Package ‘scholar’

October 14, 2022

Type Package
Title Analyse Citation Data from Google Scholar
Version 0.2.4
Maintainer Guangchuang Yu <guangchuangyu@gmail.com>
Description Provides functions to extract citation data from Google Scholar. Convenience functions are also provided for comparing multiple scholars and predicting future h-index values.
Depends R (>= 3.5.0)
Imports R.cache, dplyr, httr, rlang, rvest, stringr, xml2, tidygraph, ggraph, ggplot2
Suggests knitr, rmarkdown, prettydoc, roxygen2, testthat (>= 2.1.0), spelling
VignetteBuilder knitr
License MIT + file LICENSE
URL https://github.com/YuLab-SMU/scholar
BugReports https://github.com/YuLab-SMU/scholar/issues
RoxygenNote 7.2.1
Encoding UTF-8
Language en-GB
NeedsCompilation no
Author Guangchuang Yu [aut, cre] (<https://orcid.org/0000-0002-6485-8781>),
James Keirstead [aut],
Gregory Jefferis [aut] (<https://orcid.org/0000-0002-0587-9355>),
Gordon Getzinger [ctb],
Jorge Cimentada [ctb],
Max Czapanskiy [ctb],
Dominique Makowski [ctb]
Repository CRAN
Date/Publication 2022-08-09 08:20:02 UTC
R topics documented:

- author_position ........................................... 2
- compare_scholars .......................................... 3
- compare_scholar_careers .................................. 4
- format_authors ............................................. 5
- format_publications ........................................ 5
- get_article_cite_history .................................. 6
- get_article_scholar_url .................................... 6
- get_citation_history ........................................ 7
- get_coauthors ............................................. 7
- get_complete_authors ...................................... 8
- get_journalrank ............................................. 9
- get_num_articles ............................................ 10
- get_num_distinct_journals ................................ 10
- get_num_top_journals ....................................... 11
- get_oldest_article .......................................... 11
- get_profile ................................................ 12
- get_publications ............................................ 13
- get_publication_abstract .................................. 14
- get_publication_data_extended ............................ 14
- get_publication_date ....................................... 15
- get_publication_url ........................................ 15
- get_scholar_id ............................................. 16
- get_scholar_resp ........................................... 16
- plot_coauthors ............................................. 17
- predict_h_index ............................................ 18
- set_scholar_mirror .......................................... 19

Index 20

author_position  Get author order.

Description

Get author rank in authors list.

Usage

author_position(authorlist, author)

Arguments

- authorlist list of publication authors
- author author's name to look for
Value
dataframe with author’s position and normalized position (a normalized index, with 0 corresponding, 1 to last and 0.5 to the middle. Note that single authorship will be considered as last, i.e., 1).

Author(s)
Dominique Makowski

Examples

library(scholar)

id <- "bg0BZ-QAAAAJ&hl"

authorlist <- scholar::get_publications(id)$author
author <- scholar::get_profile(id)$name

author_position(authorlist, author)

---

**compare_scholars**

*Compare the citation records of multiple scholars*

Description

Compares the citation records of multiple scholars. This function compiles a data frame comparing the citations received by each of the scholar’s publications by year of publication.

Usage

```r
compare_scholars(ids, pagesize = 100)
```

Arguments

- `ids`: a vector of Google Scholar IDs
- `pagesize`: an integer specifying the number of articles to fetch for each scholar

Value

da data frame giving the ID of each scholar and the total number of citations received by work published in a year.
Examples
{
## How do Richard Feynmann and Stephen Hawking compare?
ids <- c("B7vSqZsAAAAJ", "qj74uXkAAAAJ")
df <- compare_scholars(ids)
}

Description

Compare the careers of multiple scholars based on their citation histories. The scholar’s career is defined by the number of citations to his or her work in a given year (i.e. the bar chart at the top of a scholar’s profile). The function has a career option that allows users to compare scholars directly, i.e. relative to the first year in which their publications are cited.

Usage

compare_scholar_careers(ids, career = TRUE)

Arguments
	nids a character vector of Google Scholar IDs
tcareer a boolean, should a column be added to the results measuring the year relative to the first citation year. Default = TRUE

Examples
{
## How do Richard Feynmann and Stephen Hawking compare?
# Compare Feynman and Stephen Hawking
ids <- c("B7vSqZsAAAAJ", "qj74uXkAAAAJ")
df <- compare_scholar_careers(ids)
}
**format_authors**

---

### Description

This function converts first and middle names to initials

### Usage

`format_authors(string)`

### Arguments

- `string`: a character vector of names

---

**format_publications**

---

### Description

Format publication list

### Usage

`format_publications(scholar.profile, author.name = NULL)`

### Arguments

- `scholar.profile`: scholar profile ID
- `author.name`: name of author to be highlighted using bold font

### Value

- a vector of formatted publications

### Author(s)

R Thériault and modified by Guangchuang Yu

### Examples

```r
## Not run:
library(scholar)
format_publications("DO5oG40AAAAJ")
## End(Not run)
```
get_article_cite_history

*Gets the citation history of a single article*

**Description**

Gets the citation history of a single article

**Usage**

```r
get_article_cite_history(id, article)
```

**Arguments**

- `id`  a character string giving the id of the scholar
- `article`  a character string giving the article id.

**Value**

a data frame giving the year, citations per year, and publication id

get_article_scholar_url

*Gets the URL to the google scholar website of an article.

**Description**

Gets the URL to the google scholar website of an article.

**Usage**

```r
get_article_scholar_url(id, pubid)
```

**Arguments**

- `id`  a character string specifying the Google Scholar ID.
- `pubid`  a character string specifying the article id.

**Value**

a String that contains the URL to the scholar website of the article
get_citation_history

Get historical citation data for a scholar

Description

Gets the number of citations to a scholar’s articles over the past nine years.

Usage

get_citation_history(id)

Arguments

id

a character string specifying the Google Scholar ID. If multiple ids are specified, only the first value is used and a warning is generated.

Details

This information is displayed as a bar plot at the top of a standard Google Scholar page and only covers the past nine years.

Value

a data frame giving the number of citations per year to work by the given scholar

get_coauthors

Gets the network of coauthors of a scholar

Description

Gets the network of coauthors of a scholar

Usage

get_coauthors(id, n_coauthors = 5, n_deep = 1)

Arguments

id

a character string specifying the Google Scholar ID. If multiple ids are specified, only the first value is used and a warning is generated.

n_coauthors

Number of coauthors to explore. This number should usually be between 1 and 10 as choosing many coauthors can make the network graph too messy.

n_deep

The number of degrees that you want to go down the network. When n_deep is equal to 1 then grab_coauthor will only grab the coauthors of Joe and Mary, so Joe -> Mary -> All coauthors. This can get out of control very quickly if n_deep is set to 2 or above. The preferred number is 1, the default.
get_complete_authors

Details

Considering that scraping each publication for all coauthors is error prone, get_coauthors grabs only the coauthors listed on the google scholar profile (on the bottom right of the profile), not from all publications.

Value

A data frame with two columns showing all authors and coauthors.

See Also

plot_coauthors

Examples

## Not run:
library(scholar)
coauthor_network <- get_coauthors('amYIKXQA4A&hl')
plot_coauthors(coauthor_network)
## End(Not run)

get_complete_authors Get the Complete list of authors for a Publication

Description

Found as Muhammad Qasim Pasta’s solution here https://github.com/jkeirstead/scholar/issues/21

Usage

get_complete_authors(id, pubid, delay = 0.4, initials = TRUE)

Arguments

id a Google Scholar ID
pubid a Publication ID from a given google Scholar ID
delay average delay between requests. A delay is needed to stop Google identifying you as a bot
initials if TRUE (default), first and middle names will be abbreviated

Value

a string containing the complete list of authors
get_journalrank

Author(s)
Muhammad Qasim Pasta
Abram B. Fleishman
James H. Conigrave

get_journalrank

Get journal ranking.

Description
Get journal ranking for a journal list.

Usage
get_journalrank(journals, max.distance = 0.05)

Arguments
journals
   a character list giving the journal list
max.distance
   maximum distance allowed for a match between journal and journal list. Expressed either as integer, or as a fraction of the pattern length times the maximal transformation cost (will be replaced by the smallest integer not less than the corresponding fraction), or a list with possible components

Value
Journal ranking data.

Author(s)
Dominique Makowski and Guangchuang Yu

Examples
## Not run:
library(scholar)
id <- get_publications("bg0BZ-QAAAAJ&hl")
impact <- get_journalrank(journals=id$journal)
id <- cbind(id, impact)
## End(Not run)
**get_num_articles**

*Calculates how many articles a scholar has published*

**Description**

Calculate how many articles a scholar has published.

**Usage**

```r
get_num_articles(id)
```

**Arguments**

- `id` a character string giving the Google Scholar ID

**Value**

an integer value (max 100)

---

**get_num_distinct_journals**

*Gets the number of distinct journals in which a scholar has published*

**Description**

Gets the number of distinct journals in which a scholar has published. Note that Google Scholar doesn’t provide information on journals *per se*, but instead gives a title for the containing publication where applicable. So a journal here might actually be a journal, a book, a report, or some other publication outlet.

**Usage**

```r
get_num_distinct_journals(id)
```

**Arguments**

- `id` a character string giving the Google Scholar id

**Value**

the number of distinct journals
get_num_top_journals

Description

Gets the number of top journals in which a scholar has published. The definition of a 'top journal' comes from Acuna et al. and the original list was based on the field of neuroscience. This function allows users to specify that list for themselves, or use the default Acuna et al. list.

Usage

get_num_top_journals(id, journals)

Arguments

id a character string giving a Google Scholar ID

Source


get_oldest_article

Description

Gets the year of the oldest article published by a given scholar.

Usage

get_oldest_article(id)

Arguments

id a character string giving the Google Scholar ID

Value

the year of the oldest article
get_profile

Gets profile information for a scholar

Description

Gets profile information for a researcher from Google Scholar. Each scholar profile page gives the researcher’s name, affiliation, their homepage (if specified), and a summary of their key citation and publication availability metrics. The scholar ID can be found by searching Google Scholar at http://scholar.google.com.

Usage

get_profile(id)

Arguments

id a character string specifying the Google Scholar ID. If multiple ids are specified, only the first value is used and a warning is generated. See the example below for how to profile multiple scholars.

Value

a list containing the scholar’s name, affiliation, citations, impact and publication availability metrics, research interests, homepage and coauthors.

Metrics include:

- total_cites combined citations to all publications
- h_index the largest number h such that h publications each have at least h citations
- i10_index the number of publications that each have at least 10 citations
- available the number of publications that have a version online that can be read for free (though not necessarily reusable under an open access license)
- not_available the number of publications only available behind a paywall

Examples

```r
## Gets profiles of some famous physicists
ids <- c("xJaxiEEAAAAJ", "qj74uXkAAAAJ")
profiles <- lapply(ids, get_profile)
```
get_publications

Description

Gets the publications of a specified scholar.

Usage

get_publications(
  id,
  cstart = 0,
  cstop = Inf,
  pagesize = 100,
  flush = FALSE,
  sortby = "citation"
)

Arguments

id a character string specifying the Google Scholar ID. If multiple IDs are specified, only the publications of the first scholar will be retrieved.
cstart an integer specifying the first article to start counting. To get all publications for an author, omit this parameter.
cstop an integer specifying the last article to process.
pagesize an integer specifying the number of articles to fetch in one batch. It is recommended to leave the default value of 100 unless you experience time-out errors. Note this is not the total number of publications to fetch.
flush should the cache be flushed? Search results are cached by default to speed up repeated queries. If this argument is TRUE, the cache will be cleared and the data reloaded from Google.
sortby a character with value "citation" or value "year" specifying how results are sorted.

Details

Google uses two id codes to uniquely reference a publication. The results of this method includes cid which can be used to link to a publication’s full citation history (i.e. if you click on the number of citations in the main scholar profile page), and pubid which links to the details of the publication (i.e. if you click on the title of the publication in the main scholar profile page.)

Value

a data frame listing the publications and their details. These include the publication title, author, journal, number, cites, year, and two id codes (see details).
get_publication_abstract

*Gets the abstract for a publication id.*

**Description**

Gets the abstract for a publication id.

**Usage**

```r
get_publication_abstract(id, pub_id, flush = FALSE)
```

**Arguments**

- **id**: a character string specifying the Google Scholar ID.
- **pub_id**: a character string specifying the publication id.
- **flush**: Whether or not to clear the cache

**Value**

a String that contains the abstract of the publication.

gget_publication_data_extended

*Gets the full data for a publication*

**Description**

Gets the full data for a publication

**Usage**

```r
gget_publication_data_extended(id, pub_id, flush = FALSE)
```

**Arguments**

- **id**: a character string specifying the Google Scholar ID.
- **pub_id**: a character string specifying the publication id.
- **flush**: Whether or not to clear the cache

**Value**

a list that contains the full data
get_publication_date

Description
Gets the full date for a publication

Usage
get_publication_date(id, pub_id, flush = FALSE)

Arguments
id a character string specifying the Google Scholar ID.
pub_id a character string specifying the publication id.
flush Whether or not to clear the cache

Value
a String that contains the publication date

get_publication_url

Description
Gets the PDF URL for a publication id.

Usage
get_publication_url(id, pub_id, flush = FALSE)

Arguments
id a character string specifying the Google Scholar ID.
pub_id a character string specifying the publication id.
flush Whether or not to clear the cache

Value
a String that contains the URL to the PDF of the publication.
**get_scholar_id**

*Search for Google Scholar ID by name and affiliation*

**Description**

Search for Google Scholar ID by name and affiliation

**Usage**

```r
get_scholar_id(last_name = "", first_name = "", affiliation = NA)
```

**Arguments**

- **last_name**: Researcher last name.
- **first_name**: Researcher first name.
- **affiliation**: Researcher affiliation.

**Value**

Google Scholar ID as a character string.

**Examples**

```r
get_scholar_id(first_name = "kristopher", last_name = "mcneill")
get_scholar_id(first_name = "michael", last_name = "sander", affiliation = NA)
get_scholar_id(first_name = "michael", last_name = "sander", affiliation = "eth")
get_scholar_id(first_name = "michael", last_name = "sander", affiliation = "ETH Zurich")
get_scholar_id(first_name = "michael", last_name = "sander", affiliation = "Mines")
get_scholar_id(first_name = "james", last_name = "babler")
```

**get_scholar_resp**

*Recursively try to GET a Google Scholar Page storing session cookies*

**Description**

see `scholar-package` documentation for details about Scholar session cookies.

**Usage**

```r
get_scholar_resp(url, attempts_left = 5)
```

**Arguments**

- **url**: URL to fetch
- **attempts_left**: The number of times to try and fetch the page
plot_coauthors

Value

an `httr::response` object

See Also

`httr::GET`

---

**plot_coauthors**

*Plot a network of coauthors*

Description

Plot a network of coauthors

Usage

`plot_coauthors(network, size_labels = 5)`

Arguments

- `network`: A data frame given by `get_coauthors`
- `size_labels`: Size of the label names

Value

A `ggplot2` object but prints a plot as a side effect.

See Also

`get_coauthors`

Examples

```r
## Not run:
library(scholar)
coauthor_network <- get_coauthors('amYIKQAAAJ&hl')
plot_coauthors(coauthor_network)

## End(Not run)
```
predict_h_index  

Predicts the h-index for a researcher

Description

Predicts the h-index for a researcher each year for ten years into the future using Acuna et al’s method (see source). The model was fit to data from neuroscience researchers with an h-index greater than 5 and between 5 to 12 years since publishing their first article. So naturally if this isn’t you, then the results should be taken with a large pinch of salt.

Usage

predict_h_index(id, journals)

Arguments

id  
a character string giving the Google Scholar ID

journals  
optional character vector of top journals. See get_num_top_journals for more details.

Details

Since the model is calibrated to neuroscience researchers, it is entirely possible that very strange (e.g. negative) h-indices will be predicted if you are a researcher in another field. A warning will be displayed if the sequence of predicted h-indices contains a negative value or is non-increasing.

Value

a data frame giving predicted h-index values in future

Note

A scientist has an h-index of n if he or she publishes n papers with at least n citations each. Values returned are fractional so it’s up to your own vanity whether you want to round up or down.

Source


Examples

```r
## Predict h-index of original method author
## Not run:
id <- "DO5oG40AAAAJ"
df <- predict_h_index(id)

## End(Not run)
```
Description

set scholar mirror

Usage

set_scholar_mirror(mirror = NULL)

Arguments

mirror compatible scholar mirror

Details

setting google scholar mirror

Author(s)

Guangchuang Yu
Index

author_position, 2

compare_scholar_careers, 4
compare_scholars, 3

format_authors, 5
format_publications, 5

GET, 17
get_article_cite_history, 6
get_article_scholar_url, 6
get_citation_history, 7
get_coauthors, 7, 17
get_complete_authors, 8
get_journalrank, 9
get_num_articles, 10
get_num_distinct_journals, 10
get_num_top_journals, 11, 18
get_oldest_article, 11
get_profile, 12
get_publication_abstract, 14
get_publication_data_extended, 14
get_publication_date, 15
get_publication_url, 15
get_publications, 13
get_scholar_id, 16
get_scholar_resp, 16

plot_coauthors, 8, 17
predict_h_index, 18

response, 17

set_scholar_mirror, 19