Package 'seededlda'

May 9, 2026

Type Package

Title Seeded Sequential LDA for Topic Modeling

Version 1.4.3

Description
Seeded Sequential LDA can classify sentences of texts into pre-define topics with a small number of seed words (Watanabe & Baturo, 2023) <doi:10.1177/08944393231178605>. Implements Seeded LDA (Lu et al., 2010) <doi:10.1109/ICDMW.2011.125> and Sequential LDA (Du et al., 2012) <doi:10.1007/s10115-011-0425-1> with the distributed LDA algorithm (Newman, et al., 2009) for parallel computing.

License GPL-3

LazyData TRUE

URL https://github.com/koheiw/seededlda,
https://koheiw.github.io/seededlda/

BugReports https://github.com/koheiw/seededlda/issues

Encoding UTF-8

Depends R (>= 3.5.0)

Imports quanteda (>= 4.0.0), methods, proxyC (>= 0.3.1), Matrix, utils

LinkingTo Rcpp, RcppArmadillo (>= 0.7.600.1.0), quanteda, testthat

Suggests spelling, testthat, topicmodels

RoxygenNote 7.3.3

Language en-US

NeedsCompilation yes

Author Kohei Watanabe [aut, cre, cph],
Phan Xuan-Hieu [aut, cph] (GibbsLDA++)

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

Repository CRAN

Date/Publication 2025-09-28 13:20:02 UTC

Contents

`as.dictionary.textmodel_lda` *Create a dictionary from topic terms*

Description

`as.dictionary()` creates a quanteda::dictionary object from top topic terms.

Usage

```
## S3 method for class 'textmodel_lda'
as.dictionary(x, n = 10, separator = NULL, ...)
```

Arguments

`x`	a model fitted by `textmodel_lda()`.
`n`	the number of terms in the dictionary for each topic.
`separator`	the character in between multi-word dictionary values.
`...`	not used.

Value

Returns a quanteda::dictionary object.

data_corpus_moviereviews

Movie reviews from Pang and Lee (2004)

Description

A corpus object containing 2,000 movie reviews.

Source

https://www.cs.cornell.edu/people/pabo/movie-review-data/

References

Pang, B., Lee, L. (2004) "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts.", Proceedings of the ACL.

divergence *Optimize the number of topics for LDA*

Description

`divergence()` computes the regularized topic divergence scores to help users to find the optimal number of topics for LDA.

Usage

```
divergence(
  x,
  min_size = 0.01,
  select = NULL,
  regularize = TRUE,
  newdata = NULL,
  ...
)
```

Arguments

x	a LDA model fitted by `textmodel_seededlda()` or `textmodel_lda()`.
min_size	the minimum size of topics for regularized topic divergence. Ignored when `regularize = FALSE`.
select	names of topics for which the divergence is computed.
regularize	if `TRUE`, returns the regularized divergence.
newdata	if provided, `theta` and `phi` are estimated through fresh Gibbs sampling.
...	additional arguments passed to textmodel_lda.

Details

`divergence()` computes the average Jensen-Shannon divergence between all the pairs of topic vectors in `x$phi`. The divergence score maximizes when the chosen number of topic `k` is optimal (Deveaud et al., 2014). The regularized divergence penalizes topics smaller than `min_size` to avoid fragmentation (Watanabe & Baturo, forthcoming).

Value

Returns a singple numeric value.

References

Deveaud, Romain et al. (2014). "Accurate and Effective Latent Concept Modeling for Ad Hoc Information Retrieval". doi:10.3166/DN.17.1.61-84. *Document Numérique*.

Watanabe, Kohei & Baturo, Alexander. (2023). "Seeded Sequential LDA: A Semi-supervised Algorithm for Topic-specific Analysis of Sentences". doi:10.1177/08944393231178605. *Social Science Computer Review*.

See Also

perplexity

`perplexity` *Optimize the hyper-parameters for LDA*

Description

`perplexity()` computes the perplexity score to help users to chose the optimal values of hyper-parameters for LDA.

Usage

```
perplexity(x, newdata = NULL, ...)
```

Arguments

`x`	a LDA model fitted by `textmodel_seededlda()` or `textmodel_lda()`.
`newdata`	if provided, `theta` and `phi` are estimated through fresh Gibbs sampling.
`...`	additional arguments passed to textmodel_lda.

Details

`perplexity()` predicts the distribution of words in the dfm based on `x$alpha` and `x$gamma` and then compute the sum of disparity between their predicted and observed frequencies. The perplexity score minimizes when the chosen values of hyper-parameters such as `k`, `alpha` and `gamma` are optimal.

Value

Returns a singple numeric value.

See Also

divergence

`sizes` *Compute the sizes of topics*

Description

Compute the sizes of topics as the proportions of topic words in the corpus.

Usage

```
sizes(x)
```

Arguments

x	a LDA model fitted by `textmodel_seededlda()` or `textmodel_lda()`

Value

a numeric vector in the same lengths as k.

`terms` *Extract most likely terms*

Description

`terms()` returns the most likely terms for topics based on the `phi` parameter.

Usage

```
terms(x, n = 10)
```

Arguments

x	a LDA model fitted by `textmodel_seededlda()` or `textmodel_lda()`.
n	the number of terms to be extracted.

Details

Topic terms are sorted in the descending order within topics based on the values in `x$phi`.

Value

Returns a character matrix with the most frequent words in each topic.

`textmodel_lda` *Unsupervised Latent Dirichlet allocation*

Description

Implements unsupervised Latent Dirichlet allocation (LDA). Users can run Seeded LDA by setting `gamma > 0`.

Usage

```
textmodel_lda(
  x,
  k = 10,
  max_iter = 2000,
  auto_iter = FALSE,
  alpha = 0.5,
  beta = 0.1,
  gamma = 0,
  adjust_alpha = 0,
  model = NULL,
  update_model = FALSE,
  batch_size = 1,
  verbose = quanteda_options("verbose")
)
```

Arguments

`x`	the dfm on which the model will be fit.
`k`	the number of topics.
`max_iter`	the maximum number of iteration in Gibbs sampling.
`auto_iter`	if `TRUE`, stops Gibbs sampling on convergence before reaching `max_iter`. See details.
`alpha`	the values to smooth topic-document distribution.
`beta`	the values to smooth topic-word distribution.
`gamma`	a parameter to determine change of topics between sentences or paragraphs. When `gamma > 0`, Gibbs sampling of topics for the current document is affected by the previous document's topics.
`adjust_alpha`	[experimental] if `adjust_alpha > 0`, automatically adjust `alpha` by the size of the topics. The smallest value of adjusted `alpha` will be `alpha * (1 - adjust_alpha)`.
`model`	a fitted LDA model; if provided, `textmodel_lda()` inherits parameters from an existing model. See details.

update_model	if TRUE, update the terms of model to recognize unseen words.
batch_size	split the corpus into the smaller batches (specified in proportion) for distributed computing; it is disabled when a batch include all the documents batch_size = 1.0. See details.
verbose	logical; if TRUE print diagnostic information during fitting.

Details

If auto_iter = TRUE, the iteration stops even before max_iter when delta <= 0. delta is computed to measure the changes in the number of words whose topics are updated by the Gibbs sampler in every 100 iteration as shown in the verbose message.

If batch_size < 1.0, the corpus is partitioned into sub-corpora of ndoc(x) * batch_size documents for Gibbs sampling in sub-processes with synchronization of parameters in every 10 iteration. Parallel processing is more efficient when batch_size is small (e.g. 0.01). The algorithm is the Approximate Distributed LDA proposed by Newman et al. (2009). User can changed the number of sub-processes used for the parallel computing via options(seededlda_threads).

set.seed() should be called immediately before textmodel_lda() or textmodel_seededlda() to control random topic assignment. If the random number seed is the same, the serial algorithm produces identical results; the parallel algorithm produces non-identical results because it classifies documents in different orders using multiple processors.

To predict topics of new documents (i.e. out-of-sample), first, create a new LDA model from a existing LDA model passed to model in textmodel_lda(); second, apply topics() to the new model. The model argument takes objects created either by textmodel_lda() or textmodel_seededlda().

Value

Returns a list of model parameters:

k	the number of topics.
last_iter	the number of iterations in Gibbs sampling.
max_iter	the maximum number of iterations in Gibbs sampling.
auto_iter	the use of auto_iter
adjust_alpha	the value of adjust_alpha.
alpha	the smoothing parameter for theta.
beta	the smoothing parameter for phi.
epsilon	the amount of adjustment for adjust_alpha.
gamma	the gamma parameter for Sequential LDA.
phi	the posterior distribution of words over topics.
theta	the posterior distribution of topics over documents.
words	the raw frequency count of words assigned to topics.
data	the original input of x.
concatenator	the concatenator in x.
call	the command used to execute the function.
version	the version of the seededlda package.

References

Newman, D., Asuncion, A., Smyth, P., & Welling, M. (2009). Distributed Algorithms for Topic Models. The Journal of Machine Learning Research, 10, 1801–1828.

Examples

```
require(seededlda)
require(quanteda)

corp <- head(data_corpus_moviereviews, 500)
toks <- tokens(corp, remove_punct = TRUE, remove_symbols = TRUE, remove_number = TRUE)
dfmt <- dfm(toks) %>%
    dfm_remove(stopwords("en"), min_nchar = 2) %>%
    dfm_trim(max_docfreq = 0.1, docfreq_type = "prop")

lda <- textmodel_lda(dfmt, k = 6, max_iter = 500) # 6 topics
terms(lda)
topics(lda)
```

---

textmodel_seededlda *Semisupervised Latent Dirichlet allocation*

Description

Implements semisupervised Latent Dirichlet allocation (Seeded LDA). `textmodel_seededlda()` allows users to specify topics using a seed word dictionary. Users can run Seeded Sequential LDA by setting `gamma > 0`.

Usage

```
textmodel_seededlda(
  x,
  dictionary,
  levels = 1,
  valuetype = c("glob", "regex", "fixed"),
  case_insensitive = TRUE,
  residual = 0,
  weight = 0.01,
  max_iter = 2000,
  auto_iter = FALSE,
  alpha = 0.5,
  beta = 0.1,
  gamma = 0,
  adjust_alpha = 0,
  batch_size = 1,
  ...,
  verbose = quanteda_options("verbose")
)
```

Arguments

x	the dfm on which the model will be fit.
dictionary	a quanteda::dictionary() with seed words that define topics.
levels	levels of entities in a hierarchical dictionary to be used as seed words. See also quanteda::flatten_dictionary.
valuetype	see quanteda::valuetype
case_insensitive	see quanteda::valuetype
residual	the number of undefined topics. They are named "other" by default, but it can be changed via base::options(seededlda_residual_name).
weight	determines the size of pseudo counts given to matched seed words.
max_iter	the maximum number of iteration in Gibbs sampling.
auto_iter	if TRUE, stops Gibbs sampling on convergence before reaching max_iter. See details.
alpha	the values to smooth topic-document distribution.
beta	the values to smooth topic-word distribution.
gamma	a parameter to determine change of topics between sentences or paragraphs. When gamma > 0, Gibbs sampling of topics for the current document is affected by the previous document's topics.
adjust_alpha	[experimental] if adjust_alpha > 0, automatically adjust alpha by the size of the topics. The smallest value of adjusted alpha will be alpha * (1 - adjust_alpha).
batch_size	split the corpus into the smaller batches (specified in proportion) for distributed computing; it is disabled when a batch include all the documents batch_size = 1.0. See details.
...	passed to quanteda::dfm_trim to restrict seed words based on their term or document frequency. This is useful when glob patterns in the dictionary match too many words.
verbose	logical; if TRUE print diagnostic information during fitting.

Value

The same as textmodel_lda() with extra elements for dictionary.

References

Lu, Bin et al. (2011). "Multi-aspect Sentiment Analysis with Topic Models". doi:10.5555/2117693.2119585. *Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops*.

Watanabe, Kohei & Zhou, Yuan. (2020). "Theory-Driven Analysis of Large Corpora: Semisupervised Topic Classification of the UN Speeches". doi:10.1177/0894439320907027. *Social Science Computer Review*.

Watanabe, Kohei & Baturo, Alexander. (2023). "Seeded Sequential LDA: A Semi-supervised Algorithm for Topic-specific Analysis of Sentences". doi:10.1177/08944393231178605. *Social Science Computer Review*.

Examples

```
require(seededlda)
require(quanteda)

corp <- head(data_corpus_moviereviews, 500)
toks <- tokens(corp, remove_punct = TRUE, remove_symbols = TRUE, remove_number = TRUE)
dfmt <- dfm(toks) %>%
    dfm_remove(stopwords("en"), min_nchar = 2) %>%
    dfm_trim(max_docfreq = 0.1, docfreq_type = "prop")

dict <- dictionary(list(people = c("family", "couple", "kids"),
                        space = c("alien", "planet", "space"),
                        moster = c("monster*", "ghost*", "zombie*"),
                        war = c("war", "soldier*", "tanks"),
                        crime = c("crime*", "murder", "killer")))
lda_seed <- textmodel_seededlda(dfmt, dict, residual = TRUE, min_termfreq = 10,
                                max_iter = 500)
terms(lda_seed)
topics(lda_seed)
```

---

textmodel_seqlda *Sequential Latent Dirichlet allocation*

Description

Implements Sequential Latent Dirichlet allocation (Sequential LDA). `textmodel_seqlda()` allows the users to classify sentences of texts. It considers the topics of previous document in inferring the topics of currency document. `textmodel_seqlda()` is a shortcut equivalent to `textmodel_lda(gamma = 0.5)`. Seeded Sequential LDA is `textmodel_seededlda(gamma = 0.5)`.

Usage

```
textmodel_seqlda(
  x,
  k = 10,
  max_iter = 2000,
  auto_iter = FALSE,
  alpha = 0.5,
  beta = 0.1,
  batch_size = 1,
  model = NULL,
  verbose = quanteda_options("verbose")
)
```

Arguments

x	the dfm on which the model will be fit.
k	the number of topics.
max_iter	the maximum number of iteration in Gibbs sampling.
auto_iter	if TRUE, stops Gibbs sampling on convergence before reaching max_iter. See details.
alpha	the values to smooth topic-document distribution.
beta	the values to smooth topic-word distribution.
batch_size	split the corpus into the smaller batches (specified in proportion) for distributed computing; it is disabled when a batch include all the documents batch_size = 1.0. See details.
model	a fitted LDA model; if provided, textmodel_lda() inherits parameters from an existing model. See details.
verbose	logical; if TRUE print diagnostic information during fitting.

Value

The same as textmodel_lda()

References

Du, Lan et al. (2012). "Sequential Latent Dirichlet Allocation". doi.org/10.1007/s10115-011-0425-1. *Knowledge and Information Systems*.

Watanabe, Kohei & Baturo, Alexander. (2023). "Seeded Sequential LDA: A Semi-supervised Algorithm for Topic-specific Analysis of Sentences". doi:10.1177/08944393231178605. *Social Science Computer Review*.

Examples

```
require(seededlda)
require(quanteda)

corp <- head(data_corpus_moviereviews, 500) %>%
    corpus_reshape()
toks <- tokens(corp, remove_punct = TRUE, remove_symbols = TRUE, remove_number = TRUE)
dfmt <- dfm(toks) %>%
    dfm_remove(stopwords("en"), min_nchar = 2) %>%
    dfm_trim(max_docfreq = 0.01, docfreq_type = "prop")

lda_seq <- textmodel_seqlda(dfmt, k = 6, max_iter = 500) # 6 topics
terms(lda_seq)
topics(lda_seq)
```

topics *Extract most likely topics*

Description

`topics()` returns the most likely topics for documents based on the `theta` parameter.

Usage

```
topics(x, min_prob = 0, select = NULL)
```

Arguments

`x`	a LDA model fitted by `textmodel_seededlda()` or `textmodel_lda()`
`min_prob`	ignores topics if their probability is lower than this value.
`select`	returns the selected topic with the highest probability; specify by the names of columns in `x$theta`.

Details

Users can access the original matrix `x$theta` for likelihood scores; run `max.col(x$theta)` to obtain the same result as `topics(x)`.

Value

Returns predicted topics as a vector.

Index