Package 'topics'

May 8, 2026

Type Package

Title Creating and Significance Testing Language Features for Visualisation

Version 0.70

Description
Implements differential language analysis with statistical tests and offers various language visualization techniques for n-grams and topics. It also supports the 'text' package. For more information, visit <https://r-topics.org/> and <https://www.r-text.org/>.

License GPL-3

URL https://r-topics.org/

BugReports https://github.com/theharmonylab/topics/issues

Encoding UTF-8

Archs x64

SystemRequirements Python (>= 3.6.0)

LazyData true

BuildVignettes true

Imports textmineR, quanteda, ggplot2, dplyr, ggwordcloud, tibble, methods, readr, stopwords, Matrix, stringr, stringi, rlang, tidyr, purrr, data.table, ggforce, patchwork, utils

RoxygenNote 7.3.3

Suggests text, rJava, mallet, glmnet, knitr, rmarkdown, testthat, covr, vdiffr, httr, arrow

VignetteBuilder knitr

Depends R (>= 4.00)

NeedsCompilation no

Author Leon Ackermann [aut] (ORCID: <https://orcid.org/0009-0008-8583-8748>), Zhuojun Gu [aut] (ORCID: <https://orcid.org/0009-0000-1610-4830>), Oscar Kjell [aut, cre] (ORCID: <https://orcid.org/0000-0002-2728-6278>)

Maintainer Oscar Kjell <oscar.kjell@psy.lu.se>

Repository CRAN

Date/Publication 2026-02-16 09:50:02 UTC

Contents

dep_wor_data . . . 2
topicsDtm . . . 3
topicsDtmEval . . . 5
topicsGrams . . . 6
topicsGridLegend . . . 8
topicsModel . . . 10
topicsPlot . . . 12
topicsPlotOverview . . . 16
topicsPredict . . . 16
topicsScatterLegend . . . 18
topicsTest . . . 20
topicsTutorialData . . . 22

Index **23**

`dep_wor_data` *Example data about mental health descirptions .*

Description

Example data about mental health descirptions .

Usage

```
dep_wor_data
```

Format

A data frame with 500 participants and 13 variables:

Depselect Words that respondents have selected from a pre-defined list

Worselect Words that respondents have selected from a pre-defined list

Depword Wrods where respondents describe their experience with depression in life

Worword Words where respondents describe their experience with depression in life

Depphrase phrases where respondents describe their experience with depression in life

Worphrase Phrases where respondents describe their experience with anxiety in life

Deptext Text where respondents describe their experience with depression in life

Wortext Text where respondents describe their experience with anxiety in life

Gender Respondents gender 0=male, 1=female

Age respondents age in years

PHQ9tot total score of the respondents PHQ-9

GAD7tot total score of the respondents GAD-7

MentalSickDays self-reported sick days due to mental health issues

MentalSickDaysYear self-reported sick days due to mental health issues in the last year

HealthCareMental self-reported health care visits due to mental health

SickDaysMonth self-reported sick days last month

SickDaysYear self-reported sick days last year

HealthCareVisits self-reported health care visits

Source

<https://osf.io/preprints/psyarxiv/p67db>

`topicsDtm` *Document Term Matrix*

Description

This function creates a document term matrix

Usage

```
topicsDtm(
  data,
  ngram_window = c(1, 3),
  stopwords = stopwords::stopwords("en", source = "snowball"),
  removalword = "",
  pmi_threshold = NULL,
  occurance_rate = 0,
  removal_mode = "percentage",
  removal_rate_most = 0,
  removal_rate_least = 0,
  shuffle = TRUE,
  lower = TRUE,
  remove_punctuation = TRUE,
  remove_numbers = TRUE,
  stem_lemma_function = NULL,
  verbose = FALSE,
  seed = 42L,
  threads = 1
)
```

Arguments

`data`	(list) A list containing the text data with each entry belonging to a unique id
`ngram_window`	(list) The minimum and maximum n-gram length, e.g., c(1,3)
`stopwords`	(stopwords) The stopwords to remove, e.g., stopwords::stopwords("en", source = "snowball")

`removalword` (string) Character vector of words to remove; e.g., "word1" or c("word1", "word2", ...)

`pmi_threshold` (integer; experimental) Pointwise Mutual Information (PMI) measures the association between terms by comparing their co-occurrence probability to their individual probabilities, highlighting term pairs that occur together more often than expected by chance; in this implementation, terms with average PMI below the specified threshold (pmi_threshold) are removed from the document-term matrix.

`occurance_rate` (numerical) The occurance rate (0-1) removes words that occur less then in (occurance_rate)*(number of documents). Example: If the training dataset has 1000 documents and the occurrence rate is set to 0.05, the code will remove terms that appear in less than 49 documents.

`removal_mode` (string) Mode of removal -> one of c("none", "frequency", "term", "percentage"). frequency removes all words under a certain frequency or over a certain frequency, as indicated by removal_rate_least and removal_rate_most. term removes an absolute number of terms that are most frequent and least frequent. percentage removes the number of terms indicated by removal_rate_least and removal_rate_most relative to the number of terms in the matrix

`removal_rate_most` (integer) The rate of most frequent words to be removed, functionality depends on removal_mode

`removal_rate_least` (integer) The rate of least frequent words to be removed, functionality depends on removal_mode

`shuffle` (boolean) Shuffle the data before analyses

`lower` (boolean) If TRUE, converts all text to lowercase before embedding.

`remove_punctuation` (boolean) If TRUE, replaces non-alphanumeric characters with spaces.

`remove_numbers` (boolean) If TRUE, replaces all numerical digits with spaces.

`stem_lemma_function` (function). A custom function for stemming or lemmatization.

`verbose` Logical. If TRUE, displays progress bars and status updates during text vectorization and the embedding process. Defaults to FALSE.

`seed` (integer) A seed to set for reproducibility

`threads` (integer) The number of threads to use; also called cpu in (CreateDtm).

Value

The document term matrix

Examples

```
# Create a Dtm and remove the terms that occur less than 4 times and more than 500 times.

dtm <- topicsDtm(data = dep_wor_data$Depphrase,
```

```
                removal_mode = "frequency",
                removal_rate_least = 4,
                removal_rate_most = 500)

# Create Dtm and remove the 1 least and 1 most frequent terms.
dtm <- topicsDtm(data = dep_wor_data$Depphrase,
                removal_mode = "term",
                removal_rate_least = 1,
                removal_rate_most = 1)

# Create Dtm and remove the 1% least frequent and 1% most frequent terms.
# The percentage values are scaled to values between 0 and 1.
dtm <- topicsDtm(data = dep_wor_data$Depphrase,
                removal_mode = "percentage",
                removal_rate_least = 0.01,
                removal_rate_most = 0.01)
```

`topicsDtmEval` *Summarize and Visualize your Document Term Matrix*

Description

This function creates a frequency table of your DTM and generates up to four plots for visualization

Usage

```
topicsDtmEval(dtm)
```

Arguments

`dtm`	(R_obj) The document term matrix -> output of topicsDtm function

Value

A named list containing:

dtm_summary A dataframe of terms and their frequencies.

frequency_plot A bar plot of all term frequencies with example terms.

frequency_plot_30_least A bar plot of the 30 least frequent terms (if numer of terms > 30).

frequency_plot_30_most A bar plot of the 30 most frequent terms (if numer of terms > 30).

histogram_of_frequencies A histogram of term frequencies (this is the same information as in the frequency_plot but presented differently).

topicsGrams *N-grams*

Description

Compute n-grams and per-document relative frequencies from text.

Usage

```
topicsGrams(
  data,
  ngram_window = c(1, 3),
  stopwords = stopwords::stopwords("en", source = "snowball"),
  removalword = "",
  pmi_threshold = NULL,
  occurance_rate = 0,
  removal_mode = "frequency",
  removal_rate_most = NULL,
  removal_rate_least = NULL,
  shuffle = TRUE,
  lower = TRUE,
  remove_punctuation = TRUE,
  remove_numbers = TRUE,
  stem_lemma_function = NULL,
  verbose = FALSE,
  seed = 42L,
  threads = 1,
  top_frequent = NULL,
  freq_per_user_format = c("auto", "wide", "long"),
  max_wide_cells = 5e+07
)
```

Arguments

data	A character vector of texts or a data.frame/tibble (first column used as text).
ngram_window	Integer vector specifying n-gram sizes. If length 1, only that n is used (e.g., 2). If length 2 and increasing, it is interpreted as an inclusive range (e.g., c(1, 3) -> 1,2,3). Otherwise it is treated as an explicit set (e.g., c(1, 3) means only 1 and 3).
stopwords	Character vector of stopwords to remove (e.g., stopwords::stopwords("en", source = "snowball")).
removalword	Character vector of words to remove from the raw text prior to tokenization.
pmi_threshold	Numeric PMI threshold used to filter multi-word n-grams. If NULL, PMI filtering is skipped.

`occurance_rate` Numeric in [0,1]. Terms are removed if they occur in fewer than `round(N_docs * occurance_rate) - 1` documents, where `N_docs` is the number of texts. Example: with 1000 documents and `occurance_rate = 0.05`, terms occurring in fewer than ~50 documents are removed.

`removal_mode` Character. Removal mode passed to `filter_ngrams()`. Common options include `"frequency"` and `"percentage"` (depending on your helper implementation).

`removal_rate_most` Numeric. Rate/threshold for removing the most frequent n-grams (interpreted by `filter_ngrams()` according to `removal_mode`).

`removal_rate_least` Numeric. Rate/threshold for removing the least frequent n-grams (interpreted by `filter_ngrams()` according to `removal_mode`).

`shuffle` Logical. If `TRUE`, texts are shuffled (seed-controlled) before processing. The original ids are preserved in output.

`lower` Logical. If `TRUE`, text is lowercased prior to tokenization.

`remove_punctuation` Logical. Passed to `quanteda::tokens(remove_punct = ...)`.

`remove_numbers` Logical. Passed to `quanteda::tokens(remove_numbers = ...)`.

`stem_lemma_function` Optional function applied to tokens (via `quanteda::tokens_apply()`). Should accept and return a character vector of tokens.

`verbose` Logical. If `TRUE`, prints simple progress messages (if implemented).

`seed` Integer. Random seed used when `shuffle = TRUE`.

`threads` Integer. Number of threads used by **quanteda** via `quanteda_options(threads = ...)`.

`top_frequent` Integer or `NULL`. If set, keeps only the `top_frequent` most frequent n-grams after filtering (recommended for large corpora). If `NULL`, keeps all retained n-grams.

`freq_per_user_format` Output format for `freq_per_user`:

- `"wide"` Wide document-by-ngram table (may be large).
- `"long"` Long sparse table with columns `id`, `ngram`, `rel_freq`.
- `"auto"` Uses wide format only if `N_docs * N_ngrams <= max_wide_cells`; otherwise returns long.

`max_wide_cells` Numeric. Safety limit for `"auto"`: if `N_docs * N_ngrams` exceeds this, `freq_per_user` is returned in long format to avoid dense allocation and excessive memory use.

Details

This function extracts n-grams using and returns: (1) a tibble of n-grams with frequency/prevalence and document frequency, (2) per-document relative frequencies for the retained n-grams (wide or long format), (3) filtering statistics, (4) a named list of settings (for reproducibility).

Value

A named list with:

settings A named list of settings used to generate the results (for reproducibility).

n_grams_pmi PMI information if available from your PMI helper, otherwise a message.

ngrams A tibble of retained n-grams and statistics (e.g., `freq`, `prevalence`, `num_docs`).

freq_per_user Per-document relative frequencies, either wide or long depending on `freq_per_user_format`.

stats A tibble summarizing how many n-grams were removed by each filtering step, by n-gram size.

`topicsGridLegend` *Plot a grid (matrix) legend (available for the text-package)*

Description

Plot a grid (matrix) legend (available for the text-package)

Usage

```
topicsGridLegend(
 bivariate_color_codes = c("#398CF9", "#60A1F7", "#5dc688", "#e07f6a", "#EAEAEA",
    "#40DD52", "#FF0000", "#EA7467", "#85DB8E"),
  filtered_test,
  cor_var = "",
  save_dir,
  figure_format = "svg",
  seed = 42,
  width = 10,
  height = 8,
  y_axes_1 = 2,
  legend_title,
  legend_title_size,
  titles_color,
  legend_x_axes_label,
  legend_y_axes_label,
  topic_data_all,
  legend_number_color,
  legend_number_size,
  title_font = "sans"
)
```

Arguments

bivariate_color_codes	A vector of color codes specifying the colors for the 3x3 grid legend. Default: c("#398CF9", "#60A1F7", "#5dc688", "#e07f6a", "#EAEAEA", "#40DD52", "#FF0000", "#EA7467", "#85DB8E").
filtered_test	A data frame containing the filtered topic data. Must include a 'color_categories' column.
cor_var	A string used to name the correlation variable, included in the file name of the saved plot. Default: "".
save_dir	Directory where the grid legend plot will be saved. Default: "./results".
figure_format	File format for the saved grid legend plot. Examples: "svg", "png", "pdf". Default: "svg".
seed	Seed for reproducibility, ensuring consistent plot generation. Default: 42.
width	Width of the saved grid legend in inches. Default: 10.
height	Height of the saved grid legend in inches. Default: 8.
y_axes_1	Specifies axis alignment for the grid legend. Options: 2 (2D grid with x and y axes) or 1 (1D legend for x-axis only). Default: 2.
legend_title	Title text for the grid legend. Must be provided by the user.
legend_title_size	Font size of the legend title text. Must be provided by the user.
titles_color	Color of the title and axis labels in the legend. Must be provided by the user.
legend_x_axes_label	Label for the x-axis of the grid legend. Must be provided by the user.
legend_y_axes_label	Label for the y-axis of the grid legend. Must be provided by the user.
topic_data_all	A data frame containing all topic data, including the 'color_categories' column used for counting topics.
legend_number_color	Color of the numeric annotations in the grid legend. Must be provided by the user.
legend_number_size	Font size of the numeric annotations in the grid legend. Must be provided by the user.' @return A legend plot saved that can be combined with the plot object.
title_font	Font family used for all non-word text elements in the plots (e.g., titles, axis labels, tick labels, legend text, annotations). Default: "sans". Examples: "serif", "mono", or a system-installed font family name.

`topicsModel` *Topic modelling*

Description

The function to create and train and an LDA model.

Usage

```
topicsModel(
  dtm,
  num_topics = 20,
  num_top_words = 10,
  num_iterations = 1000,
  seed = 42
)
```

Arguments

`dtm`	(R_obj) The document term matrix -> output of topicsDtm function
`num_topics`	(integer) The number of topics to be created
`num_top_words`	(integer) The number of top words to be displayed
`num_iterations`	(integer) The number of iterations to run the model
`seed`	(integer) A seed to set for reproducibility

Value

A named list containing the following elements:

name Description

instances Java object reference: A list of all documents used for topic modeling, in which each document is preprocessed (e.g., tokenized and vectorized). This object is part of the Mallet package's internal structure.

inferencer Java object reference: This is the topic inferencer, which allows the inference of topic distributions for new, unseen documents based on the trained model.

top_terms_mallet A data frame containing the top terms of each topic, showing which concepts each topic likely represents. The number of top terms shown here can be adjusted with the argument num_top_words.

top_terms A data frame containing the top terms of each topic, showing which concepts each topic likely represents. The number of top terms shown here can be adjusted with the argument num_top_words.

phi A matrix of the topic-word distribution: Each row represents a topic, and each column represents a word from the document term matrix. The values show the probability of a word given a topic P(wordltopic).

topic_docs A matrix of document-topic distribution: Each row represents a document, and each column represents a topic. The values show the probability of a topic given a document, P(topic|document).

frequencies A data frame of term frequencies. word = every word in the document term matrix, word.freq = the frequency of each word across all documents, doc.freq = the number of documents in which each word appears.

vocabulary A character vector of all unique terms in the document term matrix.

labels A list of topic labels. These short labels are the most representative term for each topic, making it easy to identify and understand them.

theta A data frame of document-topic probabilities: each row represents a document, and each column represents a topic. Similar to topic_docs, this shows the contribution of each topic to each document. Each row sums to 1, representing the document's composition of topics.

prevalence A numeric vector showing the overall prevalence (prominence) of each topic in the corpus. The prevalences are expressed as percentages relative to the other topics and add up to 100 Higher values indicate topics that are present in more documents.

coherence A numeric vector showing the coherence of each topic. Coherence scores indicate how semantically consistent and interpretable the topics are. Higher coherence generally indicates better-quality topics.

pred_model A list containing components of the predictive model, including phi (word-topic probability matrix), theta (document-topic probabilities matrix), alpha (Dirichlet prior of topics), gamma (hyperparameters of word-topic assignments), and data (sparse matrix representing the document term matrix.)

dtm_settings A list of settings used for preprocessing and building the document term matrix (dtm), including n-gram ranges, stopword removal, frequency thresholds, and random seed settings.

summary A summary data frame comprising of the topic numbers, labels, coherence scores, prevalence scores, and top terms.

Examples

```
# Create LDA Topic Model
save_dir_temp <- tempfile()
dtm <- topicsDtm(data = dep_wor_data$Depphrase)

model <- topicsModel(
dtm = dtm, # output of topicsDtm()
num_topics = 20,
num_top_words = 10,
num_iterations = 1000,
seed = 42)
```

topicsPlot *Plot word clouds*

Description

This function create word clouds and topic figures

Usage

```
topicsPlot(
  model = NULL,
  ngrams = NULL,
  test = NULL,
  p_alpha = 0.05,
  p_adjust_method = "none",
  ngrams_max = 30,
  ngram_select = "estimate",
  color_scheme = "default",
  word_font = "sans",
  title_font = "sans",
  overview_plot = TRUE,
  highlight_topic_words = NULL,
  scale_size = FALSE,
  plot_topics_idx = NULL,
  allowed_word_overlap = NULL,
  plot_n_most_prevalent_topics = NULL,
  save_dir = NULL,
  figure_format = "svg",
  width = 6,
  height = 5,
  max_size = 10,
  seed = 42,
  scatter_legend_dot_size = c(3, 8),
  scatter_legend_bg_dot_size = c(1, 3),
  scatter_legend_dots_alpha = 0.8,
  scatter_legend_bg_dots_alpha = 0.2,
  scatter_legend_n = c(1, 1, 1, 1, 0, 1, 1, 1, 1),
  scatter_legend_method = c("mean"),
  scatter_legend_specified_topics = NULL,
  scatter_legend_topic_n = FALSE,
  scatter_show_axis_values = TRUE,
  scatter_legend_circles = FALSE,
  scatter_legend_circles_radius = 0,
  scatter_legend_circles_num = 4,
  grid_legend_title = "legend_title",
  grid_legend_title_size = 5,
  grid_legend_title_color = "black",
```

```
  grid_legend_x_axes_label = "legend_x_axes_label",
  grid_legend_y_axes_label = "legend_y_axes_label",
  grid_legend_number_color = "white",
  grid_legend_number_size = 15
)
```

Arguments

model	(list) A trained topics model, e.g., from topicsModel(). Should be NULL if plotting ngrams.
ngrams	(list) The output from the the topicsGram() function. Should be NULL if plotting topics. Note 1: it is not possible to plot tags like <place>; so the < are replaced with underscore. Note 2: it is not possible to plot dash - alone, it is replaced with '_-_'.
test	(list) The test results; if plotting according to dimension(s) include the object from topicsTest() function.
p_alpha	(integer) The p-value threshold to use for significance testing.
p_adjust_method	(character) Method to adjust/correct p-values for multiple comparisons (default = "none"; see also "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr").
ngrams_max	(integer) The maximum number of n-grams to plot.
ngram_select	(character) Method to select ngrams_max, when using both ngram and test use "prevalence" or "estimate"; if you only use ngrams use "pmi", "frequency", or "prevalence".
color_scheme	(string 'default' or vector) The color scheme. For plots not including a test, the color_scheme should in clude 2 colours (1 gradient pair), such as: c("lightgray", "darkblue) For 1 dimensional plots of n-grams it should contain 4 colours (2 gradient pairs), such as: c("#EAEAEA", "darkred", # negative ngrams colors "#EAEAEA", "darkgreen" # positve ngrams colors) For 1-dimension plots of topics, it should contain 6 colours (3 gradient pairs), such as c("#EAEAEA", "darkred", # negative topics colors "#EAEAEA", "darkgray", # colours of topics not significantly associated "#EAEAEA", "darkgreen" # positve topics colors) For 2-dimensional plots of topics, the color scheme should contain 18 colours (9 gradient pairs), such as: c("lightgray", "#398CF9", # quadrant 1 (upper left corner) "lightgray", "#60A1F7", # quadrant 2 "lightgray", "#5dc688", # quadrant 3 (upper right corner) "lightgray", "#e07f6a", # quadrant 4

"lightgray", "darkgray", # quadrant 5 (middle square)
"lightgray", "#40DD52", # quadrant 6
"lightgray", "#FF0000", # quadrant 7 (bottom left corner)
"lightgray", "#EA7467", # quadrant 8
"lightgray", "#85DB8E") # quadrant 9 (bottom right corner).

`word_font` Font family used for the word text in the wordclouds (i.e., the plotted words only). Default: "sans". Examples: "serif", "mono", or a system-installed font family name (e.g., "Arial", "Times New Roman").

`title_font` Font family used for all non-word text elements in the plots (e.g., titles, axis labels, tick labels, legend text, annotations). Default: "sans". Examples: "serif", "mono", or a system-installed font family name.

`overview_plot` (boolean) Whether to produce an overview plot, including some of the topics and the ditribution (experimental).

`highlight_topic_words`
(str vector) Words to highlight in topics (e.g., negative words). Format: highlight_topic_words = c("not", "never"). The default value is NULL.

`scale_size` (logical) Whether to scale the size of the words.

`plot_topics_idx`
(vector) The index or indices of the topics to plot (look in the model-object for the indices). They can, for example, be c(1, 3:5) to plot topic t_1, t_3, t_4 and t_5) (optional).

`allowed_word_overlap`
(numeric) A filter function determining the maximum number of identical words in the topics to be plotted. This filter removes topics within each "color group" and also include removing topics from the distribution and grid legends; (Note that the adjustment for multiple comparison is taking place before these are removed; i.e., the adjusted p-values are not affected by this filter).

`plot_n_most_prevalent_topics`
(numeric) Plots the n most prevalent topics in a given model.

`save_dir` (string) The directory to save the plots.

`figure_format` (string) Set the figure format, e.g., ".svg", or ".png".

`width` (integer) The width of the topic (units = "in").

`height` (integer) The width of the topic (units = "in").

`max_size` (integer) The maximum size of the words.

`seed` (integer) The seed to set for reproducibility.

`scatter_legend_dot_size`
(integer) The size of dots in the scatter legend. If set to "prevalence", the size will change accordingly.

`scatter_legend_bg_dot_size`
(integer) The size of background dots in the scatter legend.

`scatter_legend_dots_alpha`
(numeric) The transparency alphe level of the dots.

`scatter_legend_bg_dots_alpha`
(numeric) The transparency alphe level of the background dots.

scatter_legend_n
: (numeric or vector) A vector determining the number of dots to emphasize in each quadrant of the scatter legend. For example: c(1,1,1,1,0,1,1,1,1) result in one dot in each quadrant except for the middle quadrant.

scatter_legend_method
: (string) The method to filter topics to be emphasized in the scatter legend; either "mean", "max_x", or "max_y".

scatter_legend_specified_topics
: (vector) Specify which topic(s) to emphasize in the scatter legend.

scatter_legend_topic_n
: (boolean) If TRUE, the topic numbers are shown in the scatter legend.

scatter_show_axis_values
: (boolean) If TRUE, the estimate values are shown on the distribution plot axes.

scatter_legend_circles
: Plot concentric circles for the scatter legend

scatter_legend_circles_radius
: Radius of first concentric circle

scatter_legend_circles_num
: Number of Concentric circles For example, c("t_1", "t_2"). If set, scatter_legend_method will have no effect.

grid_legend_title
: Title of the grid topic plot.

grid_legend_title_size
: Title size of the grid topic plot.

grid_legend_title_color
: Legend title color of the grid topic plot.

grid_legend_x_axes_label
: x-axis label of the grid topic plot.

grid_legend_y_axes_label
: y-axis label of the grid topic plot.

grid_legend_number_color
: Text color in the legend boxes of the grid topic plot.

grid_legend_number_size
: Text size in the legend boxes.

Value

The function provides a list of topic plots (if there are any significant topics), a legend plot, and a plot showing the topic distribution. If save_dir is specified, it saves all plots in this directory. If you want to show all plots irrespective of the topics' significance, set p_alpha = 1.

topicsPlotOverview *Combine topics and distribution legend (experimental)*

Description

Combine topics and distribution legend (experimental)

Usage

```
topicsPlotOverview(plot_list, overview_plot_type, title_font = "sans")
```

Arguments

plot_list	(list) Output from the topicsPlot function.
overview_plot_type	(character) Number of dimensions to plot (1 or 2).
title_font	Font family used for all non-word text elements in the plots (e.g., titles, axis labels, tick labels, legend text, annotations). Default: "sans". Examples: "serif", "mono", or a system-installed font family name.

Value

An overview plot including topics and distribution legend

topicsPredict *topicsPredict, topicsPreds, topicsAssess and topicsClassify*

Description

The function to predict the topics of a new document with the trained model.

Usage

```
topicsPredict(
  model,
  data,
  num_iterations = 200,
  sampling_interval = 10,
  burn_in = 10,
  seed = 42,
  create_new_dtm = FALSE
)

topicsAssess(
  model,
```

```
  data,
  num_iterations = 200,
  sampling_interval = 10,
  burn_in = 10,
  seed = 42,
  create_new_dtm = FALSE
)

topicsClassify(
  model,
  data,
  num_iterations = 200,
  sampling_interval = 10,
  burn_in = 10,
  seed = 42,
  create_new_dtm = FALSE
)

topicsPreds(
  model,
  data,
  num_iterations = 200,
  sampling_interval = 10,
  burn_in = 10,
  seed = 42,
  create_new_dtm = FALSE
)
```

Arguments

model	(list) The trained model.
data	(tibble) The text variable for which you want to infer the topic distribution. This can be the same data as used to create the dtm or new data.
num_iterations	(integer) The number of iterations to run the model.
sampling_interval	The number of iterations between consecutive samples collected. during the Gibbs Sampling process. This technique, known as thinning, helps reduce the correlation between consecutive samples and improves the quality of the final estimates by ensuring they are more independent. Purpose: By specifying a sampling_interval, you avoid collecting highly correlated samples, which can lead to more robust and accurate topic distributions. Example: If sampling_interval = 10, the algorithm collects a sample every 10 iterations (e.g., at iteration 10, 20, 30, etc.). Typical Values: Default: 10; Range: 5 to 50 (depending on the complexity and size of the data).
burn_in	The number of initial iterations discarded during the Gibbs Sampling process. These early iterations may not be representative of the final sampling distribution because the model is still stabilizing. Purpose: The burn_in period allows

	the model to converge to a more stable state before collecting samples, improving the quality of the inferred topic distributions. Example: If burn_in = 50, the first 50 iterations of the Gibbs Sampling process are discarded, and sampling begins afterward. Typical Values: Default: 50 to 100 Range: 10 to 1000 (larger datasets or more complex models may require a longer burn-in period).
seed	(integer) A seed to set for reproducibility.
create_new_dtm	(boolean) If applying the model on new data (not used in training), it can help to make a new dtm. Currently this is experimental, and using the textmineR::CreateDtm() function rather than the topicsDtm() function, which has more functions.

Value

A tibble of the predictions: The rows represent the documents, and the columns represent the topics. The values in the cells indicate the proportion of each topic within the corresponding document.

Examples

```
# Predict topics for new data with the trained model

dtm <- topicsDtm(
data = dep_wor_data$Depphrase)

model <- topicsModel(dtm = dtm, # output of topicsDtm()
                     num_topics = 20,
                     num_top_words = 10,
                     num_iterations = 1000,
                     seed = 42)

preds <- topicsPredict(model = model, # output of topicsModel()
                     data = dep_wor_data$Depphrase)
```

---

topicsScatterLegend *Plot a distribution plot (available for the text-package)*

Description

Plot a distribution plot (available for the text-package)

Usage

```
topicsScatterLegend(
  bivariate_color_codes,
  filtered_test,
  num_popout = 1,
  way_popout_topics = "mean",
  user_spec_topics = NULL,
  allow_topic_num_legend = FALSE,
```

```
  scatter_show_axis_values = TRUE,
  y_axes_1 = 2,
  cor_var = "",
  label_x_name = "x",
  label_y_name = "y",
  save_dir,
  figure_format = "svg",
  scatter_popout_dot_size = c(1, 5),
  scatter_bg_dot_size = c(1, 5),
  scatter_legend_dots_alpha = 0.8,
  scatter_legend_bg_dots_alpha = 0.2,
  scatter_legend_circles = FALSE,
  scatter_legend_circles_radius = 0,
  scatter_legend_circles_num = 4,
  width = 10,
  height = 8,
  seed = 42,
  title_font = "sans"
)
```

Arguments

bivariate_color_codes	A vector of color codes specifying colors for different categories in the scatter plot. Default: c("#398CF9", "#60A1F7", "#5dc688", "#e07f6a", "#EAEAEA", "#40DD52", "#FF0000", "#EA7467", "#85DB8E").
filtered_test	A data frame containing the input data for the scatter plot. Must include columns like ‘color_categories‘ and other variables used in the function.
num_popout	The number of topics to "pop out" in each category. Default: 1. Can be a single integer (applies to all categories) or a vector for specific categories.
way_popout_topics	The method for selecting pop-out topics. Options: "mean", "max_y", or "max_x". Default: "mean".
user_spec_topics	A vector of user-specified topics to highlight in the scatter plot. Default: NULL.
allow_topic_num_legend	Logical; if TRUE, displays topic numbers in the legend. Default: FALSE.
scatter_show_axis_values	Show values on the axises.
y_axes_1	Specifies axis alignment for the scatter legend. Options: 1 (x-axis) or 2 (y-axis). Default: 2.
cor_var	A string used for naming the correlation variable in labels or file names. Default: "".
label_x_name	Label for the x-axis in the scatter plot. Default: "x".
label_y_name	Label for the y-axis in the scatter plot. Default: "y".
save_dir	Directory where the scatter legend plot will be saved. Default: "./results".

figure_format
: File format for the saved scatter plot. Examples: "svg", "png", "pdf". Default: "svg".

scatter_popout_dot_size
: Size of the dots for pop-out topics in the scatter legend. Set to "prevalence" for dot size changing based on topic prevalence. Default: 15.

scatter_bg_dot_size
: Size of the dots for background topics in the scatter legend. Default: 9.

scatter_legend_dots_alpha
: The transparency of the dots

scatter_legend_bg_dots_alpha
: The transparency of the dots

scatter_legend_circles
: Plot concentric circles for the scatter legend

scatter_legend_circles_radius
: Radius of first concentric circle

scatter_legend_circles_num
: Number of Concentric circles

width
: Width of the saved scatter plot in inches. Default: 10.

height
: Height of the saved scatter plot in inches. Default: 8.

seed
: Seed for reproducibility, ensuring consistent plot generation. Default: 42.

title_font
: Font family used for all non-word text elements in the plots (e.g., titles, axis labels, tick labels, legend text, annotations). Default: "sans". Examples: "serif", "mono", or a system-installed font family name.

`topicsTest` *Test topics or n-grams*

Description

Statistically test topics or n-grams in relation to one or two other variables using regression or t-test.

Usage

```
topicsTest(
  data,
  model = NULL,
  preds = NULL,
  ngrams = NULL,
  x_variable = NULL,
  y_variable = NULL,
  controls = c(),
  test_method = "default",
  p_adjust_method = "fdr",
  complete_cases = FALSE,
  seed = 42
)
```

Arguments

data	(tibble) The tibble containing the variables to be tested.
model	(list) A trained model LDA-model from the topicsModel() function.
preds	(tibble) The predictions from the topicsPred() function.
ngrams	(list) Output of the n-gram function.
x_variable	(string) The x variable name to be predicted, and to be plotted (only needed for regression or correlation).
y_variable	(string) The y variable name to be predicted, and to be plotted (only needed for regression or correlation).
controls	(vector) The control variables (not supported yet).
test_method	(string) The test method to use. "default" checks if x_variable and y_variable only contain 0s and 1s, for which it applies logistic regression; otherwise it applies linear regression. Alternatively, if only specifying x_variable, the user may manually specify either "linear_regression" or "logistic_regression".
p_adjust_method	(character) Method to adjust/correct p-values for multiple comparisons (default = "fdr"; see also "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "none").
complete_cases	If TRUE, it will only use complete cases for x_variable, y_variable, controls, and preds.
seed	(integer) The seed to set for reproducibility

Value

A list of the test results, test method, and prediction variable.

Examples

```
# Test the topic document distribution in respect to a variable

dtm <- topicsDtm(
  data = dep_wor_data$Depphrase)

model <- topicsModel(
  dtm = dtm, # output of topicsDtm()
  num_topics = 20,
  num_top_words = 10,
  num_iterations = 1000,
  seed = 42)

preds <- topicsPreds(
 model = model, # output of topicsModel()
 data = dep_wor_data$Depphrase)

test <- topicsTest(
  model = model, # output of topicsModel()
  data=dep_wor_data,
```

```
  preds = preds, # output of topicsPreds()
  test_method = "linear_regression",
  x_variable = "Age")
```

`topicsTutorialData` *Download and Prepare Tutorial Data*

Description

Downloads the Big5 Essays dataset from Hugging Face, filters by word count, and returns a sample of a specified size.

Usage

```
topicsTutorialData(
  sample_size = 1000,
  min_word_count = 250,
  max_word_count = 750,
  seed = 42,
  verbose = TRUE
)
```

Arguments

`sample_size`	(integer) The number of rows to return. Default is 1000.
`min_word_count`	(integer) Minimum words per essay. Default is 250.
`max_word_count`	(integer) Maximum words per essay. Default is 750.
`seed`	(integer) Seed for reproducibility. Default is 42.
`verbose`	(boolean) Whether to print status messages. Default is TRUE.

Value

A processed tibble ready for topic modeling.

Index

∗ **datasets**
dep_wor_data, 2

dep_wor_data, 2

topicsAssess *(*topicsPredict*)*, 16
topicsClassify *(*topicsPredict*)*, 16
topicsDtm, 3
topicsDtmEval, 5
topicsGrams, 6
topicsGridLegend, 8
topicsModel, 10
topicsPlot, 12
topicsPlotOverview, 16
topicsPredict, 16
topicsPreds *(*topicsPredict*)*, 16
topicsScatterLegend, 18
topicsTest, 20
topicsTutorialData, 22