Package ‘openalexR’

July 11, 2024

Type Package

Title Getting Bibliographic Records from 'OpenAlex' Database Using 'DSL' API

Version 1.4.0

Description A set of tools to extract bibliographic content from 'OpenAlex' database using API <https://docs.openalex.org>.

License MIT + file LICENSE

URL https://github.com/ropensci/openalexR, https://docs.ropensci.org/openalexR/

BugReports https://github.com/ropensci/openalexR/issues

Imports httr, jsonlite, progress, tibble

Suggests testthat (>= 3.0.0), dplyr, knitr, rmarkdown, tidyr, purrr, ggplot2, coro, covr

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Config/testthat/edition 3

Depends R (>= 2.10)

NeedsCompilation no

Author Massimo Aria [aut, cre, cph] (<https://orcid.org/0000-0002-8517-9411>), Corrado Cuccurullo [ctb] (<https://orcid.org/0000-0002-7401-8575>), Trang Le [aut] (<https://orcid.org/0000-0003-3737-6565>), June Choe [aut] (<https://orcid.org/0000-0002-0701-921X>)

Maintainer Massimo Aria <aria@unina.it>

Repository CRAN

Date/Publication 2024-07-11 12:40:02 UTC

Contents

`authors2df` *Convert OpenAlex collection of authors' records from list format to data frame*

Description

It converts bibliographic collection of authors' records gathered from OpenAlex database https://openalex.org/ into data frame. The function converts a list of authors' records obtained using `oa_request` into a data frame/tibble.

Usage

```
authors2df(
  data,
  verbose = TRUE,
  pb = if (verbose) oa_progress(length(data)) else NULL
)
```

Arguments

`data`	List. Output of `oa_request`.
`verbose`	Logical. If TRUE, print information about the dataframe conversion process. Defaults to TRUE.
`pb`	Progress bar object. If verbose, computed from ‘oa_progress‘. NULL otherwise.

Value

a data.frame.

For more extensive information about OpenAlex API, please visit: <https://docs.openalex.org>

Examples

```
## Not run:

# Query to search information about all authors affiliated to the University of Naples Federico II
# which have authored at least 100 publications:

# University of Naples Federico II is associated to the OpenAlex id I71267560.


query_author <- oa_query(
  identifier = NULL,
  entity = "authors",
  last_known_institutions.id = "I71267560",
  works_count = ">500"
)

res <- oa_request(
  query_url = query_author,
  count_only = FALSE,
  verbose = FALSE
)

df <- oa2df(res, entity = "authors")

df

## End(Not run)
```

`concepts2df` *Convert OpenAlex collection of concepts' records from list format to data frame*

Description

It converts bibliographic collection of concepts' records gathered from OpenAlex database https://openalex.org/ into data frame. The function converts a list of concepts' records obtained using `oa_request` into a data frame/tibble.

Usage

```
concepts2df(
  data,
  verbose = TRUE,
  pb = if (verbose) oa_progress(length(data)) else NULL
)
```

Arguments

`data`	List. Output of `oa_request`.
`verbose`	Logical. If TRUE, print information about the dataframe conversion process. Defaults to TRUE.
`pb`	Progress bar object. If verbose, computed from 'oa_progress'. NULL otherwise.

Value

a data.frame.

For more extensive information about OpenAlex API, please visit: <https://docs.openalex.org>

Examples

```
## Not run:

# Query to search information about all Italian educational institutions


query_inst <- oa_query(
  entity = "concepts",
  display_name.search = "electrodynamics"
)

res <- oa_request(
  query_url = query_inst,
  count_only = FALSE,
  verbose = FALSE
)

df <- oa2df(res, entity = "concepts")

df

## End(Not run)
```

`concept_abbrev` *Concepts and abbreviations.*

Description

0-level concepts and corresponding abbreviations. Reference: https://www.ncbi.nlm.nih.gov/nlmcatalog/journals/

Usage

```
concept_abbrev
```

Format

A data frame with 19 observations and 3 variables: `id`, `display_name`, and `abbreviation`.

`countrycode` *Index of Countries and their alpha-2 and aplha-3 codes.*

Description

Data frame contains the list of countries and their alpha-2 and aplha-3 codes.

Usage

```
countrycode
```

Format

A data frame with 250 rows and 3 variables:

Country country names

Alpha2 countries' alpha-2 codes

Alpha3 countries' alpha-3 codes

`funders2df` *Convert OpenAlex collection of funders' records from list format to data frame*

Description

It converts bibliographic collection of funders' records gathered from OpenAlex database https://openalex.org/ into data frame. The function converts a list of funders' records obtained using `oa_request` into a data frame/tibble.

Usage

```
funders2df(
  data,
  verbose = TRUE,
  pb = if (verbose) oa_progress(length(data)) else NULL
)
```

Arguments

`data`	List. Output of `oa_request`.
`verbose`	Logical. If TRUE, print information about the dataframe conversion process. Defaults to TRUE.
`pb`	Progress bar object. If verbose, computed from 'oa_progress'. NULL otherwise.

Value

a data.frame.

For more extensive information about OpenAlex API, please visit: <https://docs.openalex.org>

Examples

```
## Not run:

# Get funders located in Canada with more than 100,000 citations

res <- oa_request(
  "https://api.openalex.org/funders?filter=country_code:ca,cited_by_count:>100000"
)

df <- oa2df(res, entity = "funders")

df

## End(Not run)
```

institutions2df — *Convert OpenAlex collection of institutions' records from list format to data frame*

Description

It converts bibliographic collection of institutions' records gathered from OpenAlex database https://openalex.org/ into data frame. The function converts a list of institutions' records obtained using `oa_request` into a data frame/tibble.

Usage

```
institutions2df(
  data,
  verbose = TRUE,
  pb = if (verbose) oa_progress(length(data)) else NULL
)
```

Arguments

`data`	List. Output of `oa_request`.
`verbose`	Logical. If TRUE, print information about the dataframe conversion process. Defaults to TRUE.
`pb`	Progress bar object. If verbose, computed from 'oa_progress'. NULL otherwise.

Value

a data.frame.

For more extensive information about OpenAlex API, please visit: <https://docs.openalex.org>

Examples

```
## Not run:

# Query to search information about all Italian educational institutions

query_inst <- oa_query(
  entity = "institutions",
  country_code = "it",
  type = "education"
)

res <- oa_request(
  query_url = query_inst,
  count_only = FALSE,
  verbose = FALSE
)
```

```
oa2df(res, entity = "institutions")

## End(Not run)
```

`oa2bibliometrix` *Convert OpenAlex collection from data frame to bibliometrix object*

Description

It converts bibliographic collections gathered from OpenAlex database https://openalex.org/ into a bibliometrix data frame (https://bibliometrix.org/) Column names follow https://images.webofknowledge.com/images/help/W

Usage

```
oa2bibliometrix(df)
```

Arguments

`df`	is bibliographic collection of works donwloaded from OpenALex.

Value

a data.frame with class "bibliometrix".

Examples

```
## Not run:

# Query to search all works citing the article:
#  Aria, M., & Cuccurullo, C. (2017). bibliometrix:
#   An R-tool for comprehensive science mapping analysis.
#   Journal of informetrics, 11(4), 959-975.

#  published in 2021.
#  The paper is associated to the OpenAlex id W2755950973.

#  Results have to be sorted by relevance score in a descending order.

query <- oa_query(
  identifier = NULL,
  entity = "works",
  cites = "W2755950973",
  from_publication_date = "2021-01-01",
  to_publication_date = "2021-12-31",
  search = NULL,
  endpoint = "https://api.openalex.org"
)

res <- oa_request(
  query_url = query,
  count_only = FALSE,
  verbose = FALSE
)

df <- oa2df(res, entity = "works")

M <- oa2bibliometrix(df)

## End(Not run)
```

---

oa2df — *Convert OpenAlex collection from list to data frame*

Description

It converts bibliographic collections gathered from OpenAlex database https://openalex.org/ into data frame. The function converts a collection of records about works, authors, institutions, venues or concepts obtained using `oa_request` into a data frame/tibble.

Usage

```
oa2df(
  data,
  entity,
  options = NULL,
  count_only = FALSE,
  group_by = NULL,
  abstract = TRUE,
  verbose = TRUE
)
```

Arguments

`data`	List. Output of `oa_request`.
`entity`	Character. Scholarly entity of the search. The argument can be one of c("works", "authors", "institutions", "concepts", "funders", "sources", "publishers", "topics").
`options`	List. Additional parameters to add in the query. For example: - ‘select‘ Character vector. Top-level fields to show in output. Defaults to NULL, which returns all fields. https://docs.openalex.org/how-to-use-the-api/get-single-entities/select-fields - ‘sort‘ Character. Attribute to sort by. For example: "display_name" for sources or "cited_by_count:desc" for works. See more at <https://docs.openalex.org/how-to-use-the-api/get-lists-of-entities/sort-entity-lists>.

	- ‘sample‘ Integer. Number of (random) records to return. Should be no larger than 10,000. Defaults to NULL, which returns all records satisfying the query. Read more at <https://docs.openalex.org/how-to-use-the-api/get-lists-of-entities/sample-entity-lists>.
	- ‘seed‘ Integer. A seed value in order to retrieve the same set of random records in the same order when used multiple times with ‘sample‘. IMPORTANT NOTE: Depending on your query, random results with a seed value may change over time due to new records coming into OpenAlex. This argument is likely only useful when queries happen close together (within a day).
`count_only`	Logical. If TRUE, the function returns only the number of item matching the query. Defaults to FALSE.
`group_by`	Character. Attribute to group by. For example: "oa_status" for works. See more at <https://docs.openalex.org/how-to-use-the-api/get-groups-of-entities>.
`abstract`	Logical. If TRUE, the function returns also the abstract of each item. Ignored if entity is different from "works". Defaults to TRUE.
`verbose`	Logical. If TRUE, print information about the dataframe conversion process. Defaults to TRUE.

Value

A tibble/dataframe result of the original OpenAlex result list.

Examples

```
## Not run:

# Query to search all works citing the article:
#  Aria, M., & Cuccurullo, C. (2017). bibliometrix:
#   An R-tool for comprehensive science mapping analysis.
#   Journal of informetrics, 11(4), 959-975.

#  published in 2021.
#  The paper is associated to the OpenAlex id W2755950973.

#  Results have to be sorted by relevance score in a descending order.

query <- oa_query(
  entity = "works",
  cites = "W2755950973",
  from_publication_date = "2021-01-01",
  to_publication_date = "2021-04-30"
)

res <- oa_request(
  query_url = query,
  count_only = FALSE,
  verbose = FALSE
)

oa2df(res, entity = "works")
```

```
## End(Not run)
```

`oa_entities` *Available entities in the OpenAlex database*

Description

Available entities in the OpenAlex database

Usage

```
oa_entities()
```

Value

Character vector of 5 entity options.

Examples

```
oa_entities()
```

`oa_fetch` *Fetching records*

Description

A composition function to perform query building, requesting, and convert the result to a tibble/data frame.

Usage

```
oa_fetch(
  entity = if (is.null(identifier)) NULL else id_type(shorten_oaid(identifier[[1]])),
  identifier = NULL,
  ...,
  options = NULL,
  search = NULL,
  group_by = NULL,
  output = c("tibble", "dataframe", "list"),
  abstract = TRUE,
  endpoint = "https://api.openalex.org",
  per_page = 200,
  paging = NULL,
  pages = NULL,
  count_only = FALSE,
  mailto = oa_email(),
  api_key = oa_apikey(),
  verbose = FALSE
)
```

Arguments

`entity`	Character. Scholarly entity of the search. The argument can be one of c("works", "authors", "institutions", "concepts", "funders", "sources", "publishers", "topics"). If not provided, ‘entity‘ is guessed from ‘identifier‘.
`identifier`	Character. OpenAlex ID(s) as item identifier(s). See more at <https://docs.openalex.org/how-to-use-the-api/get-single-entities#the-openalex-id>.
`...`	Additional filter arguments.
`options`	List. Additional parameters to add in the query. For example: - ‘select‘ Character vector. Top-level fields to show in output. Defaults to NULL, which returns all fields. https://docs.openalex.org/how-to-use-the-api/get-single-entities/select-fields - ‘sort‘ Character. Attribute to sort by. For example: "display_name" for sources or "cited_by_count:desc" for works. See more at <https://docs.openalex.org/how-to-use-the-api/get-lists-of-entities/sort-entity-lists>. - ‘sample‘ Integer. Number of (random) records to return. Should be no larger than 10,000. Defaults to NULL, which returns all records satisfying the query. Read more at <https://docs.openalex.org/how-to-use-the-api/get-lists-of-entities/sample-entity-lists>. - ‘seed‘ Integer. A seed value in order to retrieve the same set of random records in the same order when used multiple times with ‘sample‘. IMPORTANT NOTE: Depending on your query, random results with a seed value may change over time due to new records coming into OpenAlex. This argument is likely only useful when queries happen close together (within a day).
`search`	Character. Search is just another kind of filter, one that all five endpoints support. But unlike the other filters, search does NOT require an exact match. This is particularly useful in author queries because many authors have middle names, which may not exist or do so in a variety of forms. The ‘display_name‘ filter requires an exact match and will NOT find all these authors. For example, author "Phillip H. Kuo" and "Phillip Hsin Kuo" can only be found either using search = "Phillip Kuo" or display_name = c("Phillip H. Kuo", "Phillip Hsin Kuo"). To filter using search, append .search to the end of the attribute you’re filtering for.
`group_by`	Character. Attribute to group by. For example: "oa_status" for works. See more at <https://docs.openalex.org/how-to-use-the-api/get-groups-of-entities>.
`output`	Character. Type of output, either a list or a tibble/data.frame.
`abstract`	Logical. If TRUE, the function returns also the abstract of each item. Default to `abstract = TRUE`. The argument is ignored if entity is different from "works".
`endpoint`	Character. URL of the OpenAlex Endpoint API server. Defaults to endpoint = "https://api.openalex.org".

`per_page`	Numeric. Number of items to download per page. The per-page argument can assume any number between 1 and 200. Defaults to 200.
`paging`	Character. Either "cursor" for cursor paging or "page" for basic paging. When used with ‘options$sample‘ and or ‘pages‘, paging is also automatically set to basic paging: ‘paging = "page"‘ to avoid duplicates and get the right page. See https://docs.openalex.org/how-to-use-the-api/get-lists-of-entities/paging.
`pages`	Integer vector. The range of pages to return. If NULL, return all pages.
`count_only`	Logical. If TRUE, the function returns only the number of item matching the query. Defaults to FALSE.
`mailto`	Character string. Gives OpenAlex an email to enter the polite pool.
`api_key`	Character string. Your OpenAlex Premium API key, if available.
`verbose`	Logical. If TRUE, print information on querying process. Default to `verbose = FALSE`. To shorten the printed query URL, set the environment variable openalexR.print to the number of characters to print: `Sys.setenv(openalexR.print = 70)`.

Value

A data.frame or a list. Result of the query.

Examples

```
## Not run:

paper_meta <- oa_fetch(
  identifier = "W2755950973",
  entity = "works",
  count_only = TRUE,
  abstract = TRUE,
  verbose = TRUE
)

oa_fetch(
  entity = "works",
  doi = c(
    "10.1371/journal.pone.0266781",
    "10.1371/journal.pone.0267149"
  ),
  verbose = TRUE,
  count_only = TRUE
)

oa_fetch(
  entity = "works",
  doi = c(
    "10.1371/journal.pone.0266781",
    "10.1371/journal.pone.0267149"
  ),
  options = list(select = c("doi", "id", "cited_by_count", "type")),
  verbose = TRUE
)

oa_fetch(
  identifier = c("A5069892096", "A5023888391"),
  verbose = TRUE
)

## End(Not run)
```

`oa_generate` *Iterating through records*

Description

A generator for making request to OpenAlex API Returns one record at a time.

Usage

```
oa_generate(...)
```

Arguments

`...`	arguments passed to the generator including ‘query_url‘, ‘mailto‘, ‘api_key‘, and ‘verbose‘. See ‘oa_request‘ for details on these arguments.

Value

Generator function.

Examples

```
if (require("coro")) {
  # Example 1: basic usage getting one record at a time
  query_url <- "https://api.openalex.org/works?filter=cites%3AW1160808132"
  oar <- oa_generate(query_url, verbose = TRUE)
  p1 <- oar() # record 1
  p2 <- oar() # record 2
  p3 <- oar() # record 3
  head(p1)
  head(p3)

  # Example 2: using `coro::loop()` to iterate through the generator
  query_url <- "https://api.openalex.org/works?filter=cited_by%3AW1847168837"
  oar <- oa_generate(query_url)
  coro::loop(for (x in oar) {
    print(x$id)
  })

  # Example 3: save records in blocks of 100
  query_url <- "https://api.openalex.org/works?filter=cites%3AW1160808132"
  oar <- oa_generate(query_url)
  n <- 100
  recs <- vector("list", n)
  i <- 0

  coro::loop(for (x in oar) {
    j <- i %% n + 1
    recs[[j]] <- x
    if (j == n) {
      # saveRDS(recs, sprintf("rec-%s.rds", i %/% n))
      recs <- vector("list", n) # reset recs
    }
    i <- i + 1
  })
  head(x)
  j
  # 398 works total, so j = 98 makes sense.

  # You can also manually call the generator until exhausted
  # using `while (!coro::is_exhausted(record_i))`.
  # More details at https://coro.r-lib.org/articles/generator.html.

}
```

oa_ngrams   *Get N-grams of works*

Description

Some work entities in OpenAlex include N-grams (word sequences and their frequencies) of their full text. The N-grams are obtained from Internet Archive, which uses the spaCy parser to index scholarly works. See <https://docs.openalex.org/api-entities/works/get-n-grams> for coverage and more technical details.

Usage

```
oa_ngrams(
  works_identifier,
  ...,
  endpoint = "https://api.openalex.org",
  verbose = FALSE
)
```

Arguments

`works_identifier`	Character. OpenAlex ID(s) of "works" entities as item identifier(s). These IDs start with "W". See more at <https://docs.openalex.org/api-entities/works#id>.

`...`	Unused.
`endpoint`	Character. URL of the OpenAlex Endpoint API server. Defaults to endpoint = "https://api.openalex.org".
`verbose`	Logical. If TRUE, print information on querying process. Default to `verbose = FALSE`. To shorten the printed query URL, set the environment variable openalexR.print to the number of characters to print: `Sys.setenv(openalexR.print = 70)`.

Value

A dataframe of paper metadatada and a list-column of ngrams.

Note

A faster implementation is available for ‘curl‘ >= v5.0.0, and ‘oa_ngrams‘ will issue a one-time message about this. This can be suppressed with ‘options("oa_ngrams.message.curlv5" = FALSE)‘.

Examples

```
## Not run:

ngrams_data <- oa_ngrams(c("W1963991285", "W1964141474"))

# 10 most common ngrams in the first work
first_paper_ngrams <- ngrams_data$ngrams[[1]]
first_paper_ngrams[
  order(first_paper_ngrams$ngram_count, decreasing = TRUE),
][
  1:10,
]

# Missing N-grams are `NULL` in the `ngrams` list-column
oa_ngrams("https://openalex.org/W2284876136")

## End(Not run)
```

`oa_query` *Generate an OpenAlex query from a set of parameters*

Description

It generates a valid query, written following the OpenAlex API Language, from a set of parameters.

Usage

```
oa_query(
  filter = NULL,
  multiple_id = FALSE,
  identifier = NULL,
  entity = if (is.null(identifier)) NULL else id_type(identifier[[1]]),
  options = NULL,
  search = NULL,
  group_by = NULL,
  endpoint = "https://api.openalex.org",
  verbose = FALSE,
  ...
)
```

Arguments

`filter`	Character string. Filters narrow the list down to just entities that meet a particular condition–specifically, a particular value for a particular attribute. Filters are formatted as attribute = value. The complete list of filter attributes for each entity can be found at <https://docs.openalex.org/how-to-use-the-api/get-lists-of-entities/filter-entity-lists>. For example, ‘cited_by_count = ">100"‘, ‘title.search = c("bibliometric analysis", "science mapping")‘, or ‘to_publication_date = "2021-12-31"‘.
`multiple_id`	Logical. Whether there are multiple identifiers.
`identifier`	Character. OpenAlex ID(s) as item identifier(s). See more at <https://docs.openalex.org/how-to-use-the-api/get-single-entities#the-openalex-id>.
`entity`	Character. Scholarly entity of the search. The argument can be one of c("works", "authors", "institutions", "concepts", "funders", "sources", "publishers", "topics"). If not provided, ‘entity‘ is guessed from ‘identifier‘.
`options`	List. Additional parameters to add in the query. For example: - ‘select‘ Character vector. Top-level fields to show in output. Defaults to NULL, which returns all fields. https://docs.openalex.org/how-to-use-the-api/get-single-entities/select-fields - ‘sort‘ Character. Attribute to sort by. For example: "display_name" for sources or "cited_by_count:desc" for works. See more at <https://docs.openalex.org/how-to-use-the-api/get-lists-of-entities/sort-entity-lists>. - ‘sample‘ Integer. Number of (random) records to return. Should be no larger than 10,000. Defaults to NULL, which returns all records satisfying the query. Read more at <https://docs.openalex.org/how-to-use-the-api/get-lists-of-entities/sample-entity-lists>. - ‘seed‘ Integer. A seed value in order to retrieve the same set of random records in the same order when used multiple times with ‘sample‘. IMPORTANT NOTE: Depending on your query, random results with a seed value may change over time due to new records coming into OpenAlex. This argument is likely only useful when queries happen close together (within a day).

`search`	Character. Search is just another kind of filter, one that all five endpoints support. But unlike the other filters, search does NOT require an exact match. This is particularly useful in author queries because many authors have middle names, which may not exist or do so in a variety of forms. The ‘display_name‘ filter requires an exact match and will NOT find all these authors. For example, author "Phillip H. Kuo" and "Phillip Hsin Kuo" can only be found either using search = "Phillip Kuo" or display_name = c("Phillip H. Kuo", "Phillip Hsin Kuo"). To filter using search, append .search to the end of the attribute you’re filtering for.
`group_by`	Character. Attribute to group by. For example: "oa_status" for works. See more at <https://docs.openalex.org/how-to-use-the-api/get-groups-of-entities>.
`endpoint`	Character. URL of the OpenAlex Endpoint API server. Defaults to endpoint = "https://api.openalex.org".
`verbose`	Logical. If TRUE, print information on querying process. Default to `verbose = FALSE`. To shorten the printed query URL, set the environment variable openalexR.print to the number of characters to print: `Sys.setenv(openalexR.print = 70)`.
`...`	Additional filter arguments.

Value

a character containing the query in OpenAlex format.

For more extensive information about OpenAlex API, please visit: <https://docs.openalex.org>.

Examples

```
## Not run:

query_auth <- oa_query(identifier = "A5069892096", verbose = TRUE)

### EXAMPLE 1: Full record about an entity.

# Query to obtain allinformation about a particular work/author/institution/etc.:

#  The following paper is associated to the OpenAlex-id W2755950973.

#  Aria, M., & Cuccurullo, C. (2017). bibliometrix:
#   An R-tool for comprehensive science mapping analysis.
#   Journal of informetrics, 11(4), 959-975.

query_work <- oa_query(
  identifier = "W2755950973",
  verbose = TRUE
)

#  The author Massimo Aria is associated to the OpenAlex-id A5069892096:

query_auth <- oa_query(identifier = "A5069892096", verbose = TRUE)

### EXAMPLE 2: all works citing a particular work.
```

```
# Query to search all works citing the article:
#  Aria, M., & Cuccurullo, C. (2017). bibliometrix:
#   An R-tool for comprehensive science mapping analysis.
#   Journal of informetrics, 11(4), 959-975.

#  published in 2021.
#  The paper is associated to the OpenAlex id W2755950973.

#  Results have to be sorted by relevance score in a descending order.

query1 <- oa_query(
  entity = "works",
  cites = "W2755950973",
  from_publication_date = "2021-01-01",
  to_publication_date = "2021-12-31",
  verbose = TRUE
)

### EXAMPLE 3: All works matching a string in their title

# Query to search all works containing the exact string
# "bibliometric analysis" OR "science mapping" in the title, published in the first half of 2021.

# Results have to be sorted by relevance score in a descending order.

query2 <- oa_query(
  entity = "works",
  title.search = c("bibliometric analysis", "science mapping"),
  from_publication_date = "2021-01-01",
  to_publication_date = "2021-06-30",
  options = list(sort = "cited_by_count:desc"),
  verbose = TRUE
)

## End(Not run)
```

oa_random *oa_fetch but for a random query*

Description

oa_fetch but for a random query

Usage

```
oa_random(
  entity = oa_entities(),
  output = c("tibble", "dataframe", "list"),
  endpoint = "https://api.openalex.org"
)
```

Arguments

`entity`	Character. Scholarly entity of the search. The argument can be one of c("works", "authors", "institutions", "concepts", "funders", "sources", "publishers", "topics"). If not provided, ‘entity‘ is guessed from ‘identifier‘.
`output`	Character. Type of output, either a list or a tibble/data.frame.
`endpoint`	Character. URL of the OpenAlex Endpoint API server. Defaults to endpoint = "https://api.openalex.org".

Value

A data.frame or a list. One row or one element. Result of the random query. If you would like to select more than one random entity, say, 10, use ‘options = list(sample = 10)‘ argument in ‘oa_fetch‘.

Examples

```
oa_random()
```

`oa_request` *Get bibliographic records from OpenAlex database*

Description

‘oa_request‘ makes a request and downloads bibliographic records from OpenAlex database https://openalex.org/. The function `oa_request` queries OpenAlex database using a query formulated through the function `oa_query`.

Usage

```
oa_request(
  query_url,
  per_page = 200,
  paging = "cursor",
  pages = NULL,
  count_only = FALSE,
  mailto = oa_email(),
  api_key = oa_apikey(),
  verbose = FALSE
)
```

Arguments

`query_url`	Character string. A search query formulated using the OpenAlex API language and can be generated with `oa_query`.
`per_page`	Numeric. Number of items to download per page. The per-page argument can assume any number between 1 and 200. Defaults to 200.
`paging`	Character. Either "cursor" for cursor paging or "page" for basic paging. When used with ‘options$sample‘ and or ‘pages‘, paging is also automatically set to basic paging: ‘paging = "page"‘ to avoid duplicates and get the right page. See https://docs.openalex.org/how-to-use-the-api/get-lists-of-entities/paging.
`pages`	Integer vector. The range of pages to return. If NULL, return all pages.
`count_only`	Logical. If TRUE, the function returns only the number of item matching the query. Defaults to FALSE.
`mailto`	Character string. Gives OpenAlex an email to enter the polite pool.
`api_key`	Character string. Your OpenAlex Premium API key, if available.
`verbose`	Logical. If TRUE, print information about the querying process. Defaults to TRUE.

Value

a data.frame or a list of bibliographic records.

For more extensive information about OpenAlex API, please visit: <https://docs.openalex.org>

Examples

```
## Not run:

### EXAMPLE 1: Full record about an entity.

# Query to obtain all information about a particular work/author/institution/etc.:

#  The following paper is associated to the OpenAlex-id W2755950973.

#  Aria, M., & Cuccurullo, C. (2017). bibliometrix:
#   An R-tool for comprehensive science mapping analysis.
#   Journal of informetrics, 11(4), 959-975.

res <- oa_request(query_url = "https://api.openalex.org/works/W2755950973")

#  The author Massimo Aria is associated to the OpenAlex-id A5069892096.

query_author <- oa_query(
  identifier = "A5069892096",
  entity = "authors"
)
query_author
res <- oa_request(
  query_url = query_author,
  count_only = FALSE,
  verbose = FALSE
)

### EXAMPLE 2: all works citing a particular work.

# Query to search all works citing the article:
#  Aria, M., & Cuccurullo, C. (2017). bibliometrix:
#   An R-tool for comprehensive science mapping analysis.
#   Journal of informetrics, 11(4), 959-975.

#  published in 2021.
#  The paper is associated to the OpenAlex id W2755950973.

#  Results have to be sorted by relevance score in a descending order.

query2 <- oa_query(
  identifier = NULL,
  entity = "works",
  cites = "W2755950973",
  from_publication_date = "2021-12-01",
  to_publication_date = "2021-12-31",
  search = NULL,
  endpoint = "https://api.openalex.org"
)

res2 <- oa_request(
  query_url = query2,
  count_only = FALSE,
  verbose = FALSE
)

### EXAMPLE 3: All works matching a string in their title

# Query to search all works containing the exact string
# "bibliometric analysis" OR "science mapping" in the title, published in 2020 or 2021.

# Results have to be sorted by relevance score in a descending order.


query3 <- oa_query(
  entity = "works",
  title.search = c("bibliometric analysis", "science mapping"),
  from_publication_date = "2021-12-01",
  to_publication_date = "2021-12-31"
)

res3 <- oa_request(
  query_url = query3,
  count_only = FALSE,
  verbose = FALSE
)

### EXAMPLE 4: How to check how many works match a query
```

```
# Query to search all works containing the exact string
# "bibliometric analysis" OR "science mapping" in the title, published in 2020 or 2021.

# Query only to know how many works could be retrieved (count_only=TRUE)

query4 <- oa_query(
  entity = "works",
  title.search = c("bibliometric analysis", "science mapping"),
  from_publication_date = "2020-01-01",
  to_publication_date = "2021-12-31"
)

res4 <- oa_request(
  query_url = query4,
  count_only = TRUE,
  verbose = FALSE
)

res4$count # number of items retrieved by our query

## End(Not run)
```

`oa_snowball` *A function to perform a snowball search and convert the result to a tibble/data frame.*

Description

A function to perform a snowball search and convert the result to a tibble/data frame.

Usage

```
oa_snowball(
  identifier = NULL,
  ...,
  id_type = c("short", "original"),
  mailto = oa_email(),
  endpoint = "https://api.openalex.org",
  verbose = FALSE,
  citing_params = list(),
  cited_by_params = list()
)
```

Arguments

`identifier`	Character vector of openalex identifiers.
`...`	Additional arguments to pass to ‘oa_fetch‘ when querying the input works, such as ‘doi‘.

`id_type`	Type of OpenAlex IDs to return. Defaults to "short", which remove the prefix https://openalex.org/ in the works' IDs, for example, W2755950973. If "original", the OpenAlex IDs are kept as are, for example, https://openalex.org/W2755950973
`mailto`	Character string. Gives OpenAlex an email to enter the polite pool.
`endpoint`	Character. URL of the OpenAlex Endpoint API server. Defaults to endpoint = "https://api.openalex.org".
`verbose`	Logical. If TRUE, print information on querying process. Default to `verbose = FALSE`. To shorten the printed query URL, set the environment variable openalexR.print to the number of characters to print: `Sys.setenv(openalexR.print = 70)`.
`citing_params`	parameters used in the search of works citing the input works.
`cited_by_params`	parameters used in the search of works cited by the input works.

Value

A list containing 2 elements: - nodes: dataframe with publication records. The last column 'oa_input' indicates whether the work was one of the input 'identifier'(s). - edges: publication link dataframe of 2 columns 'from, to' such that a row 'A, B' means A -> B means A cites B. In bibliometrics, the "citation action" comes from A to B.

Examples

```
## Not run:

snowball_docs <- oa_snowball(
  identifier = c("W2741809807", "W2755950973"),
  citing_params = list(from_publication_date = "2022-01-01"),
  cited_by_params = list(),
  verbose = TRUE
)

# Identical to above, but searches using paper DOIs
snowball_docs_doi <- oa_snowball(
  doi = c("10.1016/j.joi.2017.08.007", "10.7717/peerj.4375"),
  citing_params = list(from_publication_date = "2022-01-01"),
  cited_by_params = list(),
  verbose = TRUE
)


## End(Not run)
```

publishers2df — *Convert OpenAlex collection of publishers' records from list format to data frame*

Description

It converts bibliographic collection of publishers' records gathered from OpenAlex database https://openalex.org/ into data frame. The function converts a list of publishers' records obtained using `oa_request` into a data frame/tibble.

Usage

```
publishers2df(
  data,
  verbose = TRUE,
  pb = if (verbose) oa_progress(length(data)) else NULL
)
```

Arguments

`data`	List. Output of `oa_request`.
`verbose`	Logical. If TRUE, print information about the dataframe conversion process. Defaults to TRUE.
`pb`	Progress bar object. If verbose, computed from 'oa_progress'. NULL otherwise.

Value

a data.frame.

For more extensive information about OpenAlex API, please visit: <https://docs.openalex.org>

Examples

```
## Not run:

# Get publishers located in Canada with more than 100,000 citations

res <- oa_request(
  "https://api.openalex.org/publishers?filter=country_codes:ca"
)

df <- oa2df(res, entity = "publishers")

df

## End(Not run)
```

show_authors *Simplify the OpenAlex authors result*

Description

This function is mostly for the package's internal use, but we export it so you can try it out. However, we expect that you'll likely write your own function to simplify the result however you want.

Usage

```
show_authors(x, simp_func = utils::head)
```

Arguments

x	Dataframe/tibble. Result of the OpenAlex query for authors already converted to dataframe/tibble.
simp_func	R function to simplify the result. Default to ‘head‘. If you want the entire table, set ‘simp_fun = identity‘

Value

Simplified tibble to display. The first column, ‘id‘ is the short-form OpenAlex ID of the authors.

Examples

```
show_authors(oa_fetch(
  identifier = c("A5023888391", "A5014077037"),
  verbose = TRUE
))
```

show_works *Simplify the OpenAlex works result*

Description

This function is mostly for the package's internal use, but we export it so you can try it out. However, we expect that you'll likely write your own function to simplify the result however you want.

Usage

```
show_works(x, simp_func = utils::head)
```

Arguments

`x`	Dataframe/tibble. Result of the OpenAlex query for authors already converted to dataframe/tibble.
`simp_func`	R function to simplify the result. Default to ‘head‘. If you want the entire table, set ‘simp_fun = identity‘.

Value

Simplified tibble to display. The first column, ‘id‘ is the short-form OpenAlex ID of the works

Examples

```
show_works(oa_fetch(
  identifier = c("W2741809807", "W2755950973"),
  verbose = TRUE
))
```

`snowball2df` *Flatten snowball result*

Description

| id|title |...|cited_by_count| referenced_works |cited_by |...| | 100|foo |...| 1| 98, 99 |101 |...| | 200|bar |...| 2| 198, 199 |201, 202 |...| | 300|wug |...| 2| 296, 297, 298, 299 |301, 302 |...|

Usage

```
snowball2df(data, verbose = FALSE)
```

Arguments

`data`	List result from ‘oa_snowball‘.
`verbose`	Logical. If TRUE, print information on wrangling process.

Value

Tibble/data.frame of works with additional columns: append ‘citing‘, ‘backward_count‘, ‘cited_by‘, ‘forward_count‘, ‘connection‘, and ‘connection_count.‘ For each work/row, these counts are WITHIN one data search, and so ‘forward_count‘ <= ‘cited_by_count‘.

Consider the universe of all works linked to a set of starting works, (‘oa_input = TRUE‘) for each work/row i: - citing: works in the universe that i cites - backward_count: number of works in the universe that i cites - cited_by: works that i is cited by - forward_count: number of works in the universe that i is cited by - connection: works in the universe linked to i - connection_count: number of works in the universe linked to i (degree of i)

Examples

```
## Not run:
flat_snow <- snowball2df(oa_snowball(
  identifier = "W1516819724",
  verbose = TRUE
))

flat_snow[, c("id", "connection", "connection_count")]

## End(Not run)
```

`sources2df` *Convert OpenAlex collection of sources' records from list format to data frame*

Description

It converts bibliographic collection of sources' records gathered from OpenAlex database https://openalex.org/ into data frame. The function converts a list of sources' records obtained using `oa_request` into a data frame/tibble.

Usage

```
sources2df(
  data,
  verbose = TRUE,
  pb = if (verbose) oa_progress(length(data)) else NULL
)
```

Arguments

`data`	List. Output of `oa_request`.
`verbose`	Logical. If TRUE, print information about the dataframe conversion process. Defaults to TRUE.
`pb`	Progress bar object. If verbose, computed from ‘oa_progress‘. NULL otherwise.

Value

a data.frame.

For more extensive information about OpenAlex API, please visit: <https://docs.openalex.org>

Examples

```
## Not run:

# Get sources from Nature

res <- oa_request(
  "https://api.openalex.org/sources?search=nature"
)

df <- oa2df(res, entity = "sources")

df

## End(Not run)
```

topics2df — *Convert OpenAlex collection of topics' records from list format to data frame*

Description

It converts collection of topics' records gathered from the OpenAlex database. The function converts a list of topics' records obtained using `oa_request` into a data frame/tibble.

Usage

```
topics2df(
  data,
  verbose = TRUE,
  pb = if (verbose) oa_progress(length(data)) else NULL
)
```

Arguments

`data`	List. Output of `oa_request`.
`verbose`	Logical. If TRUE, print information about the dataframe conversion process. Defaults to TRUE.
`pb`	Progress bar object. If verbose, computed from 'oa_progress'. NULL otherwise.

Value

a data.frame.

For more extensive information about OpenAlex API, please visit: <https://docs.openalex.org>

Examples

```
## Not run:

# Query to search information about all Italian educational institutions


query_inst <- oa_query(
  entity = "topics",
  display_name.search = "electrodynamics"
)

res <- oa_request(
  query_url = query_inst,
  count_only = FALSE,
  verbose = FALSE
)

df <- oa2df(res, entity = "topics")

df

## End(Not run)
```

`works2df` *Convert OpenAlex collection of works from list format to data frame*

Description

It converts bibliographic collection of works gathered from OpenAlex database https://openalex.org/ into data frame. The function converts a list of works obtained using `oa_request` into a data frame/tibble.

Usage

```
works2df(
  data,
  abstract = TRUE,
  verbose = TRUE,
  pb = if (verbose) oa_progress(length(data)) else NULL
)
```

Arguments

`data`	List. Output of `oa_request`.
`abstract`	Logical. If TRUE, the function returns also the abstract of each item. Defaults to TRUE.

verbose	Logical. If TRUE, print information about the dataframe conversion process. Defaults to TRUE.
pb	Progress bar object. If verbose, computed from ‘oa_progress‘. NULL otherwise.

Value

a data.frame.

For more extensive information about OpenAlex API, please visit: <https://docs.openalex.org>

Examples

```
## Not run:

# Query to search all works citing the article:
#  Aria, M., & Cuccurullo, C. (2017). bibliometrix:
#   An R-tool for comprehensive science mapping analysis.
#   Journal of informetrics, 11(4), 959-975.

#  published in 2021.
#  The paper is associated to the OpenAlex id W2755950973.

#  Results have to be sorted by relevance score in a descending order.

query <- oa_query(
  identifier = NULL,
  entity = "works",
  cites = "W2755950973",
  from_publication_date = "2021-01-01",
  to_publication_date = "2021-12-31",
  search = NULL,
  endpoint = "https://api.openalex.org"
)

res <- oa_request(
  query_url = query,
  count_only = FALSE,
  verbose = FALSE
)

df <- oa2df(res, entity = "works")

df

## End(Not run)
```

Index