

# Package: tokenizers.bpe (via r-universe)

October 11, 2024

**Type** Package

**Title** Byte Pair Encoding Text Tokenization

**Version** 0.1.3

**Maintainer** Jan Wijffels <jwijffels@bnosac.be>

**Description** Unsupervised text tokenizer focused on computational efficiency. Wraps the 'YouTokenToMe' library <<https://github.com/VKCOM/YouTokenToMe>> which is an implementation of fast Byte Pair Encoding (BPE) <<https://aclanthology.org/P16-1162/>>.

**URL** <https://github.com/bnosac/tokenizers.bpe>

**License** MPL-2.0

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.2

**Depends** R (>= 2.10)

**Imports** Rcpp (>= 0.11.5)

**LinkingTo** Rcpp

**Repository** <https://bnosac.r-universe.dev>

**RemoteUrl** <https://github.com/bnosac/tokenizers.bpe>

**RemoteRef** HEAD

**RemoteSha** 72ecec49fe51c2250807c3d0325e7ac87a6d8a10

## Contents

belgium_parliament . . . . .	2
bpe . . . . .	2
bpe_decode . . . . .	4
bpe_encode . . . . .	4
bpe_load_model . . . . .	5

<b>Index</b>	<b>7</b>
--------------	----------

---

belgium_parliament	<i>Dataset from 2017 with Questions asked in the Belgium Federal Parliament</i>
--------------------	---

---

### Description

Dataset from 2017 with Questions asked by members of the Belgian Federal Parliament. The dataset was extracted from <http://data.dekamer.be> and contains questions asked by persons in the Belgium Federal parliament. The questions are translated in Dutch and French.

The dataset contains the following information:

- doc\_id: an identifier
- text: the question itself
- language: the language of the text

### Source

<http://data.dekamer.be>, data is provided by <http://www.dekamer.be> in the public domain (CC0).

### Examples

```
data(belgium_parliament)
str(belgium_parliament)
```

---

bpe	<i>Construct a Byte Pair Encoding model</i>
-----	---

---

### Description

Construct a Byte Pair Encoding model on text

### Usage

```
bpe(
  x,
  coverage = 0.9999,
  vocab_size = 5000,
  threads = -1L,
  pad_id = 0L,
  unk_id = 1L,
  bos_id = 2L,
  eos_id = 3L,
  model_path = file.path(getwd(), "youtokentome.bpe")
)
```

## Arguments

x	path to the text file containing training data or a character vector of text with training data
coverage	fraction of characters covered by the model. Must be in the range [0, 1]. A good value to use is about 0.9999
vocab_size	integer indicating the number of tokens in the final vocabulary
threads	integer with number of CPU threads to use for model processing. If equal to -1 then minimum of the number of available threads and 8 will be used
pad_id	integer, reserved id for padding
unk_id	integer, reserved id for unknown symbols
bos_id	integer, reserved id for begin of sentence token
eos_id	integer, reserved id for end of sentence token
model_path	path to the file on disk where the model will be stored. Defaults to 'youtoken-tome.bpe' in the current working directory

## Value

an object of class `youtoken-tome` which is defined at [bpe\\_load\\_model](#)

## See Also

[bpe\\_load\\_model](#)

## Examples

```
data(belgium_parliament, package = "tokenizers.bpe")
x <- subset(belgium_parliament, language == "french")
model <- bpe(x$text, coverage = 0.999, vocab_size = 5000, threads = 1)
model
str(model$vocabulary)

text <- c("L'appartement est grand & vraiment bien situe en plein centre",
         "Proportion de femmes dans les situations de famille monoparentale.")
bpe_encode(model, x = text, type = "subwords")
bpe_encode(model, x = text, type = "ids")

encoded <- bpe_encode(model, x = text, type = "ids")
decoded <- bpe_decode(model, encoded)
decoded

## Remove the model file (Clean up for CRAN)
file.remove(model$model_path)
```

---

**bpe\_decode***Decode Byte Pair Encoding sequences to text*

---

**Description**

Decode a sequence of Byte Pair Encoding ids into text again

**Usage**

```
bpe_decode(model, x, ...)
```

**Arguments**

<code>model</code>	an object of class <code>youtokentome</code> as returned by <a href="#">bpe_load_model</a>
<code>x</code>	an integer vector of BPE id's
<code>...</code>	further arguments passed on to <code>youtokentome_encode_as_ids</code>

**Examples**

```
data(belgium_parliament, package = "tokenizers.bpe")
x <- subset(belgium_parliament, language == "french")
model <- bpe(x$text, coverage = 0.999, vocab_size = 5000, threads = 1)
model
str(model$vocabulary)

text <- c("L'appartement est grand & vraiment bien situe en plein centre",
         "Proportion de femmes dans les situations de famille monoparentale.")
bpe_encode(model, x = text, type = "subwords")
bpe_encode(model, x = text, type = "ids")

encoded <- bpe_encode(model, x = text, type = "ids")
decoded <- bpe_decode(model, encoded)
decoded

## Remove the model file (Clean up for CRAN)
file.remove(model$model_path)
```

---

**bpe\_encode***Tokenise text alongside a Byte Pair Encoding model*

---

**Description**

Tokenise text alongside a Byte Pair Encoding model

**Usage**

```
bpe_encode(
  model,
  x,
  type = c("subwords", "ids"),
  bos = FALSE,
  eos = FALSE,
  reverse = FALSE
)
```

**Arguments**

model	an object of class <code>youtokentome</code> as returned by <code>bpe_load_model</code>
x	a character vector of text to tokenise
type	a character string, either 'subwords' or 'ids' to get the subwords or the corresponding ids of these subwords as defined in the vocabulary of the model. Defaults to 'subwords'.
bos	logical if set to TRUE then token 'beginning of sentence' will be added
eos	logical if set to TRUE then token 'end of sentence' will be added
reverse	logical if set to TRUE the output sequence of tokens will be reversed

**Examples**

```
data(belgium_parliament, package = "tokenizers.bpe")
x <- subset(belgium_parliament, language == "french")
model <- bpe(x$text, coverage = 0.999, vocab_size = 5000, threads = 1)
model
str(model$vocabulary)

text <- c("L'appartement est grand & vraiment bien situe en plein centre",
         "Proportion de femmes dans les situations de famille monoparentale.")
bpe_encode(model, x = text, type = "subwords")
bpe_encode(model, x = text, type = "ids")

encoded <- bpe_encode(model, x = text, type = "ids")
decoded <- bpe_decode(model, encoded)
decoded

## Remove the model file (Clean up for CRAN)
file.remove(model$model_path)
```

---

bpe_load_model	<i>Load a Byte Pair Encoding model</i>
----------------	--

---

**Description**

Load a Byte Pair Encoding model trained with `bpe`

**Usage**

```
bpe_load_model(file, threads = -1L)
```

**Arguments**

file	path to the model
threads	integer with number of CPU threads to use for model processing. If equal to -1 then minimum of the number of available threads and 8 will be used

**Value**

an object of class `youtokentome` which is a list with elements

1. `model`: an Rcpp pointer to the model
2. `model_path`: the path to the model
3. `threads`: the threads argument
4. `vocab_size`: the size of the BPE vocabulary
5. `vocabulary`: the BPE vocabulary with is a `data.frame` with columns `id` and `subword`

**Examples**

```
## Reload a model
path <- system.file(package = "tokenizers.bpe", "extdata", "youtokentome.bpe")
model <- bpe_load_model(path)

## Build a model and load it again

data(belgium_parliament, package = "tokenizers.bpe")
x <- subset(belgium_parliament, language == "french")
model <- bpe(x$text, coverage = 0.999, vocab_size = 5000, threads = 1)
model <- bpe_load_model(model$model_path, threads = 1)

## Remove the model file (Clean up for CRAN)
file.remove(model$model_path)
```

# Index

belgium\_parliament, [2](#)  
bpe, [2](#), [5](#)  
bpe\_decode, [4](#)  
bpe\_encode, [4](#)  
bpe\_load\_model, [3-5](#), [5](#)