Package ‘LexisNexisTools’

June 3, 2020

Title Working with Files from 'LexisNexis'

Version 0.3.1

Date 2020-06-03

Description My PhD supervisor once told me that everyone doing newspaper analysis starts by writing code to read in files from the 'LexisNexis' newspaper archive (retrieved e.g., from <http://www.nexis.com/> or any of the partner sites). However, while this is a nice exercise I do recommend, not everyone has the time. This package takes files downloaded from the newspaper archive of 'LexisNexis', reads them into R and offers functions for further processing.

Depends R (>= 3.5.0)

License GPL-3

Imports data.table (>= 1.10.0), methods (>= 3.3.0), parallel (>= 3.3.0), pbapply (>= 1.3.4), quanteda (>= 1.1.0), stats (>= 3.3.0), stringdist (>= 0.9.4.0), stringi (>= 1.1.7), tibble (>= 1.4.0), utils (>= 3.3.0)

Suggests corpustools, covr, diffobj, dplyr, RSQLite, testthat, tidytext, tm, kableExtra, knitr, pdftools, rmarkdown, spelling, striprtf, xml2

Encoding UTF-8

LazyData true

URL `https://github.com/JBGruber/LexisNexisTools`

BugReports `https://github.com/JBGruber/LexisNexisTools/issues`

RoxygenNote 7.1.0

VignetteBuilder knitr

Language en-GB

NeedsCompilation no

Author Johannes Gruber [aut, cre]

Maintainer Johannes Gruber `<j.gruber.1@research.gla.ac.uk>`

Repository CRAN

Date/Publication 2020-06-03 18:40:07 UTC

R topics documented:

lnt2bibtex *Convert LNToutput to other formats*

Description

Takes output from lnt_read and converts chosen articles to a BibTeX citation.

Usage

```
lnt2bibtex(x, art_id, ...)
```

Arguments

x	An object of class LNToutput.
art_id	The ID(s) of the article(s) to convert.
...	unused.

Examples

```
LNToutput <- lnt_read(lnt_sample(copy = FALSE))

bib <- lnt2bibtex(LNToutput, art_id = 1)
```

LNToutput — *An S4 class to store the three data.frames created with lnt_read*

Description

This S4 class stores the output from lnt_read. Just like a spreadsheet with multiple worksheets, an LNToutput object consist of three data.frames which you can select using `@`. This object class is intended to be an intermediate container. As it stores articles and paragraphs in two separate data.frames, nested in an S4 object, the relevant text data is stored twice in almost the same format. This has the advantage, that there is no need to use special characters, such as "\n" to indicate a new paragraph. However, it makes the files rather big when you save them directly. They should thus usually be subsetted using `@` or converted to a different format using lnt_convert.

Slots

`meta` The metadata of the articles read in.

`articles` The article texts and respective IDs.

`paragraphs` The paragraphs (if the data.frame exists) and respective article and paragraph IDs.

LNToutput_methods — *Methods for LNToutput output objects*

Description

Methods for LNToutput output objects

Usage

```
## S4 method for signature 'LNToutput'
dim(x)

## S4 method for signature 'LNToutput'
show(object)

## S4 method for signature 'LNToutput,ANY,ANY,ANY'
x[i, j, invert = FALSE]

## S4 method for signature 'LNToutput,LNToutput'
e1 + e2
```

Arguments

`x, object`	An LNToutput object.
`i`	Rows of the meta data.frame (default) or values of j.
`j`	The column you want to use to subset the LNToutput object. Takes character strings.
`invert`	Invert the selection of i.
`e1, e2`	LNToutput objects which will be combined.

`lnt_add` *Adds or replaces articles*

Description

This functions adds a dataframe to a slot in an LNToutput object or overwrites existing entries. The main use of the function is to add an extract of one of the data.frames back to an LNToutput object after operations were performed on it.

Usage

```
lnt_add(to, what, where = "meta", replace = TRUE)
```

Arguments

`to`	an LNToutput object to which something should be added.
`what`	A data.frame which is added.
`where`	Either "meta", "articles" or "paragraphs" to indicate the slot to which data is added.
`replace`	If TRUE, will overwrite entries which have the same ID as

Details

Note, that when adding paragraphs, the Par_ID column is used to determine if entries are already present in the set. For the other data frames the article ID is used.

Author(s)

Johannes Gruber

Examples

```
# Make LNToutput object from sample
LNToutput <- lnt_read(lnt_sample(copy = FALSE))

# extract meta and make corrections
correction <- LNToutput@meta[grepl("Wikipedia", LNToutput@meta$Headline), ]
correction$Newspaper <- "Wikipedia"

# replace corrected meta information
LNToutput <- lnt_add(to = LNToutput, what = correction, where = "meta", replace = TRUE)
```

`lnt_asDate` *Convert Strings to dates*

Description

Converts dates from string formats common in LexisNexis to a date object.

Usage

```
lnt_asDate(x, format = "auto", locale = "auto", ...)
```

Arguments

`x`	A character object to be converted.
`format`	Either "auto" to guess the format based on a common order of day, month and year or provide a custom format (see stri_datetime_format for format options).
`locale`	A ISO 639-1 locale code (see `https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes`).
`...`	Not used.

Value

This function returns an object of class date.

Examples

```
LNToutput <- lnt_read(lnt_sample(copy = FALSE), convert_date = FALSE)
d <- lnt_asDate(LNToutput@meta$Date)
d
```

lnt_convert — *Convert LNToutput to other formats*

Description

Takes output from lnt_read and converts it to other formats. You can either use `lnt_convert()` and choose the output format via `to` or use the individual functions directly.

Usage

```
lnt_convert(
  x,
  to = "data.frame",
  what = "articles",
  collapse = FALSE,
  file = "LNT.sqlite",
  ...
)

lnt2df(x, what = "articles", ...)

lnt2rDNA(x, what = "articles", collapse = TRUE)

lnt2quanteda(x, what = "articles", collapse = NULL, ...)

lnt2tm(x, what = "articles", collapse = NULL, ...)

lnt2cptools(x, what = "articles", ...)

lnt2tidy(x, what = "articles", ...)

lnt2SQLite(x, file = "LNT.sqlite", ...)
```

Arguments

`x`	An object of class LNToutput.
`to`	Which format to convert into. Possible values are "rDNA", "corpustools", "tidytext", "tm", "SQLite" and "quanteda".
`what`	Either "articles" or "paragraphs" to use articles or paragraphs as text in the output object.
`collapse`	Only has an effect when `what = "articles"`. If set to TRUE, an empty line will be added after each paragraphs. Alternatively you can enter a custom string (such as `"\n"` for newline). `NULL` or `FALSE` turns off this feature.
`file`	The name of the database to be written to (for lnt2SQLite only).
`...`	Passed on to different methods (see details).

Details

lnt_convert() provides conversion methods into several formats commonly used in prominent R packages for text analysis. Besides the options set here, the ... (ellipsis) is passed on to the individual methods for tuning the outcome:

- data.frame, rDNA ... not used.
- quanteda ... passed on to `quanteda::corpus()`.
- corpustools ... passed on to `corpustools::create_tcorpus()`.
- tm ... passed on to `tm::Corpus()`.
- tidytext ... passed on to `tidytext::unnest_tokens()`.
- lnt2SQLite ... passed on to `RSQLite::dbWriteTable()`.

Examples

```
LNToutput <- lnt_read(lnt_sample(copy = FALSE))

df <- lnt_convert(LNToutput, to = "data.frame")

docs <- lnt_convert(LNToutput, to = "rDNA")

corpus <- lnt_convert(LNToutput, to = "quanteda")

tCorpus <- lnt_convert(LNToutput, to = "corpustools")

tidy <- lnt_convert(LNToutput, to = "tidytext")

Corpus <- lnt_convert(LNToutput, to = "tm")

## Not run:
dbloc <- lnt_convert(LNToutput, to = "SQLite")

## End(Not run)
```

`lnt_diff` *Display diff of similar articles*

Description

This function is a wrapper for diffPrint. It is intended to help performing a manual assessment of the difference between highly similar articles identified via lnt_similarity.

Usage

```
lnt_diff(x, min = 0.15, max = 0.3, n = 25, output_html = FALSE, ...)
```

Arguments

`x`	lnt_sim object as returned by lnt_similarity.
`min`	Minimum value of rel_dist to include in diff.
`max`	Maximum value of rel_dist to include in diff.
`n`	Size of displayed sample.
`output_html`	Set to TRUE to output html code, e.g. to use for knitting an rmarkdown document to html. Chunk option must be set to `results='asis'` in that case.
`...`	Currently not used.

Author(s)

Johannes Gruber

Examples

```
# Test similarity of articles
duplicates.df <- lnt_similarity(
  LNToutput = lnt_read(lnt_sample(copy = FALSE)),
  threshold = 0.97
)

lnt_diff(duplicates.df, min = 0.18, max = 0.30)
```

`lnt_lookup` *Lookup keywords in articles*

Description

This function looks for the provided pattern in the string or LNToutput object. This can be useful, for example, to see which of the keywords you used when retrieving the data was used in each article.

Usage

```
lnt_lookup(
  x,
  pattern,
  case_insensitive = FALSE,
  unique_pattern = FALSE,
  word_boundaries = c("both", "before", "after"),
  cores = NULL,
  verbose = TRUE
)
```

Arguments

x	An LNToutput object or a string or vector of strings.
pattern	A character vector of keywords. Word boundaries before and after the keywords are honoured (see `word_boundaries`). Regular expression can be used.
case_insensitive	If FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.
unique_pattern	If TRUE, duplicated mentions of the same pattern are removed.
word_boundaries	If TRUE or "both", lookup is performed with word boundaries at beginning and end of the pattern (i.e., pattern "protest" will not identify "protesters" etc.). Additionally word boundaries can be either just in front of the pattern ("before") or after the pattern ("after"). FALSE searches without word boundaries.
cores	The number of CPU cores to use. Use `NULL` or `1` to turn off.
verbose	A logical flag indicating whether a status bar is printed to the screen.

Details

If an LNToutput object is provided, the function will look for the pattern in the headlines and articles. The returned object is a list of hits. If a regular expression is provided, the returned word will be the actual value from the text.

Value

A list of keyword hits.

Author(s)

Johannes Gruber

Examples

```
# Make LNToutput object from sample
LNToutput <- lnt_read(lnt_sample(copy = FALSE))

# Lookup keywords
LNToutput@meta$Keyword <- lnt_lookup(
  LNToutput,
  "statistical computing"
)

# Keep only articles which mention the keyword
LNToutput_stat <- LNToutput[!sapply(LNToutput@meta$Keyword, is.null)]

# Convert list of keywords to string
LNToutput@meta$Keyword <- sapply(LNToutput@meta$Keyword, toString)
```

`lnt_read` *Read in a LexisNexis file*

Description

Read a file from LexisNexis in a supported format and convert it to an object of class LNToutput. Supported formats are TXT, DOC, RTF and PDF files.

Usage

```
lnt_read(
  x,
  encoding = "UTF-8",
  extract_paragraphs = TRUE,
  convert_date = TRUE,
  start_keyword = "auto",
  end_keyword = "auto",
  length_keyword = "auto",
  author_keyword = "auto",
  exclude_lines = "^LOAD-DATE: |^UPDATE: |^GRAFIK: |^GRAPHIC: |^DATELINE: ",
  recursive = FALSE,
  file_type = c("txt", "rtf", "doc", "pdf", "docx", "zip"),
  verbose = TRUE,
  ...
)
```

Arguments

`x`	Name(s) of file(s) or one or multiple directories containing files from LexisNexis to be converted.
`encoding`	Encoding to be assumed for input files. Defaults to UTF-8 (the LexisNexis standard value).
`extract_paragraphs`	A logical flag indicating if the returned object will include a third data frame with paragraphs.
`convert_date`	A logical flag indicating if it should be tried to convert the date of each article into Date format. For non-standard dates provided by LexisNexis it might be safer to convert dates afterwards (see lnt_asDate).
`start_keyword`	Is used to indicate the beginning of an article. All articles should have the same number of Beginnings, ends and lengths (which indicate the last line of metadata). Use regex expression such as "\d+ of \d+ DOCUMENTS$" (which would catch e.g., the format "2 of 100 DOCUMENTS") or "auto" to try all common keywords. Keyword search is case sensitive.
`end_keyword`	Is used to indicate the end of an article. Works the same way as start_keyword. A common regex would be "^LANGUAGE: " which catches language in all caps at the beginning of the line (usually the last line of an article).

`length_keyword`	Is used to indicate the end of the metadata. Works the same way as start_keyword and end_keyword. A common regex would be "^LENGTH: " which catches length in all caps at the beginning of the line (usually the last line of the metadata).
`author_keyword`	A keyword to identify the author(s) in the metadata.
`exclude_lines`	Lines in which these keywords are found are excluded. Set to `character()` if you want to turn off this feature.
`recursive`	A logical flag indicating whether subdirectories are searched for more files.
`file_type`	File types/extensions to be included in search for files.
`verbose`	A logical flag indicating whether information should be printed to the screen.
`...`	Additional arguments passed on to lnt_asDate.

Details

The function can produce an LNToutput S4 object with two or three data.frame: meta, containing all meta information such as date, author and headline and articles, containing just the article ID and the text of the articles. When extract_paragraphs is set to TRUE, the output contains a third data.frame, similar to articles but with articles split into paragraphs.

When left to 'auto', the keywords will use the following defaults, which should be the standard keywords in all languages used by 'LexisNexis':

* `start_keyword` = `"\d+ of \d+ DOCUMENTS$| Dokument \d+ von \d+$| Document \d+ de \d+$"`.

* `end_keyword` = `"^LANGUAGE: |^SPRACHE: |^LANGUE: "`.

Value

An LNToutput S4 object consisting of 3 data.frames for metadata, articles and paragraphs.

Author(s)

Johannes B. Gruber

Examples

```
LNToutput <- lnt_read(lnt_sample(copy = FALSE))
meta.df <- LNToutput@meta
articles.df <- LNToutput@articles
paragraphs.df <- LNToutput@paragraphs
```

lnt_rename *Assign proper names to LexisNexis files*

Description

Give proper names to files downloaded from 'LexisNexis' based on search term and period retrieved from each file cover page. This information is not always delivered by LexisNexis though. If the information is not present in the file, new file names will be empty.

Usage

```
lnt_rename(
  x,
  encoding = "UTF-8",
  recursive = FALSE,
  report = TRUE,
  simulate = TRUE,
  verbose = FALSE
)
```

Arguments

`x`	Can be either a character vector of LexisNexis file name(s), folder name(s) or can be left blank (see example).
`encoding`	Encoding to be assumed for input files. Defaults to UTF-8 (the LexisNexis standard value).
`recursive`	A logical flag indicating whether subdirectories are searched for more files.
`report`	A logical flag indicating whether the function will return a report which files were renamed.
`simulate`	Should the renaming be simulated instead of actually done? This can help prevent accidental renaming of unrelated files which happen to be in the same directory as the files from 'LexisNexis'.
`verbose`	A logical flag indicating whether information should be printed to the screen.

Details

Warning: This will rename all supported files in a give folder.

Author(s)

Johannes B. Gruber

Examples

```
## Not run:
# Copy sample file to current wd
lnt_sample()

# Rename files in current wd and report back if successful

report.df <- lnt_rename(
  recursive = FALSE,
  report = TRUE
)


# Or provide file name(s)
my_files <- list.files(
  pattern = ".txt", full.names = TRUE,
  recursive = TRUE, ignore.case = TRUE
)
report.df <- lnt_rename(
  x = my_files,
  recursive = FALSE,
  report = TRUE
)

# Or provide folder name(s)
report.df <- lnt_rename(x = getwd())

report.df

## End(Not run)
```

`lnt_sample` *Provides a small sample TXT/DOCX file*

Description

Copies a small TXT sample file (as used by the old Nexis) or a DOCX (as used by Nexis Uni or Lexis Advance) to the current working directory and returns the location of this newly created file. The content of the file is made up or copied from Wikipedia since real articles from LexisNexis fall under copyright laws and can not be shared.

Usage

```
lnt_sample(
  format = "txt",
  overwrite = FALSE,
  verbose = TRUE,
  path = NULL,
  copy = TRUE
)
```

Arguments

`format`	Either "txt" to get the sample.TXT file or "docx" to get the format used by Nexis Uni.
`overwrite`	Should the sample file be overwritten if found in the current working directory?
`verbose`	Display warning message if file exists in current wd.
`path`	The destination path for the sample file (current working directory if `NULL`)
`copy`	Logical. Should the file be copied to path/working directory? If `FALSE`, the function only returns the location of the sample file.

Details

A small sample database to test the functions of LexisNexisTools

Author(s)

Johannes Gruber

Examples

```
## Not run:
  lnt_sample()

## End(Not run)
```

`lnt_similarity` *Check for highly similar articles.*

Description

Check for highly similar articles by comparing all articles published on the same date. This function implements two measures to test if articles are almost identical. The function textstat_simil, which compares the word similarity of two given texts; and a relative modification of the generalized Levenshtein (edit) distance implementation in stringdist. The relative distance is calculated by dividing the string distance by the number of characters in the longer article (resulting in a minimum of 0 if articles are exactly alike and 1 if strings are completely different). Using both methods cancels out the disadvantages of each method: the similarity measure is fast but does not take the word order into account. Two widely different texts could, therefore, be identified as the same, if they employ the exact same vocabulary for some reason. The generalized Levenshtein distance is more accurate but is very computationally demanding, especially if more than two texts are compared at once.

Usage

```
lnt_similarity(
  texts,
  dates,
  LNToutput,
  IDs = NULL,
  threshold = 0.99,
  rel_dist = TRUE,
  length_diff = Inf,
  nthread = getOption("sd_num_thread"),
  max_length = Inf,
  verbose = TRUE
)
```

Arguments

`texts`	Provide texts to check for similarity.
`dates`	Provide corresponding dates, same length as `text`.
`LNToutput`	Alternatively to providing texts and dates individually, you can provide an LNToutput object.
`IDs`	IDs of articles.
`threshold`	At which threshold of similarity is an article considered a duplicate. Note that lower threshold values will increase the time to calculate the relative difference (as more articles are considered).
`rel_dist`	Calculate the relative Levenshtein distance between two articles if set to TRUE (can take very long). The main difference between the similarity and distance value is that the distance takes word order into account while similarity employs the bag of words approach.
`length_diff`	Before calculating the relative distance between articles, the length of the articles in characters is calculated. If the difference surpasses this value, calculation is omitted and the distance will set to NA.
`nthread`	Maximum number of threads to use (see stringdist-parallelization).
`max_length`	If the article is too long, calculation of the relative distance can cause R to crash (see `https://github.com/markvanderloo/stringdist/issues/59`). To prevent this you can set a maximum length (longer articles will not be evaluated).
`verbose`	A logical flag indicating whether information should be printed to the screen.

Value

A data.table consisting of information about duplicated articles. Articles with a lower similarity than the threshold will be removed, while all relative distances are still in the returned object. Before you use the duplicated information to subset your dataset, you should, therefore, filter out results with a high relative distance (e.g. larger than 0.2).

Author(s)

Johannes B. Gruber

Examples

```
## Not run:
# Copy sample file to current wd
lnt_sample()

## End(Not run)

# Convert raw file to LNToutput object
LNToutput <- lnt_read(lnt_sample(copy = FALSE))

# Test similarity of articles
duplicates.df <- lnt_similarity(
  texts = LNToutput@articles$Article,
  dates = LNToutput@meta$Date,
  IDs = LNToutput@articles$ID
)

# Remove instances with a high relative distance
duplicates.df <- duplicates.df[duplicates.df$rel_dist < 0.2]

# Create three separate data.frames from cleaned LNToutput object
LNToutput <- LNToutput[!LNToutput@meta$ID %in%
  duplicates.df$ID_duplicate]
meta.df <- LNToutput@meta
articles.df <- LNToutput@articles
paragraphs.df <- LNToutput@paragraphs
```

Index