Package 'BERTopic'

November 13, 2025

Type Package

Title Topic Modeling with 'BERTopic' in R

Version 0.1.0

Description Interface to the Python package 'BERTopic' for transformer-based topic modeling. Provides R wrappers to fit BERTopic models, transform new documents, update and reduce topics, extract topic- and document-level information, and generate interactive visualizations. Python backends and dependencies are managed via the 'reticulate' package.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 3.5)

Imports reticulate,
rlang,
tibble,
utils

Suggests Matrix,
htmltools,
testthat (>= 3.1.0)

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/testthat/edition 3

URL https://github.com/Feng-Ji-Lab/BERTopic

BugReports https://github.com/Feng-Ji-Lab/BERTopic/issues

Language en-US

Contents

`as.data.frame.bertopic_r` *Coerce to data.frame*

Description

Coerce to data.frame

Usage

```
## S3 method for class 'bertopic_r'
as.data.frame(x, ...)
```

Arguments

x	A "bertopic_r" model.
...	Unused.

Value

A data.frame equal to `bertopic_topics()`.

---

bertopic_as_document_topic_matrix *Coerce to a document-topic probability matrix*

Description

Extract the document-topic probabilities as a matrix. If probabilities were not computed during fitting, returns NULL (with a warning).

Usage

```
bertopic_as_document_topic_matrix(model, sparse = TRUE, prefix = TRUE)
```

Arguments

model	A "bertopic_r" model.
sparse	Logical; if TRUE and Matrix is available, returns a sparse matrix.
prefix	Logical; if TRUE, prefix columns as "topic_".

Value

A matrix or sparse Matrix of size n_docs x n_topics, or NULL.

---

bertopic_available *Is Python + BERTopic available?*

Description

Checks whether the active Python (as initialized by **reticulate**) can import the key modules needed for BERTopic.

Usage

```
bertopic_available()
```

Value

Logical scalar.

Examples

```
## Not run:
bertopic_available()

## End(Not run)
```

`bertopic_find_topics` *Find nearest topics for a query string*

Description

Use `BERTopic.find_topics()` to retrieve the closest topics for a query string. Augments topic IDs/scores with topic labels when available.

Usage

```
bertopic_find_topics(model, query_text, top_n = 5L)
```

Arguments

`model`	A "bertopic_r" model.
`query_text`	A length-1 character query.
`top_n`	Number of nearest topics to return.

Value

A tibble with columns `topic`, `score`, and `label`.

`bertopic_fit` *Fit BERTopic from R*

Description

A high-level wrapper around Python 'BERTopic'. Python dependencies are checked at runtime.

Usage

```
bertopic_fit(text, embeddings = NULL, ...)
```

Arguments

`text`	Character vector of documents.
`embeddings`	Optional numeric matrix (n_docs x dim). If supplied, passed through to Python.
`...`	Additional arguments forwarded to `bertopic.BERTopic(...)`.

Value

An S3 object of class "bertopic_r" containing:

- `.py`: the underlying Python model (reticulate object)
- `topics`: integer vector of topic assignments
- `probs`: numeric matrix/data frame of topic probabilities (if available)

Examples

```
## Not run:
if (reticulate::py_module_available("bertopic")) {
  m <- bertopic_fit(c("a doc", "another doc"))
  print(class(m))
}

## End(Not run)
```

`bertopic_get_document_info` *Document-level information*

Description

Retrieve document-level information for the provided documents.

Usage

```
bertopic_get_document_info(model, docs)
```

Arguments

`model`	A "bertopic_r" model.
`docs`	Character vector of documents to query (required).

Value

A tibble with document-level information.

bertopic_get_representative_docs

Representative documents for a topic

Description

Retrieve representative documents for a given topic using `BERTopic.get_representative_docs()`. Falls back across signature variants.

Usage

```
bertopic_get_representative_docs(model, topic_id, top_n = 5L)
```

Arguments

`model`	A "bertopic_r" model.
`topic_id`	Integer topic id.
`top_n`	Number of representative documents to return.

Value

A tibble with columns `rank` and `document`. If scores are available in the current BERTopic version, a `score` column is included.

bertopic_has_embedding_model

Does the model have a usable embedding model?

Description

Does the model have a usable embedding model?

Usage

```
bertopic_has_embedding_model(model)
```

Arguments

`model`	A "bertopic_r" model.

Value

Logical; TRUE if `embedding_model` is present and not None.

bertopic_load *Load a BERTopic model*

Description

Load a BERTopic model from disk that was saved with `bertopic_save()`.

Usage

```
bertopic_load(path)
```

Arguments

`path`	Path used in `bertopic_save()` (file or directory).

Value

A "bertopic_r" object with the loaded Python model.

bertopic_reduce_topics *Reduce/merge topics*

Description

Wrapper over Python `reduce_topics`, compatible with multiple signatures.

Usage

```
bertopic_reduce_topics(
  model,
  nr_topics = "auto",
  representation_model = NULL,
  docs = NULL
)
```

Arguments

`model`	A "bertopic_r" model.
`nr_topics`	Target number (integer) or "auto".
`representation_model`	Optional Python representation model.
`docs`	Optional character vector of training docs (used if required by backend).

Value

The input model (invisibly).

bertopic_save *Save a BERTopic model*

Description

Save a fitted BERTopic model to disk. Depending on the serialization method, this may produce either a single file (e.g., *.pkl / *.pt / *.safetensors) or a directory bundle. The function does not pre-create the target path; it only ensures the parent directory exists and lets BERTopic decide the layout.

Usage

```
bertopic_save(
  model,
  path,
  serialization = c("pickle", "safetensors", "pt"),
  save_embedding_model = FALSE,
  overwrite = FALSE
)
```

Arguments

model	A "bertopic_r" model.
path	Destination path (file or directory, as required by BERTopic).
serialization	One of "pickle", "safetensors", or "pt". Default "pickle".
save_embedding_model	Logical; whether to include the embedding model. Default FALSE.
overwrite	Logical; if TRUE and the target exists, it will be replaced.

Value

Invisibly returns the normalized path.

bertopic_self_check *Quick self-check for the BERTopic R interface*

Description

Runs a quick end-to-end smoke test:

- Report Python path/version.
- Verify that `bertopic` is importable and report its version.
- Minimal round trip: `fit` -> `transform` -> `save` -> `load`.

Usage

```
bertopic_self_check()
```

Value

A named list with fields:

python_ok Logical.

bertopic_ok Logical.

roundtrip_ok Logical.

details Character vector of diagnostic messages.

Examples

```
## Not run:
bertopic_self_check()

## End(Not run)
```

`bertopic_session_info` *Summarize Python/BERTopic session info*

Description

Summarize Python/BERTopic session info

Usage

```
bertopic_session_info()
```

Value

A named list containing paths, versions, and module availability:

python Path of the active Python.

libpython Path to libpython, if any.

version Python version string.

numpy Whether NumPy is available.

numpy_version NumPy version string (if available).

modules A data.frame with availability for key modules.

Examples

```
## Not run:
bertopic_session_info()

## End(Not run)
```

bertopic_set_embedding_model

Replace or set the embedding model

Description

Set a new embedding model on a fitted BERTopic instance. This enables `transform()` after loading when the embedding model was not saved.

Usage

```
bertopic_set_embedding_model(model, embedding_model)
```

Arguments

`model`	A "bertopic_r" model.
`embedding_model`	Either a character identifier (e.g., "all-MiniLM-L6-v2") or a Python embedding model object (e.g., a SentenceTransformer instance).

Value

The input model (invisibly).

bertopic_set_topic_labels

Relabel topics

Description

Set custom labels for topics. Accepts a named character vector or a data.frame with columns `topic` and `label`.

Usage

```
bertopic_set_topic_labels(model, labels)
```

Arguments

`model`	A "bertopic_r" model.
`labels`	A named character vector (names are topic ids) or a data.frame.

Value

The input model (invisibly).

bertopic_topics *Get topic info as a tibble*

Description

Get topic info as a tibble

Usage

```
bertopic_topics(model)
```

Arguments

model	A "bertopic_r" object returned by bertopic_fit().

Value

A tibble with topic-level information from Python get_topic_info().

bertopic_topics_over_time *Compute topics over time*

Description

Wrapper for Python BERTopic.topics_over_time(). Returns a tibble and attaches the original Python dataframe in the "_py" attribute for use in visualization.

Usage

```
bertopic_topics_over_time(
  model,
  docs,
  timestamps,
  nr_bins = NULL,
  datetime_format = NULL
)
```

Arguments

model	A "bertopic_r" model.
docs	Character vector of documents.
timestamps	A vector of timestamps (Date, POSIXt, or character).
nr_bins	Optional number of temporal bins.
datetime_format	Optional strftime-style format if timestamps are strings.

Value

A tibble with topics-over-time data; attribute "_py" stores the original Python dataframe.

bertopic_topic_terms *Get top terms for a topic*

Description

Get top terms for a topic

Usage

```
bertopic_topic_terms(model, topic_id, top_n = 10L)
```

Arguments

model	A "bertopic_r" model
topic_id	Integer topic id
top_n	Number of top terms to return

Value

A tibble with columns `term` and `weight`

bertopic_transform *Transform new documents with a fitted BERTopic model*

Description

Transform new documents with a fitted BERTopic model

Usage

```
bertopic_transform(model, new_text, embeddings = NULL)
```

Arguments

model	A "bertopic_r" model from `bertopic_fit()`.
new_text	Character vector of new documents.
embeddings	Optional numeric matrix for new documents.

Value

A list with `topics` and `probs` for the new documents.

bertopic_update_topics

Update topic representations

Description

Call Python `BERTopic.update_topics()` to recompute topic representations.

Usage

```
bertopic_update_topics(model, text)
```

Arguments

`model`	A "bertopic_r" model.
`text`	Character vector of training documents used in `fit`.

Value

The input model (invisibly), updated in place on the Python side.

bertopic_visualize_barchart

Visualize a topic barchart

Description

Visualize a topic barchart

Usage

```
bertopic_visualize_barchart(model, topic_id = NULL, file = NULL)
```

Arguments

`model`	A "bertopic_r" model.
`topic_id`	Integer topic id. If NULL, a set of top topics is shown.
`file`	Optional HTML output path.

`bertopic_visualize_distribution` *Visualize topic probability distribution*

Description

Wrapper around Python `BERTopic.visualize_distribution()`. This function takes a single document's topic probability vector (e.g., one row from `probs`) and returns an interactive Plotly figure as HTML or writes it to disk.

Usage

```
bertopic_visualize_distribution(
  model,
  probs,
  min_probability = NULL,
  custom_labels = FALSE,
  title = NULL,
  width = NULL,
  height = NULL,
  file = NULL
)
```

Arguments

`model`	A "bertopic_r" model.
`probs`	Numeric vector of topic probabilities for a single document.
`min_probability`	Optional numeric scalar. If provided, only probabilities greater than this value are visualized (forwarded to `min_probability` in Python).
`custom_labels`	Logical or character scalar. If logical, whether to use custom topic labels as set via `set_topic_labels()`. If character, selects labels from other aspects (e.g., "Aspect1").
`title`	Optional character plot title.
`width, height`	Optional integer figure width/height in pixels.
`file`	Optional HTML output path. If NULL, an `htmltools::HTML` object is returned.

Value

If `file` is NULL, an `htmltools::HTML` object. Otherwise, the normalized file path is returned invisibly.

`bertopic_visualize_documents` *Visualize embedded documents*

Description

Visualize embedded documents

Usage

```
bertopic_visualize_documents(model, docs = NULL, file = NULL)
```

Arguments

`model`	A "bertopic_r" model.
`docs`	Optional character vector of documents to visualize.
`file`	Optional HTML output path.

`bertopic_visualize_heatmap` *Visualize topic similarity heatmap*

Description

Visualize topic similarity heatmap

Usage

```
bertopic_visualize_heatmap(model, file = NULL)
```

Arguments

`model`	A "bertopic_r" model.
`file`	Optional HTML output path.

`bertopic_visualize_hierarchical_documents` *Visualize hierarchical documents and topics*

Description

Wrapper around Python `BERTopic.visualize_hierarchical_documents()`. This function visualizes documents and their topics in 2D at different levels of a hierarchical topic structure.

Usage

```
bertopic_visualize_hierarchical_documents(
  model,
  docs,
  hierarchical_topics,
  topics = NULL,
  embeddings = NULL,
  reduced_embeddings = NULL,
  sample = NULL,
  hide_annotations = FALSE,
  hide_document_hover = TRUE,
  nr_levels = 10L,
  level_scale = c("linear", "log"),
  custom_labels = FALSE,
  title = NULL,
  width = NULL,
  height = NULL,
  file = NULL
)
```

Arguments

`model`	A "bertopic_r" model.
`docs`	Character vector of documents used in `fit` / `fit_transform`.
`hierarchical_topics`	A data frame or Python object as returned by `BERTopic.hierarchical_topics(docs, ...)`.
`topics`	Optional integer vector of topic IDs to visualize.
`embeddings`	Optional numeric matrix of document embeddings.
`reduced_embeddings`	Optional numeric matrix of 2D reduced embeddings.
`sample`	Optional numeric (0–1) or integer controlling subsampling of documents per topic (forwarded to Python).
`hide_annotations`	Logical; if TRUE, hide cluster labels in the plot.
`hide_document_hover`	Logical; if TRUE, hide document text on hover to speed up rendering.
`nr_levels`	Integer; number of hierarchy levels to display.

level_scale	Character, either "linear" or "log", controlling how hierarchy distances are scaled across levels.
custom_labels	Logical or character scalar controlling label behavior (forwarded to Python).
title	Optional character plot title.
width, height	Optional integer figure width/height in pixels.
file	Optional HTML output path. If NULL, an htmltools::HTML object is returned.

Value

If file is NULL, an htmltools::HTML object. Otherwise, the normalized file path is returned invisibly.

`bertopic_visualize_hierarchy` *Visualize hierarchical clustering of topics*

Description

Visualize hierarchical clustering of topics

Usage

```
bertopic_visualize_hierarchy(model, file = NULL)
```

Arguments

model	A "bertopic_r" model.
file	Optional HTML output path.

`bertopic_visualize_term_rank` *Visualize term rank evolution*

Description

Visualize term rank evolution

Usage

```
bertopic_visualize_term_rank(model, file = NULL)
```

Arguments

model	A "bertopic_r" model.
file	Optional HTML output path.

bertopic_visualize_topics *Visualize topic map*

Description

Visualize topic map

Usage

```
bertopic_visualize_topics(model, file = NULL)
```

Arguments

model	A "bertopic_r" model.
file	Optional HTML output path. If NULL, returns htmltools::HTML.

bertopic_visualize_topics_over_time *Visualize topics over time*

Description

Visualize topics over time

Usage

```
bertopic_visualize_topics_over_time(
  model,
  topics_over_time,
  top_n = 10L,
  file = NULL
)
```

Arguments

model	A "bertopic_r" model.
topics_over_time	A tibble returned by bertopic_topics_over_time(), or a Python dataframe compatible with visualize_topics_over_time().
top_n	Number of topics to display.
file	Optional HTML output path.

`bertopic_visualize_topics_per_class` *Visualize topics per class*

Description

Wrapper around Python `BERTopic.visualize_topics_per_class()`. This visualizes how topics are distributed across a set of classes, using the output of Python `topics_per_class(docs, classes)`.

Usage

```
bertopic_visualize_topics_per_class(
  model,
  topics_per_class,
  top_n_topics = 10L,
  topics = NULL,
  normalize_frequency = FALSE,
  custom_labels = FALSE,
  title = NULL,
  width = NULL,
  height = NULL,
  file = NULL
)
```

Arguments

`model`	A "bertopic_r" model.
`topics_per_class`	A data frame or Python object as returned by `BERTopic.topics_per_class(docs, classes)`.
`top_n_topics`	Integer; number of most frequent topics to display.
`topics`	Optional integer vector of topic IDs to include.
`normalize_frequency`	Logical; whether to normalize each topic's frequency within classes.
`custom_labels`	Logical or character scalar controlling label behavior (forwarded to Python).
`title`	Optional character plot title.
`width, height`	Optional integer figure width/height in pixels.
`file`	Optional HTML output path. If NULL, an `htmltools::HTML` object is returned.

Value

If `file` is NULL, an `htmltools::HTML` object. Otherwise, the normalized file path is returned invisibly.

coef.bertopic_r *Coefficients (top terms) for BERTopic*

Description

Coefficients (top terms) for BERTopic

Usage

```
## S3 method for class 'bertopic_r'
coef(object, top_n = 10L, ...)
```

Arguments

object	A "bertopic_r" model.
top_n	Number of terms per topic.
...	Unused.

Value

A data.frame with columns topic, term, weight.

fortify.bertopic_r *Fortify method for ggplot2*

Description

Fortify method for ggplot2

Usage

```
fortify.bertopic_r(model, data, ...)
```

Arguments

model	A "bertopic_r" model.
data	Ignored.
...	Unused.

Value

A data.frame of document-topic assignments.

install_py_deps *Install Python dependencies for BERTopic (auto route)*

Description

Tries Conda first (recommended). If Conda is unavailable, falls back to virtualenv. On success, prints which route was used.

Usage

```
install_py_deps(
  envname = "r-bertopic",
  python_version = "3.10",
  python = NULL,
  reinstall = FALSE,
  validate = TRUE,
  verbose = TRUE
)
```

Arguments

envname	Character. Environment name (both routes). Default "r-bertopic".
python_version	Character. Python version for Conda route, e.g. "3.10".
python	Optional path to python for virtualenv route.
reinstall	Logical. Recreate the environment if it exists (route-specific).
validate	Logical. Attempt to validate imports if reticulate is not already initialized to another Python.
verbose	Logical. Print progress.

Value

Invisibly, the path to the selected Python interpreter.

install_py_deps_conda *Install Python dependencies for BERTopic (Conda route)*

Description

Creates (or reuses) a Conda environment with a pinned Python toolchain, installs the scientific stack + PyTorch (CPU) + sentence-transformers, then installs `bertopic==0.16.0` via pip. Optionally validates imports.

Usage

```
install_py_deps_conda(
  envname = "r-bertopic",
  python_version = "3.10",
  reinstall = FALSE,
  validate = TRUE,
  verbose = TRUE
)
```

Arguments

envname	Character. Conda environment name. Default "r-bertopic".
python_version	Character. Python version to use, e.g. "3.10".
reinstall	Logical. If TRUE, delete any existing env and recreate.
validate	Logical. If TRUE, bind and validate imports (will skip if reticulate is already initialized to another Python).
verbose	Logical. Print progress messages.

Value

Invisibly returns the path to the Python executable inside the env.

Examples

```
## Not run:
install_py_deps_conda(envname = "r-bertopic", python_version = "3.10")

## End(Not run)
```

---

install_py_deps_venv *Install Python dependencies for BERTopic (virtualenv route)*

Description

Creates (or reuses) a virtualenv and installs bertopic==0.16.0 plus required dependencies via pip. Optionally validates imports.

Usage

```
install_py_deps_venv(
  envname = "r-bertopic",
  python = NULL,
  reinstall = FALSE,
  validate = TRUE,
  verbose = TRUE
)
```

Arguments

envname	Character. Virtualenv name. Default "r-bertopic".
python	Character. Path to a Python executable to create the venv with. If NULL, tries to find python / python3 on PATH.
reinstall	Logical. If TRUE, delete existing venv and recreate.
validate	Logical. If TRUE, bind and validate imports (will skip if reticulate is already initialized to another Python).
verbose	Logical. Print progress messages.

Value

Invisibly returns the path to the Python executable inside the venv.

Examples

```
## Not run:
install_py_deps_venv(envname = "r-bertopic")

## End(Not run)
```

`predict.bertopic_r` *Predict method for BERTopic models*

Description

Predict method for BERTopic models

Usage

```
## S3 method for class 'bertopic_r'
predict(
  object,
  newdata,
  type = c("both", "class", "prob"),
  embeddings = NULL,
  ...
)
```

Arguments

`object`	A "bertopic_r" model.
`newdata`	Character vector of new documents.
`type`	One of "class", "prob", or "both".
`embeddings`	Optional numeric matrix of embeddings.
`...`	Reserved for future arguments.

Value

Depending on `type`, an integer vector, a matrix/data frame, or a list.

`print.bertopic_r` *Print method for bertopic_r*

Description

Print method for bertopic_r

Usage

```
## S3 method for class 'bertopic_r'
print(x, ...)
```

Arguments

x	A "bertopic_r" object.
...	Unused.

---

set_bertopic_seed *Set random seed for R and Python backends*

Description

Set random seed for R and Python backends

Usage

```
set_bertopic_seed(seed)
```

Arguments

seed	Integer seed

---

sms_spam *SMS Spam Collection (UCI) - subset for examples*

Description

A cleaned subset of the UCI SMS Spam Collection, suitable for quick examples and tests in this package. Each row is an SMS message labeled as "ham" or "spam".

Usage

```
sms_spam
```

Format

A data frame with two columns:

label Character, either "ham" or "spam".

text Character, the SMS message content (UTF-8).

Note

This dataset is included for educational/demo purposes. If you use it in publications, please cite the original authors and the UCI repository page.

Source

UCI Machine Learning Repository: SMS Spam Collection. Dataset page: https://archive.ics.uci.edu/dataset/228/sms+spam+collection Original citation: Almeida, T.A., Hidalgo, J.M.G., & Yamakami, A. (2011). Contributions to the Study of SMS Spam Filtering: New Collection and Results.

Examples

```
data(sms_spam)
head(sms_spam)
```

`summary.bertopic_r` *Summary for BERTopic models*

Description

Summary for BERTopic models

Usage

```
## S3 method for class 'bertopic_r'
summary(object, ...)
```

Arguments

`object`	A "bertopic_r" model.
`...`	Unused.

Value

Invisibly returns a named list of summary fields.

`use_bertopic` *Bind current R session to the BERTopic environment (auto route)*

Description

If a Conda env with the given name exists, prefer Conda; otherwise try a virtualenv with the same name. Stops if neither exists.

Usage

```
use_bertopic(envname = "r-bertopic")
```

Arguments

`envname`	Character. Environment name. Default "r-bertopic".

Value

Invisibly, the Python executable path.

use_bertopic_condaenv *Bind current R session to a BERTopic Conda environment*

Description

Sets RETICULATE_PYTHON to the environment's Python and initializes **reticulate**. If **reticulate** is already initialized to a different Python, this stops with an informative error.

Usage

```
use_bertopic_condaenv(envname = "r-bertopic", required = TRUE)
```

Arguments

envname	Character. Conda env name (default "r-bertopic").
required	Logical. Kept for API symmetry; unused.

Value

Invisibly returns the Python executable path in the env.

Examples

```
## Not run:
use_bertopic_condaenv("r-bertopic")

## End(Not run)
```

use_bertopic_virtualenv *Bind current R session to a BERTopic virtualenv*

Description

Sets RETICULATE_PYTHON to the Python inside the given virtualenv and initializes **reticulate**. If **reticulate** is already initialized to a different Python, this stops with an informative error.

Usage

```
use_bertopic_virtualenv(envname = "r-bertopic", required = TRUE)
```

Arguments

envname	Character. Virtualenv name (default "r-bertopic").
required	Logical. Kept for API symmetry; unused.

Value

Invisibly returns the Python executable path in the venv.

Examples

```
## Not run:
use_bertopic_virtualenv("r-bertopic")

## End(Not run)
```

Index