

Package ‘poldis’

September 18, 2026

Type Package

Title Analyse Political Texts

Version 0.2.0

Maintainer Henrique Sposito <henrique.sposito@graduateinstitute.ch>

Description Wrangle and annotate different types of political texts.
It also introduces Urgency Analysis,
a new method for the analysis of urgency in political texts.

URL <http://henriquesposito.com/poldis/>

BugReports <https://github.com/henriquesposito/poldis/issues>

License MIT + file LICENSE

Imports dplyr, stringr, purrr, stringi, quanteda, spacyr, textstem,
tidyr, stringdist

Suggests rmarkdown, testthat, tesseract, quanteda.textstats, keyATM,
messydates, pdftools, fmsb, ggplot2, tm, cli, knitr, ggthemes,
modelsummary, scales

Encoding UTF-8

LazyData True

Depends R (>= 3.5.0)

Config/roxygen2/version 8.0.0

VignetteBuilder knitr

NeedsCompilation no

Author Henrique Sposito [cre, aut, ctb],
James Hollway [ctb],
Jael Tan [ctb]

Repository CRAN

Date/Publication 2026-09-18 13:40:02 UTC

Contents

annotate_text	2
extract_context	3
extract_first_sentence	3
extract_locations	4
extract_match	5
extract_names	5
extract_text_similarities	6
gather_related_terms	7
gather_topics	8
get_urgency	8
read_pdf	9
select_priorities	10
split_text	11
US_inaugural_addresses_1993_2025	11
Index	12

annotate_text	<i>Annotate text with NLP</i>
---------------	-------------------------------

Description

This function relies on ‘{spacy}’ NLP parsing to annotate texts.

Usage

```
annotate_text(v, level = "words")
```

Arguments

v	Text vector
level	At which level would you like to parse the text? Options include "words" or "sentences". Defaults to "words".

Value

A data frame with syntax information by words or sentences in text.

Examples

```
#annotate_text(US_inaugural_addresses_1993_2025$text[2])
```

extract_context	<i>Extract context for string matches</i>
-----------------	---

Description

A function for getting string matches and the context in which they occur.

Usage

```
extract_context(match, v, level = "sentences", n = 1)
```

Arguments

match	Character string to be matched. For multiple strings, please use " " as a separator.
v	Text vector or annotated data frame.
level	At which text level do you want matches to be returned? Defaults to "sentences". Options are sentences, words, and paragraph.
n	Number of sentences or words matched before and after string match. Defaults to 1. That is, one word or one sentence before, and after, string match. For paragraphs, n is always set to one.

Value

A list of string matches and their context.

Examples

```
extract_context(match = " war ",
  v = US_inaugural_addresses_1993_2025$text[1],
  level = "sentences", n = 2)
```

extract_first_sentence	<i>Extract first sentence from text</i>
------------------------	---

Description

A lot of information is contained in the first sentence of a text. In political texts, for example, dates and locations are often contained in the first sentence of the text.

Usage

```
extract_first_sentence(v)
```

Arguments

v Text vector.

Value

A list of the first sentences in text.

Examples

```
extract_first_sentence("This is the first sentence. This is the second sentence.")
```

extract_locations	<i>Extract locations from strings</i>
-------------------	---------------------------------------

Description

Extract locations from strings

Usage

```
extract_locations(v)
```

Arguments

v Text vector.

Details

The function relies on geographical entity detection from NLP models.

Value

A data frame of locations and the number of times they appear.

Examples

```
#extract_locations(c("This is the United States", "This is Sao Paulo",  
# "I was in Rio de Janeiro and Sao Paulo, then back to the United States"))
```

extract_match	<i>Extract text matches</i>
---------------	-----------------------------

Description

Get texts in which certain "matches" occur.

Usage

```
extract_match(v, match, invert = FALSE, ignore.case = TRUE)
```

Arguments

v	Text vector or annotated data frame.
match	A regex match for a word(s) or expression. For multiple words, please use " " to divide them.
invert	Do you want texts without certain matches to be returned? By default FALSE.
ignore.case	Should case be ignored? By default, TRUE.

Value

A list the same length as text variable.

Examples

```
extract_match(c("This function was created on the 29 September 2021",  
"Today is October 12, 2021"), "October")
```

extract_names	<i>Extract a list of possible names of individuals in texts</i>
---------------	---

Description

Extract a list of possible names of individuals in texts

Usage

```
extract_names(v)
```

Arguments

v	A text vector.
---	----------------

Details

The function relies on named entity recognition from NLP models.

Value

A data frame of individual names and the number of times they appear.

Examples

```
#extract_names("This package was created by Jael, James, and I.")
```

```
extract_text_similarities
```

Extract similarities and differences in texts/segments

Description

Extract similarities and differences in texts/segments

Usage

```
extract_text_similarities(v, comparison = "similarities", method)
```

Arguments

v	Text vector or annotated data frame.
comparison	How would you like to compare texts? Options are "similarities", for comparing similarities, or "differences", for comparing differences. Defaults to "similarities".
method	A method for checking similarities or differences between texts. For similarities, defaults to "correlation" method. Other methods for similarities include "cosine", "jaccard", "ejaccard", "dice", "edice", "simple matching", and "hamann". For differences, defaults to "euclidean". Other methods for differences include "manhattan", "maximum", "canberra", and "minkowski". For more information on each of these methods and what are the implications in selecting a method, please see <code>?quanteda.textstats::textstat_simil()</code> .

Value

A matrix of similarity scores between texts.

Examples

```
#extract_text_similarities(US_inaugural_addresses_1993_2025$text)
```

gather_related_terms *Gather terms related to subjects*

Description

Gather terms related to subjects

Usage

```
gather_related_terms(v, dictionary)
```

Arguments

v	Text vector or annotated data frame.
dictionary	The dictionary of 20 major political topics from the Comparative Agendas Project (Jones et al., 2023) is used by default. Users can also declare a custom dictionary as a vector or a list. If users declare a vector, each element is treated as a independent topic. If users declare a list of subjects and related terms, function understands names as topic and words as terms.

Details

This function relies on keyword assisted topic models implemented in the ‘{keyATM}’ package to find related words based on the topics provided and texts in which they appear.

Value

A list of related terms to each of the topics declared in dictionary.

References

Eshima S, Imai K, and Sasaki T. 2024. “Keyword-Assisted Topic Models.” *American Journal of Political Science*, 68(2): 730-750. doi:10.1111/ajps.12779

Examples

```
#gather_related_terms(US_inaugural_addresses_1993_2025$text, dictionary = "CAP")
#gather_related_terms(US_inaugural_addresses_1993_2025$text,
#                      dictionary = list("military" = c("military", "gun", "war"),
#                                         "development" = c("development", "interest rate", "banks")))
#
```

gather_topics	<i>Gather topic from political discourses</i>
---------------	---

Description

Gather topic from political discourses

Usage

```
gather_topics(v, dictionary = "CAP")
```

Arguments

v	Text vector or annotated data frame. If missing, opens the political topics codebook.
dictionary	The dictionary of 20 major political topics from the Comparative Agendas Project (Jones et al., 2023) is used by default. Users can also declare a custom dictionary as a vector or a list. If users declare a vector, each element is treated as a independent topic. If users declare a list of subjects and related terms, function understands names as topic and words as terms. For more information on how the CAP topics were adapted, please run ‘gather_topics()’ to access the political topics codebook.

Value

A list of topics present in each text separated by comma.

Examples

```
summary(gather_topics(US_inaugural_addresses_1993_2025$text))
```

get_urgency	<i>Urgency Analysis</i>
-------------	-------------------------

Description

Urgency Analysis

Usage

```
get_urgency(v, summarise = "sum")
```

Arguments

v	Text vector or annotated data frame.
summarise	How to handle multiple matches for the same dictionary in the same text observation? By default, multiple matches are added together and their "sum" per text observation is returned. Users can, instead, choose the "mean", which returns the average score per dictionary per text observation, the "max" or the "min", when there are multiple matches.

Details

Urgency in political discourses is an expression of how necessary and/or how soon an action should be undertaken or completed. This is measured along four dictionaries, two related to necessity (e.g., degree of intensity and of commitment) and two related to timing (e.g., frequency and timing of action). The dictionaries include terms for each of these dimensions of urgency. Their scores were developed through an online survey that took place in June 2026. The survey results were recorded as counts of the number of participants who selected an urgency-related word as more urgent than its pair. To analyze the survey results, we employed Bradley-Terry models for paired comparisons. We use raw log odds coefficients to calculate urgency. Urgency scores for each disctionary are returned. The aggregated urgency scores is calculated by adding these scores and then divided by the coefficient for the word "must", to ensure interpretability (i.e., an aggregated score of 1 in equivalent to "we must do"). Users are welcome to change how urgency scores are aggregated and transform these scores into probabilities or odds ratios as necessary.

Value

A scored data frame for each dimension of urgency.

Examples

```
get_urgency(US_inaugural_addresses_1993_2025$text)
#a = get_urgency(select_priorities(US_inaugural_addresses_1993_2025$text))
#summary(get_urgency(US_inaugural_addresses_1993_2025$text))
#plot(get_urgency(US_inaugural_addresses_1993_2025$text))
```

read_pdf	<i>Read text from PDFs</i>
----------	----------------------------

Description

Read text from PDFs

Usage

```
read_pdf(path)
```

Arguments

path The path to a PDF file or a folder containing multiple PDFs.

Value

A list of texts.

select_priorities	<i>Select future priorities from political discourses</i>
-------------------	---

Description

Political priorities are statements in which actors express their intent or commitment to take political action in the future.

Usage

```
select_priorities(v, na.rm = TRUE)
```

Arguments

v Text vector or annotated data frame.

na.rm Would you like political statements that do not contain a political action to be removed? By default, TRUE.

Value

A data frame with syntax information by sentences and a variable identifying which of these sentences are priorities.

Examples

```
#select_priorities(US_inaugural_addresses_1993_2025$text)
```

split_text	<i>Split texts</i>
------------	--------------------

Description

Split texts into structured lists of lists according to a split sign.

Usage

```
split_text(v, splitsign = "\\.")
```

Arguments

v	Text vector or annotated data frame.
splitsign	Where do you want to split? By default sentences ("."). This can also be words, signals or other markers you want. For special characters, please use escape sign before (i.e. "\\").

Value

A list of lists the same length as vector.

Examples

```
split_text("This is the first sentence. This is the second sentence.")
```

US_inaugural_addresses_1993_2025

Inaugural speeches from United States presidents from 1993 to 2025

Description

A dataset containing nine inaugural addresses since 1993 for five different United States presidents: Clinton, Bush, Obama, Trump, and Biden.

Usage

```
data(US_inaugural_addresses_1993_2025)
```

Format

A data frame with 9 rows and 3 variables: the president, the date, and the full text of the inaugural address.

Source

The dataset was gathered from the American Presidency Project website (<https://www.presidency.ucsb.edu/>).

Index

* **datasets**

US_inaugural_addresses_1993_2025,
[11](#)

annotate_text, [2](#)

extract_context, [3](#)

extract_first_sentence, [3](#)

extract_locations, [4](#)

extract_match, [5](#)

extract_names, [5](#)

extract_text_similarities, [6](#)

gather_related_terms, [7](#)

gather_topics, [8](#)

get_urgency, [8](#)

read_pdf, [9](#)

select_priorities, [10](#)

split_text, [11](#)

US_inaugural_addresses_1993_2025, [11](#)