Package: regextable (via r-universe)

May 20, 2026

Title Pattern-Based Text Extraction and Standardization with Lookup Tables

Version 0.1.2

Description Extracts information from text using lookup tables of regular expressions. Each text entry is compared against patterns in a lookup table, returning the extracted substrings alongside optional metadata. Users can choose to extract all matching patterns (generating multiple rows per text entry) or limit extraction to the first match. This approach enables comprehensive, standardized pattern coverage when processing large or complex text datasets.

LazyData true

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports chk, dplyr, stringi, stringr, pbapply, stats

Suggests kableExtra, knitr, rmarkdown, quarto, spelling, testthat (>= 3.0.0), spacyr, tibble, googledrive, googlesheets4

VignetteBuilder quarto

URL https://github.com/judgelord/regextable, https://judgelord.github.io/regextable/

BugReports https://github.com/judgelord/regextable/issues

Config/testthat/edition 3

Depends R (>= 4.1)

Language en-US

Config/pak/sysreqs libicu-dev

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

Date/Publication 2026-04-20 19:03:11 UTC

RemoteUrl https://github.com/judgelord/regextable

RemoteRef HEAD

RemoteSha 035ef79284faf43743f3c9fa87c8bb763d299eb2

Contents

`clean_text` *Clean Text*

Description

Cleans a character vector by converting to lowercase, removing or replacing specific punctuation, normalizing commas, and squishing excess whitespace.

Usage

```
clean_text(text)
```

Arguments

`text`	Character vector to clean.

Value

A cleaned character vector.

Examples

```
clean_text(c("Hello  World!?", "This--is\tR.\nTesting: 1, 2, , 3;"))
```

cr2007_03_01 *cr2007_03_01 dataset*

Description

Sample text dataset used for demonstration of `regextable`.

Format

A tibble with 5 columns:

date Date of the record (YYYY-MM-DD)

speaker Speaker name in the text

header Header or title of the speech

url Original URL of the source text

url_txt Full text content from the source

Source

Generated for the `regextable` package.

extract *Extract Regex Pattern Matches from Text Data*

Description

Matches text against a table of regular expressions and returns extracted matches with optional metadata.

Usage

```
extract(
  data,
  regex_table,
  col_name = "text",
  pattern_col = "pattern",
  typo_table = NULL,
  typo_from_col = "typo",
  typo_to_col = "correction",
  date_col = NULL,
  date_start = NULL,
  date_end = NULL,
  data_return_cols = NULL,
  regex_return_cols = NULL,
```

```
  remove_acronyms = FALSE,
  do_clean_text = TRUE,
  unique_match = FALSE,
  use_ner = FALSE,
  ner_timing = "after",
  ner_entity_types = c("ORG"),
  verbose = TRUE,
  cl = NULL
)
```

Arguments

data	A data frame or character vector containing the text to search. If a character vector is provided, it is internally converted to a data frame and col_name is ignored.
regex_table	A data frame containing regular expression patterns and optional metadata columns.
col_name	Character string specifying the column in data that contains text to search. Default is "text".
pattern_col	Character string specifying the column in regex_table that contains regex patterns. Default is "pattern".
typo_table	Optional data frame with text replacements for corrections. Replacements are applied sequentially to the text using regex (with word boundaries) before pattern matching.
typo_from_col	Optional column in typo_table with text to replace. Default is "typo".
typo_to_col	Optional column in typo_table with replacement text. Default is "correction".
date_col	Optional column in data for date filtering. If provided, rows are filtered by date_start and date_end before pattern matching.
date_start	Optional start date (Date object or string like "YYYY-MM-DD") for filtering data when date_col is specified.
date_end	Optional end date (Date object or string like "YYYY-MM-DD") for filtering data when date_col is specified.
data_return_cols	Optional vector of column names to include from data. Default is NULL (only row_id is returned).
regex_return_cols	Optional vector of column names to include from regex_table. Default is NULL (no metadata columns added).
remove_acronyms	Logical; if TRUE, removes patterns consisting only of uppercase letters (2 or more characters) from regex_table.
do_clean_text	Logical; if TRUE, applies basic text cleaning to the input before matching.
unique_match	Logical; if TRUE, stops searching after the first match to find at most one match per row (evaluated in the order patterns appear in regex_table). If FALSE, returns all matches for all patterns.

use_ner — Logical; if TRUE, uses the 'spacyr' package to validate that matches are actual Named Entities (e.g., organizations). Note: spacyr must be initialized (e.g., via spacyr::spacy_initialize()) before calling this function.

ner_timing — Character string; either "after" or "before". If "after" (default), regex matches are found first, then validated with NER. If "before", NER extracts entities first, and regex searches only within those entities.

ner_entity_types — Character vector; the types of Named Entities to keep if use_ner is TRUE. Default is "ORG".

verbose — Logical; if TRUE, displays progress messages.

cl — A cluster object created by parallel::makeCluster(), or an integer to indicate number of child processes (integer values are ignored on Windows). Passed to pbapply::pblapply().

Details

Pattern matching is performed using R's regular expression engine and is case-insensitive by default. For each input row, the function checks patterns in regex_table and returns matches based on the unique_match parameter.

Value

A tibble with the following columns:

- row_id: Integer identifier corresponding to rows in the input data.
- Additional columns from data if data_return_cols is specified.
- Additional columns from regex_table if regex_return_cols is specified.
- pattern: The matched regular expression pattern(s).
- match: The extracted text from the data (original casing preserved).

Examples

```
# Create sample data
data <- data.frame(
  id = 1:3,
  text = c("I love apples", "Bananas are great", "Oranges and apples"),
  stringsAsFactors = FALSE
)

# Create regex patterns
patterns <- data.frame(
  pattern = c("apples", "bananas", "oranges"),
  category = c("fruit", "fruit", "fruit")
)

# Extract all matches
extract(data, patterns)

# Extract one match per row
extract(data, patterns, unique_match = TRUE)
```

`members` *members dataset*

Description

Lookup table of member names and metadata for regex matching.

Format

A tibble with 9 columns:

congress Congress number (numeric)

chamber Chamber (House/President/Senate)

bioname Full bio name of the member

pattern Regex pattern to match this member's name

icpsr Numeric ICPSR identifier

state_abbrev Two-letter state abbreviation

district_code District number (0 for President)

first_name First name of the member

last_name Last name of the member

Source

Generated for the `regextable` package.

Index