Package: biblioverlap (via r-universe)

September 16, 2024

Type Package

Title Document-Level Matching Between Bibliographic Datasets

Version 1.0.3.9000

Description Identifies and visualizes document overlap in any number of bibliographic datasets. This package implements the identification of overlapping documents through the exact match of a unique identifier (e.g. Digital Object Identifier - DOI) and, for records where the identifier is absent, through a score calculated from a set of fields commonly found in bibliographic datasets (Title, Source, Authors and Publication Year). Additionally, it provides functions to visualize the results of the document matching through a Venn diagram and/or UpSet plot, as well as a summary of the matching procedure.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

Imports cowplot, dplyr, ggplot2, ggVennDiagram, grDevices, grid, magrittr, Matrix, parallel, png, rlang, shiny, stringdist, UpSetR, uuid

Suggests DT, testthat

Depends R (>= 4.1)

URL `https://github.com/gavieira/biblioverlap`

BugReports `https://github.com/gavieira/biblioverlap/issues`

Roxygen list(markdown = TRUE)

Repository https://gavieira.r-universe.dev

RemoteUrl https://github.com/gavieira/biblioverlap

RemoteRef HEAD

RemoteSha adc31349ca93e3086be0235a650e9e5a6346512a

Contents

`biblioverApp` *Shiny App for the biblioverlap package*

Description

Shiny App for the biblioverlap package

Usage

```
biblioverApp(port = NULL, max_upload_size = 1000, launch.browser = TRUE)
```

Arguments

`port`	• port of the application
`max_upload_size`	• max upload size of documents (in MB) - Default 1000
`launch.browser`	• launch on browser - Default = TRUE

Value

opens a instance of the biblioverlap UI

Examples

```
#Running the ShinyApp
biblioverApp()
```

`biblioverlap` *Document-level matching of bibliographic datasets*

Description

This function identifies document overlap between bibliographic datasets and records it through the use of Universally Unique Identifiers (UUID).

Usage

```
biblioverlap(
  db_list,
  matching_fields = default_matching_fields,
  n_threads = 1,
  ti_penalty = 0.1,
  ti_max = 0.6,
  so_penalty = 0.1,
  so_max = 0.3,
  au_penalty = 0.1,
  au_max = 0.3,
  py_max = 0.3,
  score_cutoff = 1
)
```

Arguments

`db_list`	• named list of dataframes containing the sets of bibliographic data. Names are used to identify sets throughout the matching process.
`matching_fields`	• Five column names used in the matching. Should be universal across all datasets and provided as a named list with the following names: **DI** (unique identifier), **TI** (document title), **PY** (publication year), **SO** (publication source) and **AU** (Authors). Default values come from The Lens scholar field definition.
`n_threads`	• number of (logical) cores used in the matching procedures. Default: 1
`ti_penalty`	• penalty applied for each increment in Title's Levenshtein distance. Default: 0.1
`ti_max`	• max score value for Title. Default: 0.6
`so_penalty`	• penalty applied for each increment in Source's Levenshtein distance. Default: 0.1
`so_max`	• max score value for Source. Default: 0.3
`au_penalty`	• penalty applied for each increment in Author's Levenshtein distance. Default: 0.1
`au_max`	• max score value for Author. Default: 0.3
`py_max`	• max score value for Publication Year. Default: 0.3
`score_cutoff`	• minimum final score for a valid match between two documents. Default: 1

Details

In this procedure, any duplicates in the same dataset are removed. Then, Universally Unique Identifiers (UUID) are attributed to each record. If a match is found between two documents in a pairwise comparison, the UUID of the record from the first dataset is copied to the record on the second.

All preprocessing and modifications to the dataset are performed in a copy of the original data, which is used internally by the program. After all pairwise comparisons are completed, the UUID data is added as a new column in the original data.

Thus, the `db_list` returned by this function contains the same fields provided by the user plus the UUID column with the overlap information. This allows for further analysis using other fields (e.g. 'number of citations' or 'document type').

Value

a list object containing:

(i) `db_list`: a modified version of db_list where matching documents share the same UUID

(ii) `summary`: a summary of the results of the matching procedure

Note

In its internal data, the program will attempt to split the AU (Author) field to extract only the first author, for which it will calculate the Levenshtein distance.

It assumes that the AU field is ";" (semicolon) separated. Thus, in order to correctly perform the matching procedure to when another separator is being applied to this field, the user can either: (i) change the separator to semicolon; or (ii) create a new column containing only the first author.

Examples

```
#Example list of input dataframes
lapply(ufrj_bio_0122, head, n=1)

#List of columns for matching (identical to biblioverlap()'s defaults)
matching_cols <- list(DI = 'DOI',
                      TI = 'Title',
                      PY = 'Publication Year',
                      AU = 'Author/s',
                      SO = 'Source Title')

#Running document-level matching procedure (first two dataframes)
biblioverlap_results <- biblioverlap(ufrj_bio_0122[1:2], matching_fields = matching_cols)

#Taking a look at the matched db_list
lapply(biblioverlap_results$db_list, head, n=1)

#Taking a look at the matching results summary
biblioverlap_results$summary
```

`get_all_subset_matches` *Get all matches from a given subset of biblioverlap's results*

Description

Get all matches from a given subset of biblioverlap's results

Usage

```
get_all_subset_matches(subset_db_list, db_list)
```

Arguments

`subset_db_list`	• a subset of the results generated by `biblioverlap`
`db_list`	• the full set of results generated by `biblioverlap`

Value

- the subset data plus any other records outside the subset that have been matched to its documents

Examples

```
#Running document-level matching procedure for two datasets
biblioverlap_results <- biblioverlap(ufrj_bio_0122[1:2])$db_list

#Change the document type of one of the datasets from 'journal article' to
#'article' to emulate bibliographical source differences
biblioverlap_results[[2]][['Publication Type']] <- gsub('journal article',
'article',
biblioverlap_results[[2]][['Publication Type']])

#Generating venn diagram for the entire dataset
venn_plot(biblioverlap_results)

#Obtaining only the subset of records with publication type 'article'
biblioverlap_results_subset <- lapply(biblioverlap_results, function(db) {
db[db$'Publication Type' == "article", ] })

#Generating venn diagram for data subset
#Returns us how many documents categorized as 'article' are unique to a given
#dataset and how many find a match against other documents in the subset
#(i.e. that are also categorized as 'article', in this example)
venn_plot(biblioverlap_results_subset)

#Recovering missing matches due to bibliographical source differences
#in the subsetting process
subset_all_matches <- get_all_subset_matches(biblioverlap_results_subset,
biblioverlap_results)

#Generating venn diagram for data subset plus all its matches
#Returns us how many documents categorized as 'article' are unique to a given
#dataset and how many find a match against any other document
venn_plot(subset_all_matches)
```

`matching_summary_plot` *Plotting biblioverlap's matching summary*

Description

Plotting biblioverlap's matching summary

Usage

```
matching_summary_plot(
  matching_summary_df,
  add_logo = TRUE,
  text_size = 15,
  ...
)
```

Arguments

`matching_summary_df`	• summary of matching process generated by `biblioverlap`
`add_logo`	• boolean specifying whether to add package logo to plot or not. Default: TRUE
`text_size`	• text size of plot elements (like axes names and legend). Default: 15
`...`	• additional arguments passed down to `ggplot2::geom_text`

Value

a barplot summary of the matching results

Examples

```
#Running document-level matching procedure
biblioverlap_results <- biblioverlap(ufrj_bio_0122[1:2])

#Checking biblioverlap results (summary table)
biblioverlap_results$summary

#Plotting the matching summary
matching_summary_plot(biblioverlap_results$summary)
```

`merge_input_files` *Merge multiple input files from the same source*

Description

Merge multiple input files from the same source

Usage

```
merge_input_files(input_files, sep = ",", quote = "\"")
```

Arguments

`input_files`	• an array containing the path to all input files
`sep`	• field separator. Default: comma (',')
`quote`	• quote type used for character fields. Default: Double quotes ('"')

Details

It is fairly common to retrieve data from a single bibliographic database in small chunks. Thus, this function is designed to merge multiple files from the same source into a single file while also removing duplicate records.

Value

a single dataframe with all unique records from the input files

Examples

```
## Generating tempfiles
tempfile1 <- tempfile(fileext = ".csv")
tempfile2 <- tempfile(fileext = ".csv")
write.csv(ufrj_bio_0122$Biochemistry, file = tempfile1, row.names = FALSE)
write.csv(ufrj_bio_0122$Genetics, file = tempfile2, row.names = FALSE)

## Testing function
merged_files <- merge_input_files(c(tempfile1, tempfile2))
dim(merged_files)
head(merged_files)
```

`merge_results` *Merge biblioverlap's results into a single dataframe*

Description

Merge biblioverlap's results into a single dataframe

Usage

```
merge_results(db_list, filter = "none")
```

Arguments

`db_list`	• list of matched dataframes (with UUID column added by `biblioverlap`)
`filter`	• value determining whether to return all the data ('none'), distinct records only ('distinct') or matched records only ('matched'). Default: 'none'

Value

a single dataframe containing data from db_list, featuring an additional 'SET_NAME' column to indicate from which dataset each record came

Examples

```
#Running document-level matching procedure for two datasets
biblioverlap_results <- biblioverlap(ufrj_bio_0122[1:2])

#Obtaining the results as a single dataframe (all records)
all_data <- merge_results(biblioverlap_results$db_list)

#Checking number of total rows and overlapping documents are in the dataframe
nrow(all_data)
sum(duplicated(all_data$UUID))

#Obtaining only distinct records as a single dataframe
distinct_data <- merge_results(biblioverlap_results$db_list, filter = 'distinct')

#Checking number of total rows and overlapping documents are in the dataframe
nrow(distinct_data)
sum(duplicated(distinct_data$UUID))

#Obtaining only matched records as a single dataframe
matched_data <- merge_results(biblioverlap_results$db_list, filter = 'matched')

#Checking number of total rows and overlapping documents are in the dataframe
nrow(matched_data)
sum(duplicated(matched_data$UUID))
```

`ufrj_bio_0122` *UFRJ-affiliated documents from biological sciences disciplines (January 2022)*

Description

Data obtained from The Lens Scholarly Search in September 6, 2023.

The original data contained all documents from four major biological sciences fields published in the year 2022 by at least one author affiliated to the Universidade Federal do Rio de Janeiro (UFRJ). The data was then subsampled to documents published exclusively in January 2022 to reduce package size.

The biological disciplines featured in this dataset are Biochemistry, Genetics, Microbiology and Zoology.

Usage

```
ufrj_bio_0122
```

Format

`ufrj_bio_0122`**:**
A named list with 4 elements. Each element is a dataframe that contains the following fields:

Lens ID Unique identifier given to each record in The Lens database

DOI Digital Object Identifier

Title Document title

Publication Year Document publication year

Source Title Source (e.g. journal) where the document has been published

Author/s Document authors

Publication Type Type of the document (e.g. 'journal article', 'book chapter', etc...)

Citing Works Count Total number of citations received by document at the time of data recovery

Open Access Colour Type of open access (e.g. gold, bronze, green, etc...)

Source

https://www.lens.org

upset_plot *Plotting UpSet plot from biblioverlap results*

Description

Plotting UpSet plot from biblioverlap results

Usage

```
upset_plot(db_list, add_logo = TRUE, ...)
```

Arguments

`db_list`	• list of matched dataframes (with UUID column added by `biblioverlap`)
`add_logo`	• boolean specifying whether to add package logo to plot or not. Default: TRUE
`...`	• arguments to be passed down to `UpSetR::upset`

Value

a UpSet plot representation of document overlap between the input datasets

Examples

```
#Running document-level matching procedure
biblioverlap_results <- biblioverlap(ufrj_bio_0122[1:2])

#Checking biblioverlap results (db_list)
lapply(biblioverlap_results$db_list, head, n=1)

#Plotting the UpSet plot
upset_plot(biblioverlap_results$db_list)
```

venn_plot *Plotting Venn Diagram from biblioverlap results*

Description

Plotting Venn Diagram from biblioverlap results

Usage

```
venn_plot(db_list, add_logo = TRUE, ...)
```

Arguments

`db_list`	• list of matched dataframes (with UUID column added by `biblioverlap`)
`add_logo`	• boolean specifying whether to add package logo to plot or not. Default: TRUE
`...`	• Additional arguments that can be passed down to `ggVennDiagram::ggVennDiagram`

Value

a Venn Diagram representation of document overlap between the input datasets

Examples

```
#Running document-level matching procedure
biblioverlap_results <- biblioverlap(ufrj_bio_0122[1:2])

#Checking biblioverlap results (db_list)
lapply(biblioverlap_results$db_list, head, n=1)

#Plotting the Venn diagram
venn_plot(biblioverlap_results$db_list)
```

Index