

Package ‘tokenizers.bpe’

August 2, 2019

Type Package

Title Byte Pair Encoding Text Tokenization

Version 0.1.0

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://www.aclweb.org/anthology/P16-1162>>.

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

License MPL-2.0

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Depends R (>= 2.10)

Imports Rcpp (>= 0.11.5)

LinkingTo Rcpp

NeedsCompilation yes

Author Jan Wijffels [aut, cre, cph] (R wrapper),
BNOSAC [cph] (R wrapper),
VK.com [cph],
Gregory Popovitch [ctb, cph] (Files at src/parallel_hashmap (Apache License, Version 2.0),
The Abseil Authors [ctb, cph] (Files at src/parallel_hashmap (Apache License, Version 2.0),
Ivan Belonogov [ctb, cph] (Files at src/youtokentome (MIT License))

Repository CRAN

Date/Publication 2019-08-02 09:40:02 UTC

R topics documented:

belgium_parliament	2
bpe	3
bpe_decode	4
bpe_encode	5
bpe_load_model	6

Index	7
--------------	----------

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 *Construct a Byte Pair Encoding model*

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 'youtokentome.bpe' in the current working directory

Value

an object of class `youtokentome` 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

model	an object of class <code>youtokentome</code> as returned by bpe_load_model
x	an integer vector of BPE id's
...	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

<code>model</code>	an object of class <code>youtokentome</code> as returned by bpe_load_model
<code>x</code>	a character vector of text to tokenise
<code>type</code>	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'.
<code>bos</code>	logical if set to TRUE then token 'beginning of sentence' will be added
<code>eos</code>	logical if set to TRUE then token 'end of sentence' will be added
<code>reverse</code>	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, [3](#), [6](#)
bpe_decode, [4](#)
bpe_encode, [5](#)
bpe_load_model, [3-5](#), [6](#)