Package 'phm'

July 23, 2025

Type Package

Title Phrase Mining

Version 1.1.5

Date 2025-05-16

Description Functions to extract and handle commonly occurring principal phrases obtained from collections of texts. This package is based on, Small, E., & Cabrera, J. (2025). Principal phrase mining, an automated method for extracting meaningful phrases from text. International Journal of Computers and Applications, 47(1), 84–92.

License GPL-3

Encoding UTF-8

Imports data.table (>= 1.14.2), tm (>= 0.7-8), shiny (>= 1.7.1), Matrix (>= 1.4-1), smallstuff (>= 1.0.1), NLP (>= 0.2-1)

RoxygenNote 7.3.2

Suggests knitr, rmarkdown

VignetteBuilder knitr

URL `https://doi.org/10.1080/1206212X.2024.2448494`

NeedsCompilation no

Author Ellie Small [aut, cre] (ORCID: <`https://orcid.org/0000-0003-1313-115X`>)

Maintainer Ellie Small <`ellie_small@yahoo.com`>

Repository CRAN

Date/Publication 2025-05-16 23:50:02 UTC

Contents

`as.matrix.phraseDoc` *Convert a phraseDoc Object to a Matrix*

Description

Convert a phraseDoc Object to a Matrix

Usage

```
## S3 method for class 'phraseDoc'
as.matrix(x, ids = TRUE, sparse = FALSE, ...)
```

Arguments

`x`	A phraseDoc object.
`ids`	A logical value with TRUE (default) to use ids (if available), FALSE to use indices
`sparse`	A logical value indicates whether a sparse matrix should be returned (default FALSE)
`...`	Additional arguments

Value

A matrix with phrases as rows, texts as columns, and elements containing the number of times the phrase occurs in the text

Examples

```
tst=c("This is a test text",
      "This is a test text 2",
      "This is another test text",
      "This is another test text 2",
      "This girl will test text that man",
      "This boy will test text that man")
pd=phraseDoc(tst)
as.matrix(pd)
```

---

bestDocs *Find Informative Documents in a Corpus*

Description

Find the documents in a corpus that have the most high frequency phrases and return a corpus with just those documents

Usage

```
bestDocs(co, num = 3L, n = 10L, pd = NULL)
```

Arguments

co	A corpus with documents
num	Integer with the number of documents to return
n	Integer with the number of high frequency phrases to use
pd	phraseDoc object for the corpus in co; if NULL, a phraseDoc will be created for it.

Value

A corpus with the num documents that have the most high frequency phrases, in order of the number of high frequency phrases. The corpus returned will have the meta field oldIdx set to the index of the document in the original corpus, and the meta field hfPhrases to the number of high frequency phrases it contains.

Examples

```
v1=c("Here is some text to test phrase mining","phrase mining is fun",
  "Some text is better than no text","No text, no phrase mining")
co=tm::VCorpus(tm::VectorSource(v1))
pd=phraseDoc(co,min.freq=2)
bestDocs(co,2,2,pd)
```

`canberra` *Calculate Canberra Distance*

Description

When two vectors are given, this calculates the Canberra distance between them; This is calculated as the sum of the absolute difference between corresponding elements divided by the sum of their absolute values, for elements that are not both zero only.

Usage

```
canberra(x, y)
```

Arguments

`x`	A numeric vector
`y`	A numeric vector of the same dimension as x

Value

The Canberra distance between x and y. For example, between vectors (1,2,0) and (0,1,1), for position 1 we have (1-0)/1, for position 2 we have (2-1)/3, and for position 3 we have abs(0-1)/1, added together this results in 2 1/3, or 2.33. Note that a text distance of zero indicates that the two vectors are equal, while a text distance of 1 indicates that they have no terms in common.

Examples

```
canberra(c(1,2,0),c(0,1,1))
```

`DFSource` *Create a DFSource object from a data frame*

Description

This function will create a DFSource object from a data frame that contains at least columns id and text, but may contain several more. VCorpus will use this to read in each row from the data frame into a PlainTextDocument, storing additional variables in its metadata. It will then combine all those PlainTextDocuments in a VCorpus object.

Usage

```
DFSource(x)
```

Arguments

`x`	A dataframe with at a minimum a text and id column, and a row for each document to be stored in a corpus.

Value

A DFSource object containing the encoding set to "", the number of rows (length), the current position (position=0), the type of reader to use (reader=readDF), and the content (`x`).

Examples

```
(df=data.frame(id=1:3,text=c("First text","Second text","Third text"),
               title=c("N1","N2","N3")))
DFSource(df)
```

`distMatrix` *Calculate a Distance Matrix*

Description

Calculate a distance matrix for a numeric matrix, where a distance function is used to calculate the distance between all combinations of the columns of the matrix `M`.

Usage

```
distMatrix(M, fn = "textDist", ...)
```

Arguments

`M`	A numeric matrix
`fn`	The name of a distance function, default is "textDist".
`...`	Additional arguments to be passed to the distance function

Value

The distance matrix with the distance between all combinations of the columns of `M` according to the distance function in `fn`.

Examples

```
M=matrix(c(0,1,0,2,0,10,0,14,12,0,8,0,1,0,1,0),4)
colnames(M)=1:4;rownames(M)=c("A","B","C","D")
M
#Text distance matrix
distMatrix(M)
#Canberra distance matrix
distMatrix(M,"canberra")
```

`freqPhrases` *Display Frequent Principal Phrases*

Description

Display the most frequent principal phrases in a phraseDoc object.

Usage

```
freqPhrases(pd, n = 10)
```

Arguments

`pd`	A phraseDoc object.
`n`	Number of principal phrases to display.

Value

A vector with the `n` most frequent principal phrases and their frequencies.

Examples

```
tst=c("This is a test text",
      "This is a test text 2",
      "This is another test text",
      "This is another test text 2",
      "This girl will test text that man",
      "This boy will test text that man")
pd=phraseDoc(tst)
freqPhrases(pd, 2)
```

`getDocs` *Display Frequency Matrix for Phrases*

Description

Display a frequency matrix containing all the documents that contain any of the phrases in phrs and the number of times they occur in that document.

Usage

```
getDocs(pd, phrs, ids = TRUE)
```

Arguments

`pd`	A phraseDoc object.
`phrs`	A set of phrases.
`ids`	A logical value with TRUE (default) to return ids (if available), FALSE to return indices.

Value

A matrix with the documents and # of occurrences for the phrases in phrs.

Examples

```
tst=c("This is a test text",
      "This is a test text 2",
      "This is another test text",
      "This is another test text 2",
      "This girl will test text that man",
      "This boy will test text that man")
pd=phraseDoc(tst)
getDocs(pd, c("test text","another test text"))
```

`getElem.DFSource` *Obtain the current row of the content of a DFSource*

Description

Using the position field of x to indicate the index of the current row, we retrieve the current row of the content of a DFSource. This function is mainly used by the VCorpus function.

Usage

```
## S3 method for class 'DFSource'
getElem(x)
```

Arguments

`x`	A DFSource object

Value

A list with the current row in the content of a DFSource object. The current row index is the position in the DFSource object.

Examples

```
library(tm)
df=data.frame(id=1:3,text=c("First text","Second text","Third text"),
              title=c("N1","N2","N3"))
getElem(stepNext(DFSource(df)))
```

getPhrases *Display Frequency Matrix for Documents*

Description

Display a frequency matrix containing all the documents for which the indices are given in docs with their principal phrases and the number of times they occur in each document.

Usage

```
getPhrases(pd, doc, ids = TRUE)
```

Arguments

pd	A phraseDoc object
doc	An integer vector containing indices of documents, or a character vector containing the ids of documents (column names)
ids	A logical value with TRUE (default) to return ids (if available), FALSE to return indices

Value

A matrix with the documents and # of occurrences of principal phrases for the documents in `docs`

Examples

```
tst=c("This is a test text",
      "This is a test text 2",
      "This is another test text",
      "This is another test text 2",
      "This girl will test text that man",
      "This boy will test text that man")
pd=phraseDoc(tst)
getPhrases(pd, c(1,3))
```

getPubMed *Create a data table from a text file in PubMed format*

Description

This function takes as input a file produced via PubMed in PubMed format and outputs a data frame with the id equal to the PMID, text equal to the abstract, date, title, and author for each publication in the file.

Usage

```
getPubMed(file)
```

Arguments

`file`	path to the PubMed file

Value

A data table with a row for each publication holding the id equal to the PMID, text equal to the abstract, date, title, and author for that publication.

Examples

```
#Go to Pubmed and enter search criteria, save the result to PubMed format.
#If the file is called pubmed_result.txt and located in the current
#directory:
#PM=getPubMed("pubmed_result.txt")
#Will load the data from the search into a data table called PM
```

`phraseDoc` *phraseDoc Creation*

Description

Create an object of class phraseDoc. This will hold all principal phrases of a collection of texts that occur a minimum number of times, plus the texts they occur in and their position within those texts.

Usage

```
phraseDoc(
  co,
  mn = 2,
  mx = 8,
  ssw = stopStartWords(),
  sew = stopEndWords(),
  sp = stopPhrases(),
  min.freq = 2,
  principal = function(phrase, freq) {
     freq >= min.freq
 },
  max.phrases = 1500,
  shiny = FALSE,
  silent = FALSE
)
```

Arguments

`co`	A corpus or a character vector with each element the text of a document.
`mn`	Minimum number of words in a phrase.
`mx`	Maximum number of words in a phrase.

`ssw`	A set of words no phrase should start with.
`sew`	A set of words no phrase should end with.
`sp`	A set of phrases to be excluded.
`min.freq`	The minimum frequency of phrases to be included.
`principal`	Function that determines if a phrase is a principal phrase. By default, FALSE is returned if the phrase occurs less often than the number in `min.freq`.
`max.phrases`	Maximum number of phrases to be included.
`shiny`	TRUE if called from a shiny program. This will allow progress to be recorded on a progress meter; the function uses about 100 progress steps, so it should be created inside a withProgress function with the argument max set to at least 100.
`silent`	TRUE if you do not want progress messages.

Value

Object of class phraseDoc

Examples

```
tst=c("This is a test text",
      "This is a test text 2",
      "This is another test text",
      "This is another test text 2",
      "This girl will test text that man",
      "This boy will test text that man")
phraseDoc(tst)
```

`print.phraseDoc` *Print a phraseDoc Object*

Description

Print a phraseDoc Object

Usage

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

Arguments

`x`	Object of type phraseDoc
`...`	Additional arguments

Examples

```
tst=c("This is a test text",
      "This is a test text 2",
      "This is another test text",
      "This is another test text 2",
      "This girl will test text that man",
      "This boy will test text that man")
(pd=phraseDoc(tst))
```

---

print.textCluster *Print a textCluster Object*

Description

Print a textCluster Object

Usage

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

Arguments

`x`	Object of type textCluster
`...`	Additional arguments

Value

The total number of clusters and total number of documents are printed. There is no return value.

Examples

```
M=matrix(c(0,1,0,2,0,10,0,14,12,0,8,0,1,0,1,0),4)
colnames(M)=1:4;rownames(M)=c("A","B","C","D")
tc=textCluster(M,2)
tc
```

`readDF` *Create a PlainTextDocument from a row in a data frame*

Description

Read a row of the content of a DFSource object into a PlainTextDocument.

Usage

```
readDF(elem, language, id = "1")
```

Arguments

`elem`	A list containing the field content containing one row with data from a data frame containing at least the columns id and text, but possibly more.
`language`	abbreviation of the language used; "en" for English
`id`	Not used, but needed for VCorpus

Value

A PlainTextDocument with content equal to the contents of the text field, and meta data containing the information in the remaining fields, including the id field

Examples

```
(df=data.frame(id=1:3,text=c("First text","Second text","Third text"),
               title=c("N1","N2","N3")))
readDF(list(content=df[1,]),"en")
```

`removePhrases` *Remove Phrases from phraseDoc Object*

Description

Remove a set of phrases from a phraseDoc object.

Usage

```
removePhrases(pd, phrs)
```

Arguments

`pd`	A phraseDoc object.
`phrs`	A set of phrases.

Value

A phraseDoc object with the phrases in `phrs` removed.

Examples

```
tst=c("This is a test text",
      "This is a test text 2",
      "This is another test text",
      "This is another test text 2",
      "This girl will test text that man",
      "This boy will test text that man")
pd=phraseDoc(tst)
removePhrases(pd, c("test text","another test text"))
```

`showCluster` *Show Cluster Contents*

Description

Show all documents and their non-zero terms in a cluster, with the terms first ordered by highest number of documents the term appears in, then total frequency.

Usage

```
showCluster(tdm, clust, cl, n = 10L)
```

Arguments

`tdm`	A term frequency matrix.
`clust`	A vector indicating for each column in `tdm` what cluster they belong to
`cl`	Cluster number
`n`	Integer showing the maximum number of terms to be returned (default 10)

Value

A matrix with document names of `tdm` on the columns and terms on the rows for all columns in the cluster, where terms that appear in the most documents (columns), and within that have the highest frequency in the cluster, are shown first. Two columns are added at the end of the matrix with the the number of documents each term appears in and its total frequency in the cluster. The number of terms displayed equals the number in `n`, or less if there are less terms in the cluster. If there are no terms at all in the cluster, a list is output with the items docs and note, where docs is a vector with all document names of documents in the cluster, and the note stating that the cluster has no terms.

Examples

```
M=matrix(c(0,1,0,2,0,10,0,14,12,0,8,0,1,0,1,0),4)
colnames(M)=1:4;rownames(M)=c("A","B","C","D")
tc=textCluster(M,2)
showCluster(M,tc$cluster,1)
```

`stopEndWords` *Words that Principal Phrases do not End with*

Description

Create a vector with words that principal phrases should not end with.

Usage

```
stopEndWords()
```

Value

vector with words

Examples

```
stopEndWords()
```

`stopPhrases` *Phrases that are not Principal Phrases*

Description

Create a vector with phrases that are not principal phrases.

Usage

```
stopPhrases()
```

Value

vector with phrases

Examples

```
stopPhrases()
```

`stopStartWords` *Words that Principal Phrases do not Start with*

Description

Create a vector with words that principal phrases should not start with.

Usage

```
stopStartWords()
```

Value

vector with words

Examples

```
stopStartWords()
```

`textCluster` *Cluster a Term-Document Matrix*

Description

Combine documents (columns) into k clusters that have texts that are most similar based on their text distance. Documents with no terms are assigned to the last cluster.

Usage

```
textCluster(tdm, k, mx = 100, md = 5 * k)
```

Arguments

`tdm`	A term document matrix with terms on the rows and documents on the columns.
`k`	A positive integer with the number of clusters needed
`mx`	Maximum number of times to iterate (default 100)
`md`	Maximum number of documents to use for the initial setup (default 5*`k`).

Value

A textcluster object with three items; cluster, centroids, and size, where cluster contains a vector indicating for each column in `M` what cluster they have been assigned to, centroids contains a matrix with each column the centroid of a cluster, and size a named vector with the size of each cluster.

Examples

```
M=matrix(c(0,1,0,2,0,10,0,14,12,0,8,0,1,0,1,0),4)
colnames(M)=1:4;rownames(M)=c("A","B","C","D")
textCluster(M,2)
```

`textDist` *Calculate Text Distance*

Description

When two vectors are given, this calculates the text distance between them; text distance is calculated as the proportion of unmatched frequencies, i.e., the number of unmatched frequencies divided by the total frequencies among the two vectors. However, if neither vector has any values at all, their distance equals the number provided in the zeroes argument, which is .5 by default. When two matrices are given, the text distance between corresponding columns is calculated.

Usage

```
textDist(x, y, zeroes = 0.5)
```

Arguments

`x`	A numeric vector or matrix
`y`	A numeric vector or matrix of the same dimension as x
`zeroes`	Text distance when both vectors are zero vectors; default is .5

Value

When x and y are vectors, the text distance between them. For example, between vectors (1,2,0) and (0,1,1), a total of 5 frequencies are present. However, position 1 matches nothing when it could have matched 1 frequency, position 2 matches 1 frequency when it could have matched both positions, so 1 remains unmatched. Position 3 matches nothing when it could have matched 1. So we have 3 unmatched positions divided by 5 frequencies, resulting in a text distance of 3/5=.6. If x and y are matrices, a vector with the text distance between corresponding columns is returned. So for two 4x2 matrices, a vector with two values is returned, one with the text distance between the first columns of the matrices, and the second one with the text distance between the second columns of the matrices. For large sets of data, it is recommended to use matrices as it is much more efficient than calculating column by column.

Examples

```
#text distance between two vectors
textDist(c(1,2,0),c(0,1,1))
(M1=matrix(c(0,1,0,2,0,10,0,14),4))
(M2=matrix(c(12,0,8,0,1,3,1,2),4))
#text distance between corresponding columns of M1 and M2
textDist(M1,M2)
```

`textDistMatrix` *Calculate a Text Distance Matrix*

Description

Calculate a distance matrix for a numeric matrix, using the textDist function. It is used to calculate the text distance between all combinations of the columns of the matrix `M`.

Usage

```
textDistMatrix(M, zeroes = 0.5)
```

Arguments

`M`	A numeric matrix
`zeroes`	Text distance when both vectors are zero vectors; default is .5

Value

The text distance matrix with the text distance between all combinations of the columns of `M`. This will give the same result as the function distMatrix when run with its default distance function "textDist"; however, for large matrices textDistMatrix is much more efficient. In addition, for very large matrices distMatrix may not run, while textDistMatrix will.

Examples

```
M=matrix(c(0,1,0,2,0,10,0,14,12,0,8,0,1,0,1,0),4)
colnames(M)=1:4;rownames(M)=c("A","B","C","D")
M
#Text distance matrix
textDistMatrix(M)
```

Index