Package 'GMTM'

November 18, 2026

Title Gaussian Mixture Topic Models

Version 0.3.0

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

Description Gaussian mixture models (GMM) and k-means for topic analysis of dense document vectors. The underlying clustering functions rely on the Armadillo library (Sanderson & Curtin, 2017) <doi:10.1109/ICSPCS.2017.8270510>.

License Apache License (>= 2.0)

Encoding UTF-8

Depends R (>= 3.5.0)

Imports quanteda (>= 4.0.0), proxyC, wordvector (>= 0.6.4), Rcpp, RcppArmadillo

Suggests testthat, spelling, rmarkdown, knitr, withr

LinkingTo Rcpp, RcppArmadillo (>= 0.7.600.1.0)

Language en-US

Config/roxygen2/version 8.1.0

NeedsCompilation yes

Author Kohei Watanabe [aut, cre, cph] (ORCID: <https://orcid.org/0000-0001-6519-5265>), Sanderson Conrad [ctb, cph] (C++ code for GMM and k-means), Curtin Ryan [ctb, cph] (C++ code for GMM and k-means)

Repository CRAN

Date/Publication 2026-11-18 09:10:11 UTC

Contents

as.seedwords 2
probability.textmodel_gmm 3
terms 3
textmodel_gmm 4
textmodel_kmeans 5
topics 7

Index **8**

`as.seedwords` *Convert a dictionary to a seed word matrix*

Description

Convert a dictionary to a seed word matrix

Usage

```
as.seedwords(x, model, residual = 0, levels = 1)
```

Arguments

`x`	a quanteda::dictionary of seed words.
`model`	a wordvector::textmodel_word2vec object.
`residual`	the number of unseeded topics.
`levels`	integers specifying the levels of entries in a hierarchical dictionary.

Details

Word vectors of the dictionary values within the same key are all averaged as a single seed vector. If dictionary values are multi-word expressions, they are matched against both individual and compounded words in the `model`.

Unseeded topics are labeled "other", but it can be changed via options("GMTM.residual.name").

Value

Returns a seed word matrix.

Examples

```
library(quanteda)
library(wordvector)
options(wordvector_threads = 2)

corp <- head(wordvector::data_corpus_news2014, 1000)
toks <- tokens(corp, remove_punct = TRUE,
               remove_symbols = TRUE, remove_numbers = TRUE) %>%
        tokens_remove(stopwords("en"), min_nchar = 2)
wov <- textmodel_word2vec(toks, dim = 50)

dict <- dictionary(list(eco = "econom*", sec = "securit*", spo = "sport*",
                        pol = "politi*", cri = "crime"))
seed <- as.seedwords(dict, wov, residual = 2)
```

`probability.textmodel_gmm` *Extract the probabilities for topics*

Description

Extract the probabilities for topics

Usage

```
## S3 method for class 'textmodel_gmm'
probability(x, group = NULL, ...)
```

Arguments

`x`	a fitted model.
`group`	a factor to group documents and average their probability for topics. Ignored if `x` is a `textmodel_kmeans` object.
`...`	not used.

Value

Returns the probabilities of topics as a matrix.

`terms` *Extract lost likely topic terms from documents*

Description

Extract lost likely topic terms from documents

Usage

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

Arguments

`x`	a fitted model or a factor from `GMTM::topics()`.
`data`	a quanteda::dfm or quanteda::tokens from which words are extracted for each topic.
`n`	the number of topic words.
`...`	not used.

Details

To identify topic terms, `data` must be provided along with a fitted model because the information about the frequency of individual words are lost in document vectors.

When `x` is a `textmodel_gmm`, `data` is weighted by the documents' probabilities for topics to extract most likely terms. When `x` is `textmodel_kmeans` or the output from topics, `data` is grouped by the topic and weighted by TF-IDF to select the most distinctive words for each topic (known as c-TF-IDF).

Value

Returns a character matrix with the most distinctive words for each topic.

`textmodel_gmm` *Topic analysis using Gaussian mixture models*

Description

Perform topic analysis of document vectors using Gaussian mixture models.

Usage

```
textmodel_gmm(
  x,
  k = 10,
  model = NULL,
  seeds = NULL,
  include_prob = TRUE,
  omit = NULL,
  verbose = quanteda_options("verbose"),
  ...
)

as.textmodel_gmm(x, include_prob = TRUE)
```

Arguments

`x`	a wordvector::textmodel_doc2vec or a dense matrix of document vectors in the rows.
`k`	the number of topics to identify.
`model`	a fitted model from which initial centroids are extracted.
`seeds`	a matrix created using as.seedwords.
`include_prob`	if `TRUE`, the resulting object includes `topic.likelihood` that records probabilities for all the topics. `include_prob = FALSE` reduces the size of the object significantly but limits its usage.
`omit`	[experimental] indices of singular values of `x` to be zero. See the details.
`verbose`	print the progress if `TRUE`.
`...`	passed to the underlying function.

Details

multi-threading:

Users can change the number of threads for the parallel computing via `options(GMTM.threads)` or `OMP_THREAD_LIMIT` in the environmental variable. To reproduce results, set `options(GMTM.threads = 1)` and call `set.seed()` immediately before `textmodel_gmm()` or `textmodel_kmeans()`.

On MacOS, only one thread is used regardless of `GMTM.threads` because CRAN's toolchain for the platform does not support OpenMP.

noise reduction:

`omit` is used to reduce the noise in the `x` by applying `base::svd` before clustering. If it is not `NULL`, singular values corresponding to `omit` are set to zero, removing their variance in `x`. See Chan et al. (2020) doi:10.1080/19312458.2020.1812555 for the methodology.

additional arguments:

The number of iterations in k-means (`iter_km`) and expectation maximization (`iter_em`) stages can be set via `...`.

Value

Returns a fitted `textmodel_gmm` object.

Examples

```
library(quanteda)
library(wordvector)
options(wordvector_threads = 2)

corp <- head(wordvector::data_corpus_news2014, 1000)
toks <- tokens(corp, remove_punct = TRUE,
               remove_symbols = TRUE, remove_numbers = TRUE) %>%
        tokens_remove(stopwords("en"), min_nchar = 2)
dfmt <- dfm(toks)
wov <- textmodel_word2vec(toks, dim = 50)
dov <- as.textmodel_doc2vec(dfmt, wov)

gmm <- textmodel_gmm(dov, k = 10)
table(topics(gmm))
terms(gmm, dfmt)
```

---

textmodel_kmeans *Topic analysis using k-means*

Description

Perform topic analysis of document vectors using k-means.

Usage

```
textmodel_kmeans(
  x,
  k = 10,
  model = NULL,
  seeds = NULL,
  verbose = quanteda_options("verbose"),
  ...
)

as.textmodel_kmeans(x)
```

Arguments

x	a wordvector::textmodel_doc2vec or a dense matrix of document vectors in the rows.
k	the number of topics to identify.
model	a fitted model from which initial centroids are extracted.
seeds	a matrix created using as.seedwords.
verbose	print the progress if TRUE.
...	passed to the underlying function.

Value

Returns a fitted textmodel_kmeans object.

Examples

```
library(quanteda)
library(wordvector)
options(wordvector_threads = 2)

corp <- head(wordvector::data_corpus_news2014, 1000)
toks <- tokens(corp, remove_punct = TRUE,
               remove_symbols = TRUE, remove_numbers = TRUE) %>%
        tokens_remove(stopwords("en"), min_nchar = 2)
dfmt <- dfm(toks)
wov <- textmodel_word2vec(toks, dim = 50)
dov <- as.textmodel_doc2vec(dfmt, wov)

km <- textmodel_kmeans(dov, k = 10)
table(topics(km))
terms(km, dfmt)
```

`topics` *Extract the most likely topics of documents*

Description

Extract the most likely topics of documents

Usage

```
topics(x, group = NULL, ...)
```

Arguments

`x`	a fitted model.
`group`	a factor to group documents and average their probability for topics. Ignored if `x` is a `textmodel_kmeans` object.
`...`	not used.

Value

Returns predicted topics as a vector.

Index

as.seedwords, 2, *4*, *6*
as.textmodel_gmm *(*textmodel_gmm*)*, 4
as.textmodel_kmeans *(*textmodel_kmeans*)*, 5

probability.textmodel_gmm, 3

quanteda::dfm, *3*
quanteda::dictionary, *2*
quanteda::tokens, *3*

terms, 3
textmodel_gmm, 4
textmodel_kmeans, 5
topics, *4*, 7

wordvector::textmodel_doc2vec, *4*, *6*
wordvector::textmodel_word2vec, *2*