Package 'rtransparency'

July 1, 2026

Type Package

Title Identifies Indicators of Transparency

Version 1.0.0

Description Use this package to identify indicators of transparency within the published literature. It can identify and extract text related to indicators of transparency from specifically formatted TXT files and from PMC XML files (i.e. XML files downloaded from the PubMed Central). It builds on the original 'rtransparent' tool of Serghiou et al. (2021) <doi:10.1371/journal.pbio.3001107>.

License GPL-3

Encoding UTF-8

LazyData true

URL https://github.com/choxos/rtransparency, https://choxos.github.io/rtransparency/

BugReports https://github.com/choxos/rtransparency/issues

Depends R (>= 3.5.0)

Imports dplyr, magrittr, purrr, readr, rlang, stringr, tibble, tidyselect, utf8, xml2

Suggests furrr, future, ggplot2, knitr, readxl, rmarkdown, testthat (>= 3.0.0)

Config/Needs/benchmark readxl

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Stylianos Serghiou [aut], Ahmad Sofi-Mahmudi [cre, aut] (ORCID: <https://orcid.org/0000-0001-6829-0823>)

Maintainer Ahmad Sofi-Mahmudi <a.sofimahmudi@gmail.com>

Repository CRAN

Date/Publication 2026-07-01 09:00:02 UTC

Contents

`rtransparency-package` *rtransparency: Identify indicators of transparency in the biomedical literature*

Description

Detects and extracts eight indicators of transparency (conflicts of interest, funding, protocol registration, novelty, replication, data sharing, code sharing, and disclosure of generative-AI use) from PubMed Central XML or plain-text articles. For each indicator it returns a boolean prediction and the statement that triggered it. This package builds on the original **rtransparent** tool of Serghiou et al. (2021).

Author(s)

Maintainer: Ahmad Sofi-Mahmudi <a.sofimahmudi@gmail.com> (ORCID)

Authors:

- Ahmad Sofi-Mahmudi <a.sofimahmudi@gmail.com> (ORCID)
- Stylianos Serghiou <stelios.serghiou@gmail.com>

See Also

Useful links:

- https://github.com/choxos/rtransparency
- https://choxos.github.io/rtransparency/
- Report bugs at https://github.com/choxos/rtransparency/issues

---

rt_accuracy *Detector accuracy estimates*

Description

Sensitivity and specificity estimates for each transparency detector, used by [rt_summary()] to correct an apparent prevalence for detector error (the Rogan-Gladen correction).

Usage

```
rt_accuracy
```

Format

A tibble with 7 rows and 5 columns:

variable Indicator column name, as returned by [rt_all_pmc()].

label Human-readable indicator name.

sensitivity Detector sensitivity (true-positive rate), 0-1.

specificity Detector specificity (true-negative rate), 0-1.

source Where the estimate comes from.

Details

For conflicts of interest, funding and protocol registration these are the published, importance-weighted validation values of Serghiou et al. (2021); the detectors for these indicators are essentially those validated in the paper. For data and code sharing the detector is implemented natively in this package (it no longer wraps 'oddpub'), so the package's reproducible benchmark and regression estimates are used instead (see 'inst/benchmark'). These data/code estimates are not an untouched external validation of the native detector; supply your own values to [rt_summary()] via its 'accuracy' argument when you have study-specific or externally validated estimates. Novelty's estimate comes from a maintainer-built hand-labeled gold set (see 'inst/benchmark/results_novelty_replication.md'). Replication's sensitivity comes from a 111-positive replication-enriched validation (see 'inst/benchmark/results_replication_e with the specificity from the 2023 1000-article sample. AI-use disclosure is not included (its prevalence is too low in unselected literature for a stable estimate), so [rt_summary()] reports it uncorrected.

Source

Serghiou S, Contopoulos-Ioannidis DG, Boyack KW, Riedel N, Wallach JD, Ioannidis JPA (2021). Assessment of transparency indicators across the biomedical literature: How open is open? *PLOS Biology* 19(3): e3001107. doi:10.1371/journal.pbio.3001107. Data and code values: this package's reproducible benchmark and regression estimates ('inst/benchmark/results_data_code.md').

See Also

[rt_summary()]

`rt_ai` *Identify disclosure of generative-AI use from a TXT file.*

Description

Detects whether an article discloses the use (or non-use) of generative AI or AI-assisted tools in preparing the manuscript, from a plain-text (typically PDF-derived) file. Unlike [rt_ai_pmc()] it applies **no publication-year gate**: a plain-text file carries no reliable publication date, so 'is_ai_pred' is always 'TRUE' or 'FALSE' (never 'NA'). AI-use disclosure became an expected practice only in 2023, so the caller is responsible for restricting analysis to articles from 2023 onward. Plain text also lacks the section structure the PMC detector uses to confine the scan to back matter, acknowledgments and declaration sections, so an article that uses AI purely as a research method is more likely to be flagged than under [rt_ai_pmc()].

Usage

```
rt_ai(filename)
```

Arguments

`filename`	The name of the TXT file as a string.

Value

A tibble with the filename, the PMID (if present in the file name), whether an AI-use disclosure was found (‘is_ai_pred‘) and the matched statement (‘ai_text‘).

See Also

[rt_ai_pmc()] for the PMC XML detector, which applies the 2023 publication-year gate.

Examples

```
# Write a short example article to a temporary text file.
filepath <- file.path(tempdir(), "PMID00000000-PMC0000000.txt")
writeLines(
  "The authors used ChatGPT to assist with drafting this manuscript.",
  filepath
)

# Identify and extract an AI-use disclosure.
rt_ai(filepath)
```

`rt_ai_pmc` *Identify disclosure of generative-AI use from a PMC XML file.*

Description

Detects whether an article discloses the use (or non-use) of generative AI or AI-assisted tools in preparing the manuscript, as required of articles since 2023. The indicator is only evaluated for articles published in 2023 or later; for earlier articles ‘is_ai_pred‘ is ‘NA‘.

Usage

```
rt_ai_pmc(filename, remove_ns = FALSE)
```

Arguments

`filename`	The filename of the PMC XML file to analyze.
`remove_ns`	TRUE if an XML namespace exists, else FALSE (default).

Value

A tibble with the article IDs, the publication ‘year‘, whether an AI disclosure was found (‘is_ai_pred‘, ‘NA‘ before 2023), the matched statement (‘ai_text‘) and ‘is_success‘.

Examples

```
filepath <- system.file(
  "extdata", "PMID32171256-PMC7071725.xml", package = "rtransparency"
)
rt_ai_pmc(filepath, remove_ns = TRUE)
```

`rt_all` *Identify and extract transparency statements from a TXT file.*

Description

Takes a TXT file and examines whether any statements of Conflicts of Interest (COI), Funding, Protocol Registration, Novelty or Replication exist. If any such statements are found, it also extracts the relevant text.

Usage

```
rt_all(filename)
```

Arguments

`filename`	The name of the TXT file as a string.

Value

A dataframe of results. It returns the PMID of the article (if this was included in the filename and preceded by "PMID"), whether each of the five indicators of transparency (COI, Funding, Registration, Novelty and Replication) was identified, the relevant text identified, and whether each labelling function identified relevant text or not. The labelling functions are returned to add flexibility in how this package is used; for example, future definitions of Registration may differ from the one we used. If a labelling function returns NA it means that it was not run.

Examples

```
# Write a short example article to a temporary text file.
filepath <- file.path(tempdir(), "PMID00000000-PMC0000000.txt")
writeLines(c(
  "To our knowledge, this is the first study of its kind.",
  "Conflicts of interest: none declared.",
  "This work was supported by the National Institutes of Health (R01-000000).",
  "The protocol was registered at ClinicalTrials.gov (NCT00000000).",
  "All data and code are available at https://github.com/example/repo.",
  "We independently replicated the original analysis."
), filepath)

# Identify and extract indicators of transparency.
results_table <- rt_all(filepath)
```

`rt_all_pmc` *Identify and extract all transparency indicators from a PMC XML.*

Description

Takes a PMC XML and returns relevant meta-data, as well as whether the article carries each of the eight transparency indicators: Conflicts of Interest (COI), Funding, Protocol Registration, Novelty, Replication, Data sharing, Code sharing and disclosure of generative-AI use. Where a statement is found, the relevant text is also extracted. This is the single-call entry point; it covers the same data and code detection as [rt_data_code_pmc()] and the same AI detection as [rt_ai_pmc()].

Usage

```
rt_all_pmc(filename, remove_ns = FALSE, all_meta = FALSE)
```

Arguments

`filename`	The name of the PMC XML as a string.
`remove_ns`	TRUE if an XML namespace exists, else FALSE (default).
`all_meta`	TRUE extracts all meta-data, FALSE extracts some (default).

Value

A dataframe of results. It returns the unique identifiers of the article, whether each indicator of transparency was identified (‘is_coi_pred‘, ‘is_fund_pred‘, ‘is_register_pred‘, ‘is_novelty_pred‘, ‘is_replication_pred‘, ‘is_open_data‘, ‘is_open_code‘ and the year-gated ‘is_ai_pred‘), the relevant text identified, whether it was identified through a dedicated XML tag (such variables include "pmc" in their name, e.g. “fund_pmc_source”) and whether each labelling function identified relevant text or not. The labeling functions are returned to add flexibility in how this package is used; for example, future definitions of Registration may differ from the one we used. If a labelling function returns NA it means that it was not run. ‘is_ai_pred‘ is ‘NA‘ for articles published before 2023 (see [rt_ai_pmc()]).

Examples

```
# Path to a bundled example PMC XML file.
filepath <- system.file(
  "extdata", "PMID32171256-PMC7071725.xml", package = "rtransparency"
)

# Identify and extract meta-data and indicators of transparency.
results_table <- rt_all_pmc(filepath, remove_ns = TRUE, all_meta = TRUE)
```

`rt_all_pmc_dir` *Identify transparency indicators across many PMC XML files.*

Description

A batch wrapper around [rt_all_pmc()] for corpus-scale runs over a directory (or an explicit vector) of PMC XML files. It isolates per-file failures so a single malformed file cannot abort the run, shows a progress bar, can resume an interrupted run, and can run in parallel when the **furrr** package is installed.

Usage

```
rt_all_pmc_dir(
  dir,
  pattern = "\\.xml$",
  recursive = FALSE,
  remove_ns = FALSE,
  all_meta = FALSE,
  output = NULL,
  parallel = FALSE,
  progress = TRUE,
  chunk_size = 200L
)
```

Arguments

`dir`	A directory containing PMC XML files, or a character vector of file paths.
`pattern`	A regular expression for file names, used only when ‘dir‘ is a single existing directory (default ‘"\.xml$"‘).
`recursive`	Whether to descend into subdirectories when ‘dir‘ is a directory (default ‘FALSE‘).
`remove_ns, all_meta`	Passed through to [rt_all_pmc()].
`output`	Optional path to a CSV file for incremental, resumable output (see Details). ‘NULL‘ (default) keeps results in memory only.
`parallel`	Whether to process files in parallel via **furrr** (default ‘FALSE‘).
`progress`	Whether to show a progress bar (default ‘TRUE‘).
`chunk_size`	Number of files per write/flush when ‘output‘ is set (default ‘200‘).

Details

When ‘output‘ is supplied, results are written to that CSV in chunks as the run proceeds. Re-running with the same ‘output‘ skips files already present in it and appends only the new results, so a long run can be resumed after an interruption. Each file is processed inside [tryCatch()]; a file that errors contributes a row with ‘is_success = FALSE‘ rather than stopping the run.

Parallelism uses **furrr**’s ‘future_map()‘ and honors whatever ‘future::plan()‘ is active (for example ‘future::plan("multisession")‘); with no plan it runs sequentially. Install **furrr** and **future** to use it.

Value

A [tibble][tibble::tibble] with one row per file, carrying the same columns as [rt_all_pmc()] (plus any rows read back from a pre-existing ‘output‘). Files that could not be processed have ‘is_success = FALSE‘.

See Also

[rt_all_pmc()] for a single file.

Examples

```
# Process every PMC XML in a directory (here, the bundled example file).
dir <- system.file("extdata", package = "rtransparency")
out <- tempfile(fileext = ".csv")
res <- rt_all_pmc_dir(dir, remove_ns = TRUE, output = out, parallel = FALSE)
```

`rt_coi` *Identify and extract Conflicts of Interest (COI) statements in TXT files.*

Description

Takes a TXT file and returns data related to the presence of a COI statement, including whether a COI statement exists. If a COI statement exists, it extracts it. Detection runs through the same text helpers as [rt_coi_pmc()], so a plain-text article is scored with the same logic as a PMC XML one (only the XML-structural routes, which need tags a TXT file does not have, are unavailable).

Usage

```
rt_coi(filename)
```

Arguments

`filename`	The name of the TXT file as a string.

Value

A dataframe of results. It returns the filename, PMID (if it was part of the file name), whether a COI was found and the text identified.

Examples

```
# Write a short example article to a temporary text file.
filepath <- file.path(tempdir(), "PMID00000000-PMC0000000.txt")
writeLines(c(
  "To our knowledge, this is the first study of its kind.",
  "Conflicts of interest: none declared.",
  "This work was supported by the National Institutes of Health (R01-000000).",
```

```
  "The protocol was registered at ClinicalTrials.gov (NCT00000000).",
  "All data and code are available at https://github.com/example/repo.",
  "We independently replicated the original analysis."
), filepath)

# Identify and extract the COI statement.
results_table <- rt_coi(filepath)
```

`rt_coi_pmc` *Identify and extract Conflicts of Interest (COI) statements in PMC XML files.*

Description

Takes a PMC XML file and returns data related to the presence of a COI statement, including whether a COI statement exists. If a COI statement exists, it extracts it.

Usage

```
rt_coi_pmc(filename, remove_ns = FALSE)
```

Arguments

`filename`	The name of the PMC XML as a string.
`remove_ns`	TRUE if an XML namespace exists, else FALSE (default).

Value

A dataframe of results. It returns unique article identifiers, whether this article was deemed relevant to COI, whether a COI was found, the text that suggested the presence of COI and the name of the function that identified this text. The functions are returned to add flexibility in how this package is used, such as future definitions of COI that may differ from the one we used.

Examples

```
# Path to a bundled example PMC XML file.
filepath <- system.file(
  "extdata", "PMID32171256-PMC7071725.xml", package = "rtransparency"
)

# Identify and extract meta-data and indicators of transparency.
results_table <- rt_coi_pmc(filepath, remove_ns = TRUE)
```

rt_data_code *Identify and extract Data and Code statements in TXT files.*

Description

Takes a TXT file and returns data related to the presence of Data and/or Code statements, including whether Data and/or Code statements exist. If such statements exist, it extracts them.

Usage

```
rt_data_code(filename)
```

Arguments

filename	The name of the TXT file as a string.

Value

A dataframe of results. It returns whether text suggesting the presence of data or code was found, and if so, what this text was.

Examples

```
# Write a short example article to a temporary text file.
filepath <- file.path(tempdir(), "PMID00000000-PMC0000000.txt")
writeLines(c(
  "To our knowledge, this is the first study of its kind.",
  "Conflicts of interest: none declared.",
  "This work was supported by the National Institutes of Health (R01-000000).",
  "The protocol was registered at ClinicalTrials.gov (NCT00000000).",
  "All data and code are available at https://github.com/example/repo.",
  "We independently replicated the original analysis."
), filepath)

# Identify and extract data and code availability.
results_table <- rt_data_code(filepath)
```

rt_data_code_pmc *Identify and extract Data and Code sharing from PMC XML files.*

Description

Takes a PMC XML file and returns data related to the presence of Data or Code, including whether Data or Code have been shared. If Data or Code exist, it will extract the relevant text for each. Detection is performed by the native detector (`.detect_data_code`); the package no longer depends on oddpub or `tokenizers`.

Usage

```
rt_data_code_pmc(filename, remove_ns = TRUE, specificity = "low")
```

Arguments

filename	The filename of the XML file to be analyzed as a string.
remove_ns	TRUE if an XML namespace exists, else FALSE (default).
specificity	Retained for backward compatibility; it no longer changes the result. The native detector extracts a fixed, broad set of article text (body paragraphs and titles, back matter, footnotes and supplements) and applies repository, accession and availability-statement patterns.

Value

A dataframe of results: the unique IDs of the article, whether data or code sharing was found (is_open_data, is_open_code), the statement text that triggered each detection (open_data_statements, open_code_statements) and the persistent identifiers and URLs of what was shared (open_data_links, open_code_links). The links are the DOIs (as doi.org URLs), repository URLs and database accessions (as identifiers.org prefix:accession) extracted from the statements, separated by " ; ".

Examples

```
# Path to PMC XML
filepath <- system.file(
  "extdata", "PMID32171256-PMC7071725.xml", package = "rtransparency"
)

# Identify and extract indicators of data and code sharing
results_table <- rt_data_code_pmc(filepath, remove_ns = TRUE)
```

---

rt_data_code_pmc_list *Identify and extract Data and Code sharing from a list of PMC XML files.*

Description

Takes a list of PMC XML files and returns data related to the presence of Data or Code, including whether Data or Code have been shared. If Data or Code exist, it will extract the relevant text for each.

Usage

```
rt_data_code_pmc_list(filenames, remove_ns = TRUE, specificity = "low")
```

Arguments

filenames	A list of the PMC XML filenames as strings.
remove_ns	TRUE if an XML namespace exists, else FALSE (default).
specificity	Retained for backward compatibility; see rt_data_code_pmc.

Value

A dataframe of results, one row per file.

Examples

```
# Paths to PMC XML files
filepath <- system.file(
  "extdata", "PMID32171256-PMC7071725.xml", package = "rtransparency"
)
filepaths <- list(filepath)

# Identify and extract indicators of data and code sharing
results_table <- rt_data_code_pmc_list(filepaths, remove_ns = TRUE)
```

---

rt_demo *Simulated transparency indicators for a corpus of articles*

Description

A small, simulated set of detector output, with one row per article, used to illustrate [rt_summary()], [rt_score()] and [rt_plot()]. The values are **simulated**, not real detector output: prevalences and their trends over time are chosen to resemble published findings (frequent conflict-of- interest and funding disclosure, less frequent protocol registration, low but rising data sharing, rare code sharing, and a recent, fast-rising disclosure of generative-AI use) so the illustrations are realistic.

Usage

```
rt_demo
```

Format

A tibble with 1200 rows and 11 columns:

pmid A made-up PubMed identifier (character).

year Publication year, 2010-2026.

type Article type (research-article, review-article, systematic-review).

is_coi_pred Conflict-of-interest statement detected.

is_fund_pred Funding statement detected.

is_register_pred Protocol registration detected.

is_open_data Data sharing detected.

is_open_code Code sharing detected.

is_novelty_pred Novelty claim detected.

is_replication_pred Replication component detected.

is_ai_pred Disclosure of generative-AI use detected. 'NA' before 2023, when the practice did not yet exist (see [rt_ai_pmc()]).

See Also

[rt_summary()], [rt_score()], [rt_plot()]

`rt_fund` *Identify and extract Funding statements in TXT files.*

Description

Takes a TXT file and returns data related to the presence of a Funding statement, including whether a Funding statement exists. If a Funding statement exists, it extracts it.

Usage

```
rt_fund(filename)
```

Arguments

`filename`	The name of the TXT file as a string.

Value

A dataframe of results. It returns the PMID (if this was part of the filename), whether a funding statement was found, what this statement was and the name of the function that identified this text. The functions are returned to add flexibility in how this package is used, such as future definitions of COI that may differ from the one we used.

Examples

```
# Write a short example article to a temporary text file.
filepath <- file.path(tempdir(), "PMID00000000-PMC0000000.txt")
writeLines(c(
  "To our knowledge, this is the first study of its kind.",
  "Conflicts of interest: none declared.",
  "This work was supported by the National Institutes of Health (R01-000000).",
  "The protocol was registered at ClinicalTrials.gov (NCT00000000).",
  "All data and code are available at https://github.com/example/repo.",
  "We independently replicated the original analysis."
), filepath)

# Identify and extract the funding statement.
```

```
results_table <- rt_fund(filepath)
```

`rt_fund_pmc` *Identify and extract Funding statements in PMC XML files.*

Description

Takes a PMC XML file and returns data related to the presence of a Funding statement, including whether a Funding statement exists. If a Funding statement exists, it extracts it.

Usage

```
rt_fund_pmc(filename, remove_ns = FALSE)
```

Arguments

`filename`	The name of the PMC XML as a string.
`remove_ns`	TRUE if an XML namespace exists, else FALSE (default).

Value

A dataframe of results. It returns all unique article identifiers, whether this article was deemed relevant to funding (e.g. was the word "fund" found within the text), whether a funding statement was found, whether a statement within the PMC tags dedicated to funding was found, the text identified, whether this text is explicit (i.e. whether it clearly indicated that funding was received) and whether each of the labeling functions identified the text or not. The functions are returned to add flexibility in how this package is used; for example, future definitions of Funding may differ from the one we used.

Examples

```
# Path to a bundled example PMC XML file.
filepath <- system.file(
  "extdata", "PMID32171256-PMC7071725.xml", package = "rtransparency"
)

# Identify and extract meta-data and indicators of transparency.
results_table <- rt_fund_pmc(filepath, remove_ns = TRUE)
```

rt_meta_pmc *Extract article metadata from a PMC XML file.*

Description

Reads a PMC XML file and returns its metadata as a one-row data frame: journal, publisher, article title, authors and affiliations, identifiers (PMID, PMCID, DOI), publication dates, and figure / table / reference counts.

Usage

```
rt_meta_pmc(filename, remove_ns = FALSE)
```

Arguments

filename	The path to the PMC XML file as a string.
remove_ns	TRUE if an XML namespace should be removed, else FALSE (default).

Value

A one-row tibble of metadata. The column ‘is_success‘ indicates whether the file was parsed successfully.

Examples

```
filepath <- system.file(
  "extdata", "PMID32171256-PMC7071725.xml", package = "rtransparency"
)
rt_meta_pmc(filepath, remove_ns = TRUE)
```

rt_novelty *Identify whether a study claims novelty in TXT files.*

Description

Takes a TXT file and returns data related to the presence of novelty claims, including whether a novelty claim exists. If a novelty claim exists, it extracts the relevant text. Novelty is defined as the study claiming to report something "for the first time."

Usage

```
rt_novelty(filename)
```

Arguments

filename	The name of the TXT file as a string.

Value

A tibble of results. It returns the filename, PMID (if it was part of the file name), whether a novelty claim was found, the text identified, and whether each pattern-matching function identified relevant text or not.

Examples

```
# Write a short example article to a temporary text file.
filepath <- file.path(tempdir(), "PMID00000000-PMC0000000.txt")
writeLines(c(
  "To our knowledge, this is the first study of its kind.",
  "Conflicts of interest: none declared.",
  "This work was supported by the National Institutes of Health (R01-000000).",
  "The protocol was registered at ClinicalTrials.gov (NCT00000000).",
  "All data and code are available at https://github.com/example/repo.",
  "We independently replicated the original analysis."
), filepath)

# Identify and extract novelty claims.
results_table <- rt_novelty(filepath)
```

---

rt_novelty_pmc *Identify and extract novelty claims in PMC XML files.*

Description

Takes a PMC XML file and returns data related to the presence of novelty claims, including whether such claims exist and the relevant text. Novelty is defined as the study claiming to report something "for the first time."

Usage

```
rt_novelty_pmc(filename, remove_ns = FALSE)
```

Arguments

filename	The name of the PMC XML as a string.
remove_ns	TRUE if an XML namespace exists, else FALSE (default).

Value

A tibble of results. It returns the unique identifiers of the article, whether a novelty claim was found, the relevant text and whether each pattern-matching function identified relevant text.

Examples

```
# Path to a bundled example PMC XML file.
filepath <- system.file(
  "extdata", "PMID32171256-PMC7071725.xml", package = "rtransparency"
)

# Identify and extract novelty claims.
results_table <- rt_novelty_pmc(filepath, remove_ns = TRUE)
```

`rt_plot` *Plot transparency indicators*

Description

Produces a ‘ggplot‘ of either the prevalence of each indicator (a bar chart) or the prevalence over time (a line chart). Requires the ‘ggplot2‘ package.

Usage

```
rt_plot(
  x,
  type = c("prevalence", "trend"),
  indicators = NULL,
  by = NULL,
  year = NULL,
  adjusted = FALSE,
  accuracy = NULL,
  conf_level = 0.95
)
```

Arguments

`x`	Either a data frame with one row per article (it is summarized with [rt_summary()]) or an existing [rt_summary()] result.
`type`	‘"prevalence"‘ for a bar chart of each indicator’s prevalence (the default), or ‘"trend"‘ for prevalence over time (requires ‘year‘).
`indicators, by`	Passed to [rt_summary()] when ‘x‘ is article-level data. ‘by‘ adds facets to the ‘"prevalence"‘ plot.
`year`	For ‘type = "trend"‘, the name of the column in ‘x‘ holding the (numeric) publication year.
`adjusted`	If ‘TRUE‘, plot the sensitivity/specificity-corrected prevalence instead of the apparent prevalence. Defaults to ‘FALSE‘.
`accuracy, conf_level`	Passed to [rt_summary()].

Value

A ‘ggplot‘ object.

See Also

[rt_summary()]

Examples

```
data(rt_demo)

if (requireNamespace("ggplot2", quietly = TRUE)) {
  rt_plot(rt_demo)                          # prevalence bar chart
  rt_plot(rt_demo, type = "trend", year = "year")
}
```

`rt_read_pdf` *Convert a PDF file to text.*

Description

Takes a path to a PDF file and returns its text content as a single character string, extracted with the poppler ‘pdftotext‘ utility (the same extractor the original ‘oddpub‘ package relied on, implemented here as a standard system call). Different extractors format text differently; the detectors in this package were tuned to the layout ‘pdftotext‘ produces. To analyze the result with the plain-text detectors, write it to a ‘.txt‘ file first (see Examples).

Usage

```
rt_read_pdf(filepath)
```

Arguments

`filepath`	The path to the PDF file as a string (must end in ‘.pdf‘).

Value

A character string with the extracted text.

Examples

```
## Not run:
# Path to a PDF file.
pdf_path <- system.file(
  "extdata", "PMID32171256-PMC7071725.pdf", package = "rtransparency"
)

# Extract the text, write it to a TXT file, then run the detectors.
```

```
article_txt <- rt_read_pdf(pdf_path)
writeLines(article_txt, "article.txt")
rt_coi("article.txt")

## End(Not run)
```

`rt_register` *Identify and extract Registration statements in TXT files.*

Description

Takes a TXT file and returns data related to the presence of a Registration statement, including whether a Registration statement exists. If a Registration statement exists, it extracts it.

Usage

```
rt_register(filename)
```

Arguments

`filename`	The name of the TXT file as a string.

Value

A dataframe of results. It returns the PMID (if this was part of the filename and preceded by PMID), whether a registration statement was found, the identified statement, whether the text was deemed relevant (e.g. contained the word registration), whether a Methods section was identified, whether an NCT number was identified, whether a registration was explicitly identified (defunct) and whether each labeling function identified a relevant text or not. The labeling functions are returned to add flexibility in how this package is used; for example, future definitions of Registration may differ from the one we used.

Examples

```
# Write a short example article to a temporary text file.
filepath <- file.path(tempdir(), "PMID00000000-PMC0000000.txt")
writeLines(c(
  "To our knowledge, this is the first study of its kind.",
  "Conflicts of interest: none declared.",
  "This work was supported by the National Institutes of Health (R01-000000).",
  "The protocol was registered at ClinicalTrials.gov (NCT00000000).",
  "All data and code are available at https://github.com/example/repo.",
  "We independently replicated the original analysis."
), filepath)

# Identify and extract the registration statement.
results_table <- rt_register(filepath)
```

`rt_register_pmc` *Identify and extract Conflicts of Interest statements in PMC XML files.*

Description

Takes a PMC XML file and returns data related to the presence of a Funding statement, including whether a Funding statement exists. If a Funding statement exists, it extracts it.

Usage

```
rt_register_pmc(filename, remove_ns = FALSE)
```

Arguments

`filename`	The name of the PMC XML as a string.
`remove_ns`	TRUE if an XML namespace exists, else FALSE (default).

Value

A dataframe of results. It returns the unique article identifiers, whether this article was deemed a research, review or systematic review, whether the text was deemed relevant to registration (e.g. contained the word registration), whether a Methods section was identified, whether an NCT number was identified, whether a registration was explicitly identified (defunct), whether a registration statement was found, what the registration statement was, whether it the registration was identified from the PMC XML (i.e. it was found within a dedicated registration tag) and whether each labeling function identified a relevant text or not. The labeling functions are returned to add flexibility in how this package is used; for example, future definitions of Registration may differ from the one we used.

Examples

```
# Path to a bundled example PMC XML file.
filepath <- system.file(
  "extdata", "PMID32171256-PMC7071725.xml", package = "rtransparency"
)

# Identify and extract meta-data and indicators of transparency.
results_table <- rt_register_pmc(filepath, remove_ns = TRUE)
```

`rt_replication` *Identify whether a study includes a replication component in TXT files.*

Description

Takes a TXT file and returns data related to the presence of a replication or validation component, including whether such a component exists. Replication is defined as the study independently confirming findings from a prior study in a new sample.

Usage

```
rt_replication(filename)
```

Arguments

`filename`	The name of the TXT file as a string.

Value

A tibble of results. It returns the filename, PMID (if it was part of the file name), whether a replication component was found, the text identified, and whether each pattern-matching function identified relevant text or not.

Examples

```
# Write a short example article to a temporary text file.
filepath <- file.path(tempdir(), "PMID00000000-PMC0000000.txt")
writeLines(c(
  "To our knowledge, this is the first study of its kind.",
  "Conflicts of interest: none declared.",
  "This work was supported by the National Institutes of Health (R01-000000).",
  "The protocol was registered at ClinicalTrials.gov (NCT00000000).",
  "All data and code are available at https://github.com/example/repo.",
  "We independently replicated the original analysis."
), filepath)

# Identify and extract replication components.
results_table <- rt_replication(filepath)
```

rt_replication_pmc *Identify and extract replication components in PMC XML files.*

Description

Takes a PMC XML file and returns data related to the presence of a replication or validation component, including whether such a component exists and the relevant text. Replication is defined as the study independently confirming findings from a prior study in a new sample.

Usage

```
rt_replication_pmc(filename, remove_ns = FALSE)
```

Arguments

filename	The name of the PMC XML as a string.
remove_ns	TRUE if an XML namespace exists, else FALSE (default).

Value

A tibble of results. It returns the unique identifiers of the article, whether a replication component was found, the relevant text and whether each pattern-matching function identified relevant text.

Examples

```
# Path to a bundled example PMC XML file.
filepath <- system.file(
  "extdata", "PMID32171256-PMC7071725.xml", package = "rtransparency"
)

# Identify and extract replication components.
results_table <- rt_replication_pmc(filepath, remove_ns = TRUE)
```

rt_score *Count the transparency indicators met by each article*

Description

Adds a column giving, for each article (row), how many of the transparency indicators were detected. This is the per-article transparency score used to describe how many practices an article adheres to.

Usage

```
rt_score(data, indicators = NULL, name = "n_indicators")
```

Arguments

data	A data frame with one row per article and indicator columns named as in [rt_all_pmc()].
indicators	Optional character vector of indicator columns to count. Defaults to the five openness practices present in ‘data‘ (conflicts of interest, funding, registration, data and code); novelty and replication are excluded unless requested explicitly, as they are not adherence practices.
name	Name of the count column to add (default ‘"n_indicators"‘).

Value

‘data‘ as a tibble with the integer count column added. Rows with no assessed indicators receive ‘NA‘ for the count. Tabulate it (for example with [table()] or ‘dplyr::count()‘) for the distribution of the number of practices met.

See Also

[rt_summary()]

Examples

```
data(rt_demo)
scored <- rt_score(rt_demo)
table(scored$n_indicators)
```

---

rt_summary *Summarize transparency indicators across a corpus of articles*

Description

Takes a data frame with one row per article (such as the output of [rt_all_pmc()] joined with [rt_data_code_pmc()], stacked over many articles) and returns the prevalence of each transparency indicator. For each indicator it reports the number of articles assessed, the number in which the indicator was detected, the apparent prevalence and its Wilson confidence interval and, optionally, a prevalence corrected for the detector’s sensitivity and specificity (the Rogan-Gladen estimator).

Usage

```
rt_summary(
  data,
  indicators = NULL,
  by = NULL,
  adjust = TRUE,
  accuracy = NULL,
  conf_level = 0.95
)
```

Arguments

data	A data frame with one row per article. Indicator columns must be logical or numeric 0/1 and named as in [rt_all_pmc()]: ‘is_coi_pred‘, ‘is_fund_pred‘, ‘is_register_pred‘, ‘is_open_data‘, ‘is_open_code‘, ‘is_novelty_pred‘, ‘is_replication_pred‘ and ‘is_ai_pred‘. ‘NA‘ marks an article that was not assessed for that indicator (for example ‘is_ai_pred‘ before 2023) and is excluded from its denominator. Other values are rejected rather than silently coerced.
indicators	Optional character vector of indicator columns to summarize. Defaults to every recognized indicator present in ‘data‘.
by	Optional name of a grouping column (for example a publication year, journal or article type); the summary is then computed within each group.
adjust	If ‘TRUE‘ (default), add a prevalence corrected for detector sensitivity and specificity using ‘accuracy‘. Indicators absent from ‘accuracy‘ receive ‘NA‘ corrected values.
accuracy	A data frame of detector accuracy with columns ‘variable‘, ‘sensitivity‘ and ‘specificity‘. Defaults to [rt_accuracy].
conf_level	Confidence level for the intervals (default ‘0.95‘).

Value

A tibble with one row per indicator (per group, if ‘by‘ is given): the grouping column (when ‘by‘ is used), ‘indicator‘, ‘label‘, ‘n_articles‘, ‘n_detected‘, ‘percent‘, ‘conf_low‘, ‘conf_high‘ and, when ‘adjust = TRUE‘, ‘adj_percent‘, ‘adj_low‘ and ‘adj_high‘. Percentages and interval bounds are on the 0-100 scale.

See Also

[rt_score()], [rt_plot()], [rt_accuracy]

Examples

```
data(rt_demo)
rt_summary(rt_demo)

# Apparent prevalence only, no accuracy correction
rt_summary(rt_demo, adjust = FALSE)

# By article type
rt_summary(rt_demo, by = "type")
```

Index