Package: wordsalad (via r-universe)

September 8, 2024

Title Provide Tools to Extract and Analyze Word Vectors

Version 0.2.0.9000

Description Provides access to various word embedding methods (GloVe, fasttext and word2vec) to extract word vectors using a unified framework to increase reproducibility and correctness.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Depends R (>= 2.10)

Imports tibble, text2vec, word2vec, fastTextR

Suggests testthat

URL https://github.com/EmilHvitfeldt/wordsalad

BugReports https://github.com/EmilHvitfeldt/wordsalad/issues

Repository https://emilhvitfeldt.r-universe.dev

RemoteUrl https://github.com/emilhvitfeldt/wordsalad

RemoteRef HEAD

RemoteSha 7931d9ac20f1003cb91dd0176a8c476c93eff5cc

Contents

fairy_tales *The text of H.C. Andersen's fairy tales in English*

Description

A dataset containing 5 of H.C. andersens fairy tales translated to English. The UTF-8 plain text was sourced from http://www.andersenstories.com/.

Usage

```
fairy_tales
```

Format

A character vector with 5 elements.

Details

This is not representive of the size needed to generate good word vectors. It is just used for examples.

fasttext *Extract word vectors from fasttext word embedding*

Description

The calculations are done with the fastTextR package.

Usage

```
fasttext(
  text,
  tokenizer = text2vec::space_tokenizer,
  dim = 10L,
  type = c("skip-gram", "cbow"),
  window = 5L,
  loss = "hs",
  negative = 5L,
  n_iter = 5L,
  min_count = 5L,
  threads = 1L,
  composition = c("tibble", "data.frame", "matrix"),
  verbose = FALSE
)
```

Arguments

`text`	Character string.
`tokenizer`	Function, function to perform tokenization. Defaults to text2vec::space_tokenizer.
`dim`	Integer, number of dimension of the resulting word vectors.
`type`	Character, the type of algorithm to use, either 'cbow' or 'skip-gram'. Defaults to 'skip-gram'.
`window`	Integer, skip length between words. Defaults to 5.
`loss`	Charcter, choice of loss function must be one of "ns", "hs", or "softmax". See details for more Defaults to "hs".
`negative`	integer with the number of negative samples. Only used when loss = "ns".
`n_iter`	Integer, number of training iterations. Defaults to 5. `numeric = -1` defines early stopping strategy. Stop fitting when one of two following conditions will be satisfied: (a) passed all iterations (b) `cost_previous_iter / cost_current_iter - 1 < convergence_tol`. Defaults to -1.
`min_count`	Integer, number of times a token should appear to be considered in the model. Defaults to 5.
`threads`	number of CPU threads to use. Defaults to 1.
`composition`	Character, Either "tibble", "matrix", or "data.frame" for the format out the resulting word vectors.
`verbose`	Logical, controls whether progress is reported as operations are executed.

Details

The choice of loss functions are one of:

- "ns" negative sampling
- "hs" hierarchical softmax
- "softmax" full softmax

Value

A tibble, data.frame or matrix containing the token in the first column and word vectors in the remaining columns.

Source

https://fasttext.cc/

References

Enriching Word Vectors with Subword Information, 2016, P. Bojanowski, E. Grave, A. Joulin, T. Mikolov.

Examples

```
fasttext(fairy_tales, n_iter = 2)

# Custom tokenizer that splits on non-alphanumeric characters
fasttext(fairy_tales,
         n_iter = 2,
         tokenizer = function(x) strsplit(x, "[^[:alnum:]]+"))
```

`glove` *Extract word vectors from GloVe word embedding*

Description

The calculations are done with the text2vec package.

Usage

```
glove(
  text,
  tokenizer = text2vec::space_tokenizer,
  dim = 10L,
  window = 5L,
  min_count = 5L,
  n_iter = 10L,
  x_max = 10L,
  stopwords = character(),
  convergence_tol = -1,
  threads = 1,
  composition = c("tibble", "data.frame", "matrix"),
  verbose = FALSE
)
```

Arguments

`text`	Character string.
`tokenizer`	Function, function to perform tokenization. Defaults to text2vec::space_tokenizer.
`dim`	Integer, number of dimension of the resulting word vectors.
`window`	Integer, skip length between words. Defaults to 5.
`min_count`	Integer, number of times a token should appear to be considered in the model. Defaults to 5.
`n_iter`	Integer, number of training iterations. Defaults to 10.
`x_max`	Integer, maximum number of co-occurrences to use in the weighting function. Defaults to 10.
`stopwords`	Character, a vector of stop words to exclude from training.

convergence_tol
: Numeric, value determining the convergence criteria. `numeric = -1` defines early stopping strategy. Stop fitting when one of two following conditions will be satisfied: (a) passed all iterations (b) `cost_previous_iter / cost_current_iter - 1 < convergence_tol`. Defaults to -1.

threads
: number of CPU threads to use. Defaults to 1.

composition
: Character, Either "tibble", "matrix", or "data.frame" for the format out the resulting word vectors.

verbose
: Logical, controls whether progress is reported as operations are executed.

Value

A tibble, data.frame or matrix containing the token in the first column and word vectors in the remaining columns.

Source

https://nlp.stanford.edu/projects/glove/

References

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation.

Examples

```
glove(fairy_tales, x_max = 5)
```

---

word2vec *Extract word vectors from word2vec word embedding*

Description

The calculations are done with the word2vec package.

Usage

```
word2vec(
  text,
  tokenizer = text2vec::space_tokenizer,
  dim = 50,
  type = c("cbow", "skip-gram"),
  window = 5L,
  min_count = 5L,
  loss = c("ns", "hs"),
  negative = 5L,
  n_iter = 5L,
```

```
  lr = 0.05,
  sample = 0.001,
  stopwords = character(),
  threads = 1L,
  collapse_character = "\t",
  composition = c("tibble", "data.frame", "matrix")
)
```

Arguments

text	Character string.
tokenizer	Function, function to perform tokenization. Defaults to text2vec::space_tokenizer.
dim	dimension of the word vectors. Defaults to 50.
type	the type of algorithm to use, either 'cbow' or 'skip-gram'. Defaults to 'cbow'
window	skip length between words. Defaults to 5.
min_count	integer indicating the number of time a word should occur to be considered as part of the training vocabulary. Defaults to 5.
loss	Charcter, choice of loss function must be one of "ns" or "hs". See detaulsfor more Defaults to "ns".
negative	integer with the number of negative samples. Only used in case hs is set to FALSE
n_iter	Integer, number of training iterations. Defaults to 5.
lr	initial learning rate also known as alpha. Defaults to 0.05
sample	threshold for occurrence of words. Defaults to 0.001
stopwords	a character vector of stopwords to exclude from training
threads	number of CPU threads to use. Defaults to 1.
collapse_character	Character vector with length 1. Character used to glue together tokens after tokenizing. See details for more information. Defaults to "\t".
composition	Character, Either "tibble", "matrix", or "data.frame" for the format out the resulting word vectors.

Details

A trade-off have been made to allow for an arbitrary tokenizing function. The text is first passed through the tokenizer. Then it is being collapsed back together into strings using collapse_character as the separator. You need to pick collapse_character to be a character that will not appear in any of the tokens after tokenizing is done. The default value is a "tab" character. If you pick a character that is present in the tokens then those words will be split.

The choice of loss functions are one of:

- "ns" negative sampling
- "hs" hierarchical softmax

Value

A tibble, data.frame or matrix containing the token in the first column and word vectors in the remaining columns.

Source

https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compos
pdf

References

Mikolov, Tomas and Sutskever, Ilya and Chen, Kai and Corrado, Greg S and Dean, Jeff. 2013. Distributed Representations of Words and Phrases and their Compositionality

Examples

```
word2vec(fairy_tales)

# Custom tokenizer that splits on non-alphanumeric characters
word2vec(fairy_tales, tokenizer = function(x) strsplit(x, "[^[:alnum:]]+"))
```

Index