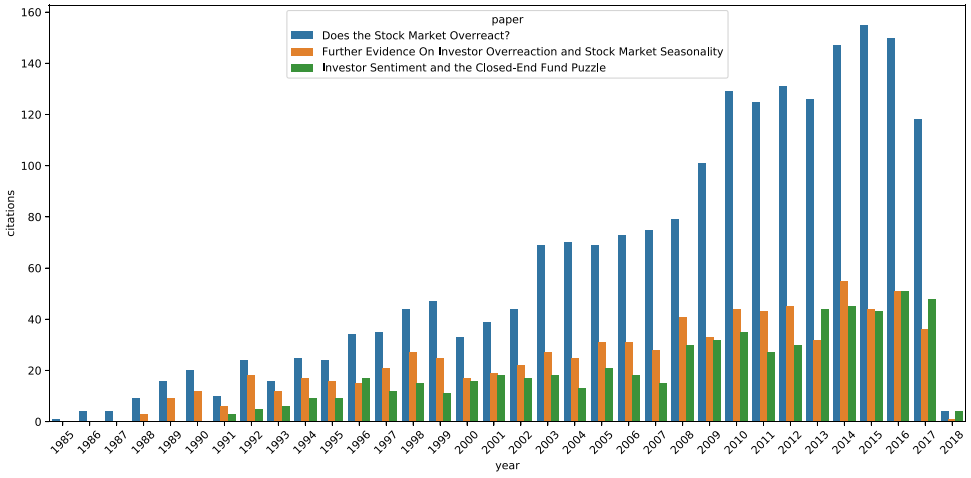

Fig. 2. Citation distribution for three selected papers. *Notes:* Figure shows the yearly citation count of three selected papers until early 2018.

3.4. Topic modeling using abstracts

A final exercise connects to the emerging field of machine learning and natural language processing. Researchers interested in understanding research topics can use `pybliometrics`, for instance, to obtain large amounts of scientific abstracts quickly and analyze them with specialized software. In source code 4 we find topics among a corpus of scientific articles using Latent Dirichlet Allocation. We use nltk [21] (Version 3.4.3) and sklearn [22] (Version 0.20.2). We use abstracts of 226 papers published in SoftwareX until June 2019 and assume 3 topics.

The first topic covers technical terms used to describe the presented software, the second topic includes terms that present the software while the third topic seems to be about terms authors use to describe what the software can be used for.

4. Impact

`pybliometrics` has several features that make it optimal for scientists interested in using Scopus data. It is not necessary to understand RESTful APIs or to know how to parse XML or json. The `pybliometrics` code is open-source. The only requirement is to have some understanding of Python. Due to its availability on PyPI, `pybliometrics` is widely available.

Using `pybliometrics`, scientists working in the field of Science of Science [23] (alternative names include Economics of Science [24] and Sociology of Science) can increase both the reproducibility and exactness of their projects. Reproducibility increases because users obtain data in the same way from the same source and it becomes transparent where the data originates from

Source Code 4: Example code for topic modelling with LDA where scopus provides abstracts of published papers.

```
from string import digits, punctuation

import nltk  # Version 3.4.3
import pandas as pd  # Version 0.24.2
from pybliometrics.scopus import ScopusSearch
from sklearn.feature_extraction.text import CountVectorizer as CV
from sklearn.decomposition import LatentDirichletAllocation as LDA

# Obtain abstracts
s = ScopusSearch('SOURCE-ID(21100422153)')  # Retrieve and cache query results
abstracts = [i.description for i in s.results if i.description]
print(len(abstracts))  # Number of abstracts
# 226
# Stem words
stemmer = nltk.snowball.SnowballStemmer('english')
def tokenize_and_stem(text):
    text = text.translate(str.maketrans({p: "" for p in punctuation + digits + "©"}))
    return [stemmer.stem(t) for t in nltk.word_tokenize(text.lower())]
# Vectorize abstracts
params = {"stop_words": "english", "tokenizer": tokenize_and_stem, "ngram_range": (1, 2),
          "max_df": 0.6}
vectorizer = CV(**params)
matrix = vectorizer.fit_transform(abstracts)
terms = vectorizer.get_feature_names()
print(len(terms))  # Number of terms
# 17536
# Perform LDA
lda = LDA(n_components=3).fit(matrix)
# Print ten most important terms per topic
for topic in lda.components_:
    text = ", ".join(terms[i] for i in topic.argsort()[:-10:-1])
    print(text)
# model, provid, function, simul, calcul, comput, librari, implement, user
# data, softwar, model, tool, develop, applic, method, user, provid
# softwar, data, develop, model, provid, analysi, simul, process, code
```

and how it was obtained [25].[1] Exactness increases because it becomes very easy to integrate updated data into the analysis.

5. Conclusions

The capability of `pybliometrics` to help researchers in bibliometric studies is based on the consistent, simple interface automating all the data retrieval, json serialization, caching and error handling. It allows researchers to speed up their analyses in reproducible ways and science evaluators to automate the retrieval of evaluation data.

Acknowledgments

We thank the various contributors listed at https://github.com/pybliometrics-dev/pybliometrics/graphs/contributors. For help preparing this paper we thank the editor, two anonymous referees, Stefano Baruffaldi, Felix Pöge and Nurzhan Sapargali.

[1] It should be noted though that using data from a database where additions, deletions and corrections are frequent, limit reproducibility per se.

Declaration of competing interest

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

References

[1] Falagas ME, Pitsouni EI, Malietzis GA, Pappas G. Comparison of PubMed, Scopus, Web of Science, and Google Scholar: strengths and weaknesses. FASEB J 2007;22(2):338–42. http://dx.doi.org/10.1007/978-94-007-7618-0_310.

[2] Harzing A-W, Alakangas S. Google Scholar, Scopus and the Web of Science: a longitudinal and cross-disciplinary comparison. Scientometrics 2016;106(2):787–804. http://dx.doi.org/10.1007/s11192-015-1798-9.

[3] Mongeon P, Paul-Hus A. The journal coverage of Web of Science and Scopus: a comparative analysis. Scientometrics 2016;106(1):213–28. http://dx.doi.org/10.1007/s11192-015-1765-5.

[4] Zeng A, Shen Z, Zhou J, Wu J, Fan Y, Wang Y, Stanley HE. The science of science: From the perspective of complex systems. Phys Rep 2017;714–715:1–73. http://dx.doi.org/10.1016/j.physrep.2017.10.001.

[5] Andrikopoulos A, Samitas A, Kostaris K. Four decades of the Journal of Econometrics: Coauthorship patterns and networks. J Econometrics 2016;195(1):23–32. http://dx.doi.org/10.1016/j.jeconom.2016.04.018.

[6] Thursby JG, Haeussler C, Thursby MC, Jiang L. Prepublication disclosure of scientific results: Norms, competition, and commercial orientation. Sci Adv 2018;4(5). eaar2133. http://dx.doi.org/10.1126/sciadv.aar2133.

[7] Catalini C, Fons-Fosen C, Gaulé P, Shape Collaboration. How do travel costs?. NBER working paper series, vol. 24780, 2018, http://dx.doi.org/10.3386/w24780.
[8] Baruffaldi SH, Visentin F, Conti A. The productivity of science & engineering PhD students hired from supervisors' networks. Res Policy 2016;45(4):785–96. http://dx.doi.org/10.1016/j.respol.2015.12.006.
[9] Sauermann H, Haeussler C. Authorship and contribution disclosures. Sci Adv 2017;3(11). e1700404. http://dx.doi.org/10.1126/sciadv.1700404.
[10] Gush J, Jaffe A, Larsen V, Laws A. The effect of public funding on research output: the new zealand marsden fund. New Zealand Econ Papers 2018;52(2):227–48. http://dx.doi.org/10.1080/00779954.2017.1325921.
[11] Heckman J, Moktan S. Publishing and promotion in economics: the tyranny of the top five. NBER working paper series, vol. 25093, 2018, http://dx.doi.org/10.3386/w25093.
[12] Montoya FG, Alcayde A, Baños R, Manzano-Agugliaro F. A fast method for identifying worldwide scientific collaborations using the Scopus database. Telemat Inform 2018;35(1):168–85. http://dx.doi.org/10.1016/j.tele.2017.10.010.
[13] Atmire. DSpace developers, DSpace Elsevier patch. Tech. rep., 2017, URL https://atmire.github.io/Elsevier/#/.
[14] Berry R. Eprints Scopus Screen plugin developers, Eprints Scopus Screen plugin. Tech. rep., 2012.
[15] Gentzkow M, Shapiro JM. Code and data for the social sciences: a practitioner's guide. 2014, URL https://web.stanford.edu/~gentzkow/research/CodeAndData.pdf.
[16] Reitz K, Requests Developers. Requests: HTTP for humans. Tech. rep. Cary, NC, 2019, URL https://2python-requests.org/en/master/.
[17] Hagberg A, Schult D, Swart P, Networkx developers. NetworkX. Tech. rep., 2004, URL https://networkx.github.io/documentation/stable/index.html.
[18] McKinney W. Data structures for statistical computing in python. In: S. van der Walt, J. Millman (Eds.), Proceedings of the 9th python in science conference; 2010. p. 51–6.
[19] Waskom M, Botvinnik O, O'Kane D, Hobson P, Ostblom J, Lukauskas S, Gemperline DC, Augspurger T, Halchenko Y, Cole JB, Warmenhoven J, d. Ruiter J, Pye C, Hoyer S, Vanderplas J, Villalba S, Kunter G, Quintero E, Bachant P, Martin M, Meyer K, Miles A, Ram Y, Brunner T, Yarkoni T, Williams ML, Evans C, Fitzgerald C, Brian, Qalieh A. Mwaskom/seaborn: v090. 2018, URL https://zenodo.org/record/1313201.
[20] Jones E, Oliphant T, Peterson P, SciPy developers. SciPy: open source scientific tools for python. Tech. rep., 2001, URL http://www.scipy.org/.
[21] Bird S, Loper E, Klein E. Natural language processing with python. O'Reilly Media Inc; 2009.
[22] Pedregosa F, Michel V, Grisel O, Blondel M, Prettenhofer P, Weiss R, Vanderplas J, Cournapeau D, Varoquaux G, Gramfort A, Thirion B, Grisel O, Dubourg V, Passos A, Brucher M, Perrot, Duchesnay E. Scikit-learn: machine learning in python. J Mach Learn Res 2011;12:2825–30, URL http://scikit-learn.sourceforge.net.
[23] Fortunato S, Bergstrom CT, Börner K, Evans JA, Helbing D, Milojević S, Petersen AM, Radicchi F, Sinatra R, Uzzi B, Vespignani A, Waltman L, Wang D, Barabási A-L. Science of science. Science 2018;359(6379). eaao0185. http://dx.doi.org/10.1126/science.aao0185.
[24] Stephan PE. The economics of science. J Econ Lit 2010;34:217–73. http://dx.doi.org/10.1016/S0169-7218(10)01005-1.
[25] Sandve GK, Nekrutenko A, Taylor J, Hovig E. Ten simple rules for reproducible computational research. PLoS Comput Biol 2013;9(10). e1003285. http://dx.doi.org/10.1371/journal.pcbi.1003285.