

Package ‘LSX’

April 19, 2021

Type Package

Title Model for Semisupervised Text Analysis Based on Word Embeddings

Date 2021-04-16

Version 0.9.9

Description A word embeddings-based semisupervised model for document scaling Watanabe (2020) <doi:10.1080/19312458.2020.1832976>. LSS allows users to analyze large and complex corpora on arbitrary dimensions with seed words exploiting efficiency of word embeddings (SVD, Glove). It can generate word vectors on a users-provided corpus or incorporate a pre-trained word vectors.

License GPL-3

LazyData TRUE

Encoding UTF-8

Depends methods, R (>= 3.5.0)

Imports quanteda (>= 2.0), quanteda.textstats, stringi, digest, Matrix, RSpectra, irlba, rsvd, rsparse, proxyC, stats, ggplot2, ggrepel, reshape2, locfit

Suggests testthat

RoxygenNote 7.1.1

BugReports <https://github.com/koheiw/LSX/issues>

NeedsCompilation no

Author Kohei Watanabe [aut, cre, cph]

Maintainer Kohei Watanabe <watanabe.kohei@gmail.com>

Repository CRAN

Date/Publication 2021-04-19 09:30:02 UTC

R topics documented:

as.seedwords	2
cohesion	2
data_dictionary_ideology	3

data_dictionary_sentiment	3
data_textmodel_lss_russianprotests	3
diagnosys	4
seedwords	4
smooth_lss	5
textmodel_lss	5
textplot_simil	8
textplot_terms	8
textstat_context	9

Index 12

as.seedwords *Convenient function to convert a list to seed words*

Description

Convenient function to convert a list to seed words

Usage

```
as.seedwords(x, upper = 1, lower = 2)
```

Arguments

x	a list of characters vectors or a dictionary object
upper	numeric index or key for seed words for higher scores
lower	numeric index or key for seed words for lower scores

Value

named numeric vector for seed words with polarity scores

cohesion *Computes cohesion of components of latent semantic analysis*

Description

Computes cohesion of components of latent semantic analysis

Usage

```
cohesion(object, bandwidth = 10)
```

Arguments

object	a fitted <code>textmodel_lss</code>
bandwidth	size of window for smoothing

`data_dictionary_ideology`*Seed words for analysis of left-right political ideology*

Description

Seed words for analysis of left-right political ideology

Examples

```
as.seedwords(data_dictionary_ideology)
```

`data_dictionary_sentiment`*Seed words for analysis of positive-negative sentiment*

Description

Seed words for analysis of positive-negative sentiment

References

Turney, P. D., & Littman, M. L. (2003). Measuring Praise and Criticism: Inference of Semantic Orientation from Association. *ACM Trans. Inf. Syst.*, 21(4), 315–346. doi: [10.1145/944012.944013](https://doi.org/10.1145/944012.944013)

Examples

```
as.seedwords(data_dictionary_sentiment)
```

`data_textmodel_lss_russianprotests`*A fitted LSS model on street protest in Russia*

Description

This model was trained on a Russian media corpus (newspapers, TV transcripts and newswires) to analyze framing of street protests. The scale is protests as "freedom of expression" (high) vs "social disorder" (low). Although some slots are missing in this object (because the model was imported from the original Python implementation), it allows you to scale texts using `predict`.

References

Lankina, Tomila, and Kohei Watanabe. "‘Russian Spring’ or ‘Spring Betrayal’? The Media as a Mirror of Putin’s Evolving Strategy in Ukraine." *Europe-Asia Studies* 69, no. 10 (2017): 1526–56. doi: [10.1080/09668136.2017.1397603](https://doi.org/10.1080/09668136.2017.1397603).

diagnosys

Identify noisy documents in a corpus

Description

Identify noisy documents in a corpus

Usage

```
diagnosys(x, ...)
```

Arguments

x	character or <code>corpus</code> object whose texts will be diagnosed
...	extra arguments passed to tokens

seedwords

Seed words for Latent Semantic Analysis

Description

Seed words for Latent Semantic Analysis

Usage

```
seedwords(type)
```

Arguments

type	type of seed words currently only for sentiment (<code>sentiment</code>) or political ideology (<code>ideology</code>).
------	---

References

Turney, P. D., & Littman, M. L. (2003). Measuring Praise and Criticism: Inference of Semantic Orientation from Association. *ACM Trans. Inf. Syst.*, 21(4), 315–346. doi: [10.1145/944012.944013](https://doi.org/10.1145/944012.944013)

Examples

```
seedwords('sentiment')
```

`smooth_lss`*Smooth predicted LSS scores by local polynomial regression*

Description

Smooth predicted LSS scores by local polynomial regression

Usage

```
smooth_lss(  
  x,  
  lss_var = "fit",  
  date_var = "date",  
  span = 0.1,  
  from = NULL,  
  to = NULL,  
  engine = c("loess", "locfit"),  
  ...  
)
```

Arguments

<code>x</code>	a data.frame containing LSS scores and dates
<code>lss_var</code>	the name of the column for LSS scores
<code>date_var</code>	the name of the columns for dates
<code>span</code>	determines the level of smoothing.
<code>from</code>	start of the time period
<code>to</code>	end of the time period
<code>engine</code>	specifies the function to smooth LSS scores: <code>loess()</code> or <code>locfit()</code> . The latter should be used when $n > 10000$.
<code>...</code>	extra arguments passed to <code>loess()</code> or <code>lp()</code>

`textmodel_lss`*A word embeddings-based semisupervised model for document scaling*

Description

A word embeddings-based semisupervised model for document scaling

Usage

```

textmodel_1ss(x, ...)

## S3 method for class 'dfm'
textmodel_1ss(
  x,
  seeds,
  terms = NULL,
  k = 300,
  slice = NULL,
  weight = "count",
  cache = FALSE,
  simil_method = "cosine",
  engine = c("RSpectra", "irlba", "rsvd"),
  include_data = FALSE,
  verbose = FALSE,
  ...
)

## S3 method for class 'fcm'
textmodel_1ss(
  x,
  seeds,
  terms = NULL,
  w = 50,
  weight = "count",
  cache = FALSE,
  simil_method = "cosine",
  engine = c("rsparse"),
  verbose = FALSE,
  ...
)

```

Arguments

x	a dfm or fcm created by <code>quanteda::dfm()</code> or <code>quanteda::fcm()</code>
...	additional argument passed to the SVD engine
seeds	a character vector, named numeric vector or dictionary that contains seed words.
terms	words weighted as model terms. All the features of <code>quanteda::dfm()</code> or <code>quanteda::fcm()</code> will be used if not specified.
k	the number of singular values requested to the SVD engine. Only used when x is a dfm.
slice	a number or indices of the components of word vectors used to compute similarity; <code>slice < k</code> to truncate word vectors; useful for diagnosis and simulation.
weight	weighting scheme passed to <code>quanteda::dfm_weight()</code> . Ignored when engine is "rsparse".

cache	if TRUE, save result of SVD for next execution with identical x and settings. Use the <code>base::options(lss_cache_dir)</code> to change the location cache files to be save.
simil_method	specifies method to compute similarity between features. The value is passed to <code>quanteda.textstats::textstat_simil()</code> , "cosine" is used otherwise.
engine	choose SVD engine between <code>RSpectra::svds()</code> , <code>irlba::irlba()</code> , and <code>rsparse::GloVe()</code> .
include_data	if TRUE, fitted model include the dfm supplied as x.
verbose	show messages if TRUE.
w	the size of word vectors. Only used when x is a fcm

References

- Watanabe, Kohei. 2020. "Latent Semantic Scaling: A Semisupervised Text Analysis Technique for New Domains and Languages", *Communication Methods and Measures*. doi: [10.1080/19312458.2020.1832976](https://doi.org/10.1080/19312458.2020.1832976).
- Watanabe, Kohei. 2017. "Measuring News Bias: Russia's Official News Agency ITAR-TASS' Coverage of the Ukraine Crisis" *European Journal of Communication*. doi: [10.1177/0267323117695735](https://doi.org/10.1177/0267323117695735).

Examples

```
library("quanteda")
con <- url("https://bit.ly/2GZwLcN", "rb")
corp <- readRDS(con)
close(con)
toks <- corpus_reshape(corp, "sentences") %>%
  tokens(remove_punct = TRUE) %>%
  tokens_remove(stopwords("en")) %>%
  tokens_select("^[[\p{L}]+$", valuetype = "regex", padding = TRUE)
dfmt <- dfm(tok) %>%
  dfm_trim(min_termfreq = 10)

seed <- as.seedwords(data_dictionary_sentiment)

# SVD
lss_svd <- textmodel_lss(dfmt, seed)
summary(lss_svd)

# sentiment model on economy
eco <- head(char_keyness(tok, 'econom*'), 500)
svd_eco <- textmodel_lss(dfmt, seed, terms = eco)

# sentiment model on politics
pol <- head(char_keyness(tok, 'politi*'), 500)
svd_pol <- textmodel_lss(dfmt, seed, terms = pol)

# GloVe
fcmt <- fcm(tok, context = "window", count = "weighted", weights = 1 / (1:5), tri = TRUE)
lss_glov <- textmodel_lss(fcmt, seed)
summary(lss_glov)
```

textplot_simil	<i>Plot similarity between seed words</i>
----------------	---

Description

Plot similarity between seed words

Usage

```
textplot_simil(x, group = FALSE)
```

Arguments

x	fitted textmodel_1ss object
group	if TRUE group seed words by seed patterns and show average similarity

textplot_terms	<i>Plot polarity scores of words</i>
----------------	--------------------------------------

Description

Plot polarity scores of words

Usage

```
textplot_terms(x, highlighted = NULL)
```

Arguments

x	fitted textmodel_1ss object
highlighted	quanteda::pattern to specify words to highlight

textstat_context	<i>Identify context words using user-provided patterns</i>
------------------	--

Description

Identify context words using user-provided patterns

Usage

```
textstat_context(  
  x,  
  pattern,  
  valuetype = c("glob", "regex", "fixed"),  
  case_insensitive = TRUE,  
  window = 10,  
  min_count = 10,  
  remove_pattern = TRUE,  
  n = 1,  
  skip = 0,  
  ...  
)
```

```
char_context(  
  x,  
  pattern,  
  valuetype = c("glob", "regex", "fixed"),  
  case_insensitive = TRUE,  
  window = 10,  
  min_count = 10,  
  remove_pattern = TRUE,  
  p = 0.001,  
  n = 1,  
  skip = 0  
)
```

```
char_keyness(  
  x,  
  pattern,  
  valuetype = c("glob", "regex", "fixed"),  
  case_insensitive = TRUE,  
  window = 10,  
  min_count = 10,  
  remove_pattern = TRUE,  
  p = 0.001,  
  n = 1,  
  skip = 0  
)
```

Arguments

<code>x</code>	a tokens object created by <code>quanteda::tokens()</code> .
<code>pattern</code>	<code>quanteda::pattern()</code> to specify target words.
<code>valuetype</code>	the type of pattern matching: "glob" for "glob"-style wildcard expressions; "regex" for regular expressions; or "fixed" for exact matching. See <code>quanteda::valuetype()</code> for details.
<code>case_insensitive</code>	if TRUE, ignore case when matching.
<code>window</code>	size of window for collocation analysis.
<code>min_count</code>	minimum frequency of words within the window to be considered as collocations.
<code>remove_pattern</code>	if TRUE, keywords do not contain target words.
<code>n</code>	integer vector specifying the number of elements to be concatenated in each ngram. Each element of this vector will define a n in the n -gram(s) that are produced.
<code>skip</code>	integer vector specifying the adjacency skip size for tokens forming the ngrams, default is 0 for only immediately neighbouring words. For skipgrams, skip can be a vector of integers, as the "classic" approach to forming skip-grams is to set $skip = k$ where k is the distance for which k or fewer skips are used to construct the n -gram. Thus a "4-skip- n -gram" defined as $skip = 0:4$ produces results that include 4 skips, 3 skips, 2 skips, 1 skip, and 0 skips (where 0 skips are typical n -grams formed from adjacent words). See Guthrie et al (2006).
<code>...</code>	additional arguments passed to <code>textstat_keyness()</code> .
<code>p</code>	threshold for statistical significance of collocations.

See Also

`tokens_select()` and `textstat_keyness()`

Examples

```
#' @examples

require(quanteda)
con <- url("https://bit.ly/2GZwLcN", "rb")
corp <- readRDS(con)
close(con)
corp <- corpus_reshape(corp, 'sentences')
toks <- tokens(corp, remove_punct = TRUE)
toks <- tokens_remove(toks, stopwords("en"))

# economy keywords
eco <- char_context(toks, 'econom*')
head(eco, 20)

tstat_eco <- textstat_context(toks, 'econom*')
head(tstat_eco)
```

```
# politics keywords
pol <- char_context(toks, 'politi*')
head(pol, 20)

# politics keywords
tstat_pol <- textstat_context(toks, 'politi*')
head(tstat_pol)
```

Index

- * **data**
 - data_textmodel_lss_russianprotests, 3
- as.seedwords, 2
- char_context (textstat_context), 9
- char_keyness (textstat_context), 9
- cohesion, 2
- corpus, 4
- data_dictionary_ideology, 3
- data_dictionary_sentiment, 3
- data_textmodel_lss_russianprotests, 3
- diagnosys, 4
- dictionary, 2
- irlba::irlba(), 7
- locfit(), 5
- loess(), 5
- lp(), 5
- quanteda.textstats::textstat_simil(), 7
- quanteda::dfm(), 6
- quanteda::dfm_weight(), 6
- quanteda::fcm(), 6
- quanteda::pattern, 8
- quanteda::pattern(), 10
- quanteda::tokens(), 10
- quanteda::valuetype(), 10
- rsparse::GloVe(), 7
- RSpectra::svds(), 7
- seedwords, 4
- smooth_lss, 5
- textmodel_lss, 5
- textplot_simil, 8
- textplot_terms, 8
- textstat_context, 9
- textstat_keyness(), 10
- tokens_select(), 10