Package 'RtextSummary'

June 7, 2019

Type Package

Title Summarizes Text by Extracting Relevant Sentences

Version 0.1.0

Description Build a text summary by extracting relevant sentences from your text. The training dataset should consist of several documents, each document should have sentences separated by a period. While fitting the model, the 'term frequency - inverse document frequency' (TF-IDF) matrix that reflects how important a word is to a document is calculated first. Then vector representations for words are obtained from the 'global vectors for word representation' algorithm (GloVe). While applying the model on new data, the GloVe word vectors for each word are weighted by their TF-IDF weights and averaged to give a sentence vector or a document vector. The magnitude of this sentence vector gives the importance of that sentence within the document. Another way to obtain the importance of the sentence is to calculate cosine similarity between the sentence vector and the document vector. The output can either be at the sentence level (sentences and weights are returned) or at a document level (the summary for each document is returned). It is useful to first get a sentence level output and get quantiles of the sentence weights to determine a cutoff threshold for the weights. This threshold can then be used in the document level output. This method is a variation of the TF-IDF extractive summarization method mentioned in a review paper by Gupta (2010) <doi:10.4304/jetwi.2.3.258-268>.

License GPL-3

Encoding UTF-8

LazyData false

Imports R6, mlapi, stringr, tidyr, tokenizers, text2vec, Matrix.utils, dplyr

Depends R (>= 2.10)

RoxygenNote 6.1.0

NeedsCompilation no

Author Suryavanshi Abhijit [aut, cre]

Maintainer Suryavanshi Abhijit <abhi.surya@gmail.com>

Repository CRAN

Date/Publication 2019-06-07 08:30:03 UTC

R topics documented:

`opinosis` *opinosis dataset of 51 user reviews with topics and five summaries. each topic contains several sentences from different user reviews*

Description

opinosis dataset of 51 user reviews with topics and five summaries. each topic contains several sentences from different user reviews

Usage

```
data("opinosis")
```

Format

A data.frame:

a data.frame of 51 topics and summaries

References

http://kavita-ganesan.com/opinosis-opinion-dataset/#.XOyPdYhKhPZ

`stopwords_longlist` *a very long list of english stopwords*

Description

caution: check the list to make sure any domain-specific words are not being used as stopwords

Usage

```
data("stopwords_longlist")
```

Format

A character vector:

a vector of stopwords

TextSummary *TextSummary*

Description

Build a text summary by extracting relevant sentences from your text. The training dataset should consist of several documents, each document should have sentences separated by a period. While fitting the model, the 'term frequency - inverse document frequency' (TF-IDF) matrix that reflects how important a word is to a document is calculated first. Then vector representations for words are obtained from the 'global vectors for word representation' algorithm (GloVe). While applying the model on new data, the GloVe word vectors for each word are weighted by their TF-IDF weights and averaged to give a sentence vector or a document vector. The magnitude of this sentence vector gives the importance of that sentence within the document. Another way to obtain the importance of the sentence is to calculate cosine similarity between the sentence vector and the document vector. The output can either be at the sentence level (sentences and weights are returned) or at a document level (the summary for each document is returned). It is useful to first get a sentence level output and get quantiles of the sentence weights to determine a cutoff threshold for the weights. This threshold can then be used in the document level output. This method is a variation of the TF-IDF extractive summarization method mentioned in a review paper by Gupta (2010) <doi:10.4304/jetwi.2.3.258-268>.

Usage

```
TextSummary
```

Format

R6Class object.

Usage

For usage details see **Methods, Arguments and Examples** sections.

```
TextSummaryModel <- TextSummary$new( stopword_list )

TextSummaryModel$fit(x)

TextSummaryModel$transform(df,doc_id, txt_col,summary_col,topN=3,weight_threshold=10, return_sentenc
```

Methods

`$new( stopword_list )` Creates TextSummary model

`$fit(x)` fit model to an input vector (dataframe column) of documents

`$transform(df,doc_id, txt_col,summary_col,weight_method = c('Magnitude', 'DocSimilarity'),topN=3,we`
transform new data `df` using the model built on train data

Arguments

TextSummaryModel A `TextSummary` object

x An input vector (dataframe column) of documents, preprocessed as necessary to remove case, puncutation etc (except periods that indicate sentence boundaries)

df dataframe containing document ids and documents. Any other columns are passed through without any changes

doc_id column name that contains the document ids

txt_col column name that contains the document text

summary_col column name for the output summary. This column will be added to `df`

weight_method specifies how the sentences importance is calculated. `weight_method = "Magnitude"` gives the weights as the magnitude of the sentence vector. If `avg_weight_by_word_count = TRUE` then the magnitude is divided by the word count, which typically favors shorter sentences. If `avg_weight_by_word_count = FALSE` then the magnitude of the sentence vector is returned, which typically favors longer sentences. `weight_method = "DocSimilarity"` calculates the sentence importance as a cosine similarity between the sentence vector and the document vector. `avg_weight_by_word_count` does not play a role in the "DocSimilarity" method

topN top N sentences to keep in the output

weight_threshold threshold above which sentences are considered for inclusion in the summary

return_sentences `TRUE`: returns sentences and their weights. `topN`, `weight_threshold` and `replace_char` are ignored `FALSE`: `topN` sentences that have weights above `weight_threshold` are included in the summary.

replace_char The irrelevant sentences are replaced by `replace_char` (use `replace_char = ""` to completely remove the irrelevant sentences)

avg_weight_by_word_count if `TRUE`: the sentence weights are divided by number of words in the sentence.

Examples

```
{
library(RtextSummary)
library(stringr)
library(tidyr)
library(dplyr)

data("opinosis")

# the data is reduced to pass CRAN checks of <5 sec run-time
# delete the line below to build the model on the entire dataset
opinosis = opinosis[1:2,]%>%mutate(text = substr(text, 0, 10) )

# 'stopwords_longlist' is a very long list of stopwords.
# it is not used in this example but can be useful for other datasets
data("stopwords_longlist")

opinosis$text = stringr::str_replace_all(
  stringr::str_to_lower(opinosis$text),'[^a-z. ]','' )
```

```
# -- the model will be fit at the sentence level, which works well for this dataset
# for other datasets, also try fitting at the document level by commenting out the two lines below
tempdf = opinosis%>%
  tidyr::separate_rows(text, sep = '\\.')
# ----------------------------------------

summary.model = TextSummary$new( stopword_list = c() )
summary.model$fit(tempdf$text)

# the parameters below work well for this dataset.
# For other datasets, try changing weight_method and avg_weight_by_word_count
df_sentence_level = summary.model$transform(
  opinosis,
  doc_id = 'topics',
  txt_col = 'text',
  summary_col = 'summary',
  weight_method = 'Magnitude',
  return_sentences = TRUE,
  avg_weight_by_word_count = TRUE
)

# explore weight thresholds
quantile(df_sentence_level$wt, seq(0,1,0.1))


df_summary = summary.model$transform(
  opinosis,
  doc_id = 'topics',
  txt_col = 'text',
  summary_col = 'summary',
  weight_method = 'Magnitude',
  topN = 1,
  weight_threshold=quantile(df_sentence_level$wt, 0.3 ),
  return_sentences = FALSE,
  replace_char = '',
  avg_weight_by_word_count = TRUE
)
}
```

Index