Package ‘seededlda’

July 3, 2024

Type Package
Title Seeded Sequential LDA for Topic Modeling
Version 1.3.2
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), quanteda (>= 4.0.0), methods, proxyC (>= 0.3.1)
Imports Matrix
LinkingTo Rcpp, RcppArmadillo (>= 0.7.600.1.0), quanteda, testthat
Suggests spelling, testthat, topicmodels, keyATM
RoxygenNote 7.3.1
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 2024-07-03 07:20:02 UTC
Contents

data_corpus_moviereviews ........................................... 2
divergence .................................................................. 2
perplexity .................................................................. 3
sizes ......................................................................... 4
terms ........................................................................ 5
textmodel_lda ............................................................. 5
textmodel_seededlda .................................................. 7
textmodel_seqlda ....................................................... 9
topics ...................................................................... 11

Index 12

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 Sum-
marization 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 single numeric value.

References


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

```r
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 single 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, or words, 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`: number of terms to be extracted.

**Details**

Users can access the original matrix `x$phi` for likelihood scores.

**Value**

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**

```r
textmodel_lda(
  x,
  k = 10,
  max_iter = 2000,
  auto_iter = FALSE,
  alpha = 0.5,
  beta = 0.1,
  gamma = 0,
  model = NULL,
  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.
- **model**: a fitted LDA model; if provided, `textmodel_lda()` inherits parameters from an existing model. See details.
- **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
- **phi**: the distribution of words over topics.
- **theta**: the distribution of topics over documents.
textmodel_seededlda

words the raw frequency count of words assigned to topics.
data the original input of x.
call the command used to execute the function.
version the version of the seededlda package.

References


See Also

LDA weightedLDA

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)

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, valuertype = 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,
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.
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_llda() with extra elements for dictionary.
References


See Also

keyATM

Examples

```r
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


Examples

  require(seededlda)
  require(quanteda)
corp <- head(data_corpus_movies, 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

* textmodel
  textmodel_lda, 5

data_corpus_movieviews, 2
divergence, 2, 4

keyATM, 9

LDA, 7

perplexity, 3, 3

quanteda::dfm_trim, 8
quanteda::dictionary(), 8
quanteda::flatten_dictionary, 8
quanteda::valuetype, 8

sizes, 4

terms, 5
textmodel_lda, 3, 4, 5
textmodel_lda(), 3–5, 8, 10, 11
textmodel_seededlda, 7
textmodel_seededlda(), 3–5, 11
textmodel_seqlda, 9
topics, 11
topics(), 6

weightedLDA, 7