Package 'easyPubMed'

July 22, 2025

Type Package

Title Search and Retrieve Scientific Publication Records from PubMed

Version 2.13

Date 2019-03-25

Author Damiano Fantini

Maintainer Damiano Fantini <damiano.fantini@gmail.com>

Description Query NCBI Entrez and retrieve PubMed records in XML or text format. Process PubMed records by extracting and aggregating data from selected fields. A large number of records can be easily downloaded via this simple-to-use interface to the NCBI PubMed API.

URL https://www.data-pulse.com/dev_site/easypubmed/

Depends R(>= 3.1), utils

Suggests knitr, rmarkdown

VignetteBuilder knitr

LazyData true

Encoding UTF-8

License GPL-2

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-03-29 09:00:02 UTC

Contents

`articles_to_list` *Cast PubMed Data into a List of Articles*

Description

Convert an XML object of PubMed records into a list of strings (character vector of length 1) corresponding to individual PubMed articles. PubMed records are identified by a "/PubmedArticle" XML tag. This automatically casts all the content of each PubMed record to a character-class object without removing XML tags.

Usage

```
articles_to_list(pubmed_data, encoding = "UTF8", simplify = TRUE)
```

Arguments

`pubmed_data`	String corresponding to the name of an XML file (typically, the result of a batch_pubmed_download() call). Alternatively, a string including PubMed records with XML tags, such as the object returned by a fetch_pubmed_data() call.
`encoding`	The encoding of an input/output connection can be specified by name (for example, "ASCII", or "UTF-8", in the same way as it would be given to the function base::iconv(). See iconv() help page for how to find out more about encodings that can be used on your platform. "UTF-8" is recommended.
`simplify`	Logical; should the result be simplified to a character vector. If FALSE, results are returned as a list.

Details

The input is an XML object or a string including PubMed records (with XML tags). These are the output of easyPubMed functions: fetch_pubmed_data() or batch_pubmed_download(). The function returns a list or a character vector where each element is a different PubMed record.

Value

List or character vector including all the records from the original XML object in text format. Elements in the list are not named and are only accessible via their numeric index.

Author(s)

Damiano Fantini <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/easypubmed/

Examples

```
try({
  ## Retrieve PubMed data and return a list ot articles
  my_query <- "Damiano Fantini[AU]"
  my_query <- get_pubmed_ids(pubmed_query_string = my_query)
  my_data <- fetch_pubmed_data(my_query, encoding = "ASCII")
  listed_articles <- articles_to_list(my_data)
  custom_grep(listed_articles[[2]], "ArticleTitle", "char")
}, silent = TRUE)

## Not run:
## Download PubMed data and return a list ot articles
dami_query <- "Damiano Fantini[AU] AND 2018[PDAT]"
outfile <- batch_pubmed_download(dami_query, dest_file_prefix = "easyPM_ex001_")
listed_articles <- articles_to_list(pubmed_data = outfile)
custom_grep(listed_articles[[2]], "ArticleTitle", "char")

## End(Not run)
```

---

article_to_df *Extract Data from a PubMed Record*

Description

Extract publication-specific information from a PubMed record driven by XML tags. The input record is a string (character-class vector of length 1) and includes PubMed-specific XML tags. Data are returned as a data frame where each row corresponds to one of the authors of the PubMed article.

Usage

```
article_to_df(pubmedArticle, autofill = FALSE,
                      max_chars = 500, getKeywords = FALSE,
                      getAuthors = TRUE)
```

Arguments

pubmedArticle	String including one PubMed record.
autofill	Logical. If TRUE, missing affiliations are automatically imputed based on other non-NA addresses from the same record.
max_chars	Numeric (integer). Maximum number of characters to be extracted from the Article Abstract field. Set max_chars to -1 for extracting the full-length abstract. Set max_chars to 0 to extract no abstract.

getKeywords	Logical. If TRUE, an attempt to extract article Keywords will be made.
getAuthors	Logical. If FALSE, author information won't be extracted. This will considerably speed up the operation.

Details

Given one Pubmed Article record, this function will automatically extract a set of features. Extracted information include: PMID, DOI, article title, article abstract, publication date (year, month, day), journal name (title, abbreviation), keywords, and a set of author-specific info (names, affiliation, email address). Each row of the output data frame corresponds to one of the authors of the PubMed record. Author-independent info (publication ID, title, journal, date) are identical across all rows. If information about authors are not required, set 'getAuthors' = TRUE.

Value

Data frame including the extracted features. Each row correspond a different author.

Author(s)

Damiano Fantini <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/easypubmed/

Examples

```
try({
  ## Display some contents
  data("EPMsamples")
  #display Query String used for collecting the data
  print(EPMsamples$NUBL_1618$qry_st)
  #Get records
  BL_list <- EPMsamples$NUBL_1618$rec_lst
  cat(BL_list[[1]])
  # cast PM recort to data.frame
  BL_df <- article_to_df(BL_list[[1]], max_chars = 0)
  print(BL_df)
}, silent = TRUE)

## Not run:
## Query PubMed, retrieve a selected citation and format it as a data frame
dami_query <- "Damiano Fantini[AU] AND 2017[PDAT]"
dami_on_pubmed <- get_pubmed_ids(dami_query)
dami_abstracts_xml <- fetch_pubmed_data(dami_on_pubmed)
dami_abstracts_list <- articles_to_list(dami_abstracts_xml)
article_to_df(pubmedArticle = dami_abstracts_list[[1]], autofill = FALSE)
article_to_df(pubmedArticle = dami_abstracts_list[[2]], autofill = TRUE, max_chars = 300)[1:2,]

## End(Not run)
```

`batch_pubmed_download` *Download PubMed Records in XML or TXT Format*

Description

Performs a PubMed Query (via the get_pubmed_ids() function), downloads the resulting data (via multiple fetch_pubmed_data() calls) and then saves data in a series of xml or txt files on the local drive. The function is suitable for downloading a very large number of records.

Usage

```
batch_pubmed_download(pubmed_query_string, dest_dir = NULL,
                              dest_file_prefix = "easyPubMed_data_",
                              format = "xml", api_key = NULL,
                              batch_size = 400, res_cn = 1,
                              encoding = "UTF8")
```

Arguments

`pubmed_query_string`
: String (character-vector of length 1): this is the string used for querying PubMed (the standard PubMed Query synthax applies).

`dest_dir`
: String (character-vector of length 1): this string corresponds to the name of the existing folder where files will be saved. Existing files will be overwritten. If NULL, the current working directory will be used.

`dest_file_prefix`
: String (character-vector of length 1): this string is used as prefix for the files that are written locally.

`format`
: String (character-vector of length 1): data will be requested from Entrez in this format. Acceptable values are: c("medline","uilist","abstract","asn.1", "xml"). When format != "xml", data will be saved as text files (txt).

`api_key`
: String (character vector of length 1): user-specific API key to increase the limit of queries per second. You can obtain your key from NCBI.

`batch_size`
: Integer (1 < batch_size < 5000): maximum number of records to be saved in a single xml or txt file.

`res_cn`
: Integer (> 0): numeric index of the data batch to start downloading from. This parameter is useful to resume an incomplete download job after a system crash.

`encoding`
: The encoding of an input/output connection can be specified by name (for example, "ASCII", or "UTF-8", in the same way as it would be given to the function base::iconv(). See iconv() help page for how to find out more about encodings that can be used on your platform. Here, we recommend using "UTF-8".

Details

Download large number of PubMed records as a set of xml or txt files that are saved in the folder specified by the user. This function enforces data integrity. If a batch of downloaded data is corrupted, it is discarded and downloaded again. Each download cycle is monitored until the download job is successfully completed. This function should allow to download a whole copy of PubMed, if desired. The function informs the user about the current progress by constantly printing to console the number of batches still in queue for download. pubmed_query_string accepts standard PubMed synthax. The function will query PubMed multiple times using the same query string. Therefore, it is recommended to use a [EDAT] or a [PDAT] filter in the query if you want to ensure reproducible results.

Value

Character vector including the names of files downloaded to the local system

Author(s)

Damiano Fantini <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/easypubmed/

Examples

```
## Not run:
## Example 01: retrieve data from PubMed and save as XML file
ml_query <- "Machine Learning[TI] AND 2016[PD]"
out1 <- batch_pubmed_download(pubmed_query_string = ml_query, batch_size = 180)
readLines(out1[1])[1:30]
##
## Example 02: retrieve data from PubMed and save as TXT file
ml_query <- "Machine Learning[TI] AND 2016[PD]"
out2 <- batch_pubmed_download(pubmed_query_string = ml_query, batch_size = 180, format = "medline")
readLines(out2[1])[1:30]

## End(Not run)
```

---

custom_grep *Retrieve Text Between XML Tags*

Description

Extract text form a string containing XML or HTML tags. Text included between tags of interest will be returned. If multiple tagged substrings are found, they will be returned as different elements of a list or character vector.

Usage

```
custom_grep(xml_data, tag, format = "list")
```

Arguments

xml_data	String (of class character and length 1): corresponds to the PubMed record or any string including XML/HTML tags.
tag	String (of class character and length 1): the tag of interest (does NOT include < > chars).
format	c("list", "char"): specifies the format for the output.

Details

The input string has to be a character string (length 1) containing tags (HTML or XML format). If an XML Document is provided as input, the function will rise an error.

Value

List or vector where each element corresponds to an in-tag substring.

Author(s)

Damiano Fantini <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/easypubmed/

Examples

```
try({
  ## extract substrings based on regular expressions
  string_01 <- "I can't wait to watch the <strong>Late Night Show with"
 string_01 <- paste(string_01, "Seth Meyers</strong> tonight at <strong>11:30</strong>pm CT!")
  print(string_01)
  custom_grep(xml_data = string_01, tag = "strong", format = "char")
  custom_grep(xml_data = string_01, tag = "strong", format = "list")
}, silent = TRUE)
```

EPMsamples *PubMed Records downloaded and analyzed via easyPubMed*

Description

This dataset includes a collection of 4 examples showing how to download and analyze records from PubMed by using easyPubMed. Each element in the EPMsamples list corresponds to a different query and/or analysis. Also, each element of EPMsamples is a list including intermediates and notes about the analysis.

Usage

```
data("EPMsamples")
```

Format

The dataset is formatted as a list including 4 elements:

* 'DF_papers_abs': List of 4

* 'DF_papers_std': List of 4

* 'NUBL_dw18': List of 3

* 'NUBL_1618': List of 5

Details

The dataset was built as described in this vignette: https://www.data-pulse.com/projects/Rlibs/vignettes/building_the_easyPubMed_EPMsamples_dataset.html

Examples

```
## Display some contents
data("EPMsamples")
# The following examples are focused on example query #4 (i.e., NUBL_1618)
# Display Query String used for collecting the data
print(EPMsamples$NUBL_1618$qry_st)
# show one PubMed record element from the IL vector
NU_records <- EPMsamples$NUBL_1618$rec_lst
cat(NU_records[[1]])
# cast PM recort to data.frame
BL_df <- article_to_df(NU_records[[6]], max_chars = 0)
print(BL_df)
```

fetch_all_pubmed_ids *Retrieve All PubMed Record Identifiers Returned by a Query*

Description

Retrieve PubMed record identifiers from Entrez following a search performed via the get_pubmed_ids() function. Identifiers are returned as a character vector.

Usage

```
fetch_all_pubmed_ids(pubmed_id_list)
```

Arguments

pubmed_id_list	List: the result of a get_pubmed_ids() call.

Details

Retrieve PubMed identifiers, without any other information (such as article title, authors, publication date, and so on). The PubMed IDs can be stored or used with other software.

Value

Character vector including all PMID (PubMed Identifiers) returned by the current query.

Author(s)

Damiano Fantini <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/easypubmed/

Examples

```
## Not run:
## Fetch only PubMed Record IDs (PMIDs)
dami_query_string <- "Damiano Fantini[AU]"
dami_on_pubmed <- get_pubmed_ids(dami_query_string)
dami_pmids <- fetch_all_pubmed_ids(dami_on_pubmed)
print(dami_pmids)


## End(Not run)
```

`fetch_pubmed_data` *Retrieve PubMed Data in XML or TXT Format*

Description

Retrieve PubMed records from Entrez following a search performed via the get_pubmed_ids() function. Data are downloaded in the XML or TXT format and are retrieved in batches of up to 5000 records.

Usage

```
fetch_pubmed_data(pubmed_id_list,
                          retstart = 0,
                          retmax = 500,
                          format = "xml",
                          encoding = "UTF8")
```

Arguments

`pubmed_id_list`	List: the result of a get_pubmed_ids() call.
`retstart`	Integer (>=0): index of the first UID in the retrieved PubMed Search Result set to be included in the output (default=0, corresponding to the first record of the entire set).
`retmax`	Integer (>=1): size of the batch of PubMed records to be retrieved at one time.
`format`	Character: element specifying the output format. The following values are allowed: c("asn.1", "xml", "medline", "uilist", "abstract").
`encoding`	The encoding of an input/output connection can be specified by name (for example, "ASCII", or "UTF-8", in the same way as it would be given to the function base::iconv(). See iconv() help page for how to find out more about encodings that can be used on your platform. Here, we recommend using "UTF-8".

Details

Retrieve PubMed records based on the results of a get_pubmed_ids() query. Records are retrieved from Entrez via the PubMed API efetch function. The first entry to be retrieved may be adjusted via the retastart parameter (this allows the user to download large batches of PubMed data in multiple runs). The maximum number of entries to be retrieved can also be set adjusting the retmax parameter (1 < retmax < 5000). Data will be downloaded on the fly (no files are saved locally).

Value

An object (vector) of class "character". If format is set to "xml" (default), a single String including all PubMed records (with XML tags embedded) is returned. If a different format is selected, a vector of strings is returned, where each row corresponds to a line of the output document.

Author(s)

Damiano Fantini <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/easypubmed/ https://www.ncbi.nlm.nih.gov/books/NBK25499/table/chapter4.T._valid_values_of__retmode_and/

Examples

```
try({
  ## Example 01: retrieve data in TXT format
  library("easyPubMed")
  dami_query_string <- "Damiano Fantini[AU] AND 2018[PDAT]"
  dami_on_pubmed <- get_pubmed_ids(dami_query_string)
  Sys.sleep(1) # avoid server timeout
  dami_papers <- fetch_pubmed_data(dami_on_pubmed, format = "abstract")
  dami_papers[dami_papers == ""] <- "\n"
  cat(paste(dami_papers[1:65], collapse = ""))
}, silent = TRUE)

## Not run:
## Example 02: retrieve data in XML format
library("easyPubMed")
dami_query_string <- "Damiano Fantini[AU]"
dami_on_pubmed <- get_pubmed_ids(dami_query_string)
dami_papers <- fetch_pubmed_data(dami_on_pubmed)
titles <- custom_grep(dami_papers, "ArticleTitle", "char")
print(titles)

## End(Not run)
```

---

get_pubmed_ids *Simple PubMed Record Search*

Description

Query PubMed (Entrez) in a simple way via the PubMed API eSearch function. Calling this function results in posting the query results on the PubMed History Server. This allows later access to the resulting data via the fetch_pubmed_data() function, or other easyPubMed functions.

Usage

```
get_pubmed_ids(pubmed_query_string, api_key = NULL)
```

Arguments

pubmed_query_string	is a string (character vector of length 1) that is used for querying PubMed (standard PubMed synthax, see reference for details).
api_key	String (character vector of length 1): user-specific API key to increase the limit of queries per second. You can obtain your key from NCBI.

Details

This function will use the String provided as argument for querying PubMed via the eSearch function of the PubMed API. The Query Term can include one or multiple words, as well as the standard PubMed operators (AND, OR, NOT) and tags (i.e., [AU], [PDAT], [Affiliation], and so on). ESearch will post the UIDs resulting from the search operation onto the History server so that they can be used directly in a subsequent fetchPubmedData() call.

Value

The function returns a list. The list includes the number of records found on PubMed and the first 20 PubMed IDs (UID) retrieved by the query. The list also includes QueryKey and WebEnv that are required for a subsequent fetch_pubmed_data() call.

Author(s)

Damiano Fantini <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/easypubmed/ https://www.ncbi.nlm.nih.gov/books/NBK3827/#_pubmedhelp_Search_Field_Descriptions_and_

Examples

```
try({
  ##  Search for scientific articles written by Damiano Fantini
  ##  and print the number of retrieved records to screen.
  ##  Also print the retrieved UIDs to screen.
  ##
  dami_on_pubmed <- get_pubmed_ids("Damiano Fantini[AU]")
  print(dami_on_pubmed$Count)
  print(unlist(dami_on_pubmed$IdList))
}, silent = TRUE)
```

`get_pubmed_ids_by_fulltitle` *Simple PubMed Record Search by Full-length Title*

Description

Query PubMed (Entrez) in a simple way via the PubMed API eSearch function. This function is designed to query PubMed using a full-length publication title as query string. It performs stopword removal from the query string before querying the PubMed server. Calling this function results in posting the results on the PubMed History Server. This allows later access to the resulting data via the fetch_pubmed_data() function, or other easyPubMed functions.

Usage

```
get_pubmed_ids_by_fulltitle(fulltitle, field = "[Title]", api_key = NULL)
```

Arguments

`fulltitle`	String (character vector of length 1) that corresponds to the full-length publication title used for querying PubMed (titles should be used as is, without adding extra filters/tags).
`field`	String (character vector of length 1) with a tag indicating the PubMed record field where the full-length string (fulltitle) should be searched in. By default, this points to the 'Title' field. This field can be changed (use fields supported by PubMed) as required by the user (for example, to attempt an exact-match query using a specific sentence included in the abstract of a record).
`api_key`	String (character vector of length 1): user-specific API key to increase the limit of queries per second. You can obtain your key from NCBI.

Details

This function will use the String provided as argument for querying PubMed via the eSearch function of the PubMed API. The Query Term should include a full-length publication title, without other PubMed operators (AND, OR, NOT) nor tags (i.e., [AU], [PDAT], [Affiliation], and so on). ESearch will post the UIDs resulting from the search operation onto the History server so that they can be used directly in a subsequent fetchPubmedData() call.

Value

The function returns a list. The list includes the number of records found on PubMed and the first 20 PubMed IDs (UID) retrieved by the query. The list also includes QueryKey and WebEnv that are required for a subsequent fetch_pubmed_data() call.

Author(s)

Damiano Fantini <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/easypubmed/

Examples

```
## Not run:
##  Search for a scientific article matching a full-length title
my_query <- "Body mass index and cancer risk among Chinese patients with type 2 diabetes mellitus"
my_field <- "[Title]"
# Full-length title query (designed to query titles)
res0 <- get_pubmed_ids(my_query)
print(as.numeric(res0$Count))
# Weird count!
res <- get_pubmed_ids_by_fulltitle(my_query, field = my_field)
# Num results = 1 as expected
print(as.numeric(res$Count))


## End(Not run)
```

PubMed_stopwords *PubMed Records about Bladder Research from Northwestern University*

Description

This dataset includes a collection of 87 PubMed Records of scientific publications about Bladder biology and pathology, published by clinical and research groups from Northwestern University (Chicago, IL), between 2016 and 2018.

Usage

```
data("PubMed_stopwords")
```

Format

A character vector including all PubMed stopwords tat are typically filtered out from queries.

Details

Number of stopwords included, n=133.

Examples

```
## Display some contents
data("PubMed_stopwords")
head(PubMed_stopwords)
```

table_articles_byAuth *Extract Publication and Affiliation Data from PubMed Records*

Description

Extract Publication Info from PubMed records and cast data into a data.frame where each row corresponds to a different author. It is possible to limit data extraction to first authors or last authors only, or get information about all authors of each PubMed record.

Usage

```
table_articles_byAuth(pubmed_data,
                              included_authors = "all",
                              max_chars = 500,
                              autofill = TRUE,
                              dest_file = NULL,
                              getKeywords = TRUE,
                              encoding = "UTF8")
```

Arguments

pubmed_data	PubMed Data in XML format: typically, an XML file resulting from a batch_pubmed_download() call or an XML object, result of a fetch_pubmed_data() call.
included_authors	Character: c("first", "last", "all"). Only includes information from the first, the last or all authors of a PubMed record.
max_chars	Numeric: maximum number of chars to extract from the AbstractText field.
autofill	Logical. If TRUE, missing affiliations are imputed according to the available values (from the same article).
dest_file	String (character of length 1). Name of the file that will be written for storing the output. If NULL, no file will be saved.
getKeywords	Logical. If TRUE, the operation will attempt to extract PubMed record keywords (MESH topics, keywords).
encoding	The encoding of an input/output connection can be specified by name (for example, "ASCII", or "UTF-8", in the same way as it would be given to the function base::iconv(). See iconv() help page for how to find out more about encodings that can be used on your platform. Here, we recommend using "UTF-8".

Details

Retrieve publication and author information from PubMed data, and cast them as a data.frame.

Value

Data frame including the following fields: c("article.title","article.abstract", "date.year", "date.month", "date.day", "journal.abbrv", "journal.title", "keywords", "auth.last", "auth.fore", "auth.address", "auth.email").

Author(s)

Damiano Fantini <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/easypubmed/

Examples

```
## Not run:
## Cast PubMed record info into a data.frame

dami_query <- "Damiano Fantini[AU]"
dami_on_pubmed <- get_pubmed_ids(dami_query)
dami_abstracts_xml <- fetch_pubmed_data(dami_on_pubmed, encoding = "ASCII")
xx <- table_articles_byAuth(pubmed_data = dami_abstracts_xml,
                            included_authors = "first",
                            max_chars = 100,
                            autofill = TRUE)

print(xx[1:5, c("pmid", "lastname", "jabbrv")])
#
## Download records first
## Also, auto-fill disabled
dami_query <- "Damiano Fantini[AU]"
curr.file <- batch_pubmed_download(dami_query, dest_file_prefix = "test_bpd_", encoding = "ASCII")
xx <- table_articles_byAuth(pubmed_data = curr.file[1],
                            included_authors = "all",
                            max_chars = 20,
                            autofill = FALSE)
print(xx[1:5, c("pmid", "lastname", "jabbrv")])


## End(Not run)
```

---

trim_address *Trim and Format Address Information*

Description

Set of rules for trimming and standardizing the format of address information retrieved from PubMed records. Affiliations including more than one address will be trimmend and only the first address will be returned.

Usage

```
trim_address(addr)
```

Arguments

addr	Character string including an address as extracted from PubMed records.

Value

Character string including a formatted and trimmed address (if available).

Author(s)

Damiano Fantini <damiano.fantini@gmail.com>

References

https://www.data-pulse.com/dev_site/easypubmed/

Examples

```
addr_string <- " 2 Dept of Urology, Feinberg School of Medicine,"
addr_string <- paste(addr_string, "Chicago, US; Dept of Mol Bio as well...")
print(addr_string)
print(trim_address(addr = addr_string))
```

Index