Package 'RNewsflow'

July 29, 2019

Type Package

Title Tools for Comparing Text Messages Across Time and Media

Version 1.1.1

Date 2019-07-27

Author Kasper Welbers & Wouter van Atteveldt

Maintainer Kasper Welbers <kasperwelbers@gmail.com>

Description A collection of tools for measuring the similarity of text messages and tracing the flow of messages over
time and across media.

License MIT + file LICENSE

Depends R (>= 3.2.0), igraph, tm, Matrix (>= 1.2)

Imports slam, stringi, scales, wordcloud, data.table, methods, quanteda, Rcpp (>= 0.12.12)

LinkingTo Rcpp, RcppEigen, RcppProgress

LazyData true

SystemRequirements C++11

RoxygenNote 6.1.1

Suggests knitr, rmarkdown

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-07-29 10:20:05 UTC

R topics documented:

create_queries 2
delete.duplicates 3
directed.network.plot 5

docnet 6
document.network 6
document.network.plot 8
documents.compare 9
filter.window 10
get_doc_terms 11
get_overlap_terms 12
hourdiff_range_thresholds 13
network.aggregate 13
newsflow.compare 15
only.first.match 17
rnewsflow_dfm 18
show.window 18
tcrossprod_sparse 19
term.day.dist 21
term_char_sim 22
term_innovation 23
term_intersect 24
term_union 25

`create_queries` *(experimental) Automatically infer queries from combinations of terms in a dtm*

Description

Prepares query terms with high sparsity. Returns two matrices: a query and lookup dtm. Can either be used with one dtm as input (which becomes both the query and lookup dtm) or with a dtm and ref_dtm (reference), in which case dtm represents the queries and ref_dtm the lookup dtm.

Usage

```
create_queries(dtm, ref_dtm = NULL, min_docfreq = 2,
  max_docprob = 0.001, weight = c("tfidf", "binary"),
  norm_weight = c("max", "doc_max", "dtm_max", "none"),
  min_obs_exp = NA, union_sim_thres = NA, union_sim_thres2 = NA,
  combine_all = T, only_dtm_combs = T, use_dtm_and_ref = T,
  verbose = F)
```

Arguments

`dtm`	A quanteda dfm
`ref_dtm`	Optionally, another quanteda dfm. If given, the ref_dtm will be used to calculate the docfreq/docprob scores.
`min_docfreq`	The minimum frequency for terms or combinations of terms
`max_docprob`	The maximum probability (document frequency / N) for terms or combinations of terms

`weight` Determine how to weight the queries (if ref_dtm is used, uses the idf of the ref_dtm). Default is "binary" (does/does not occur). "tfidf" uses common tf-idf weighting (actually just idf, since scores are binary). The ref_dfm will always be binary.

`norm_weight` Normalize the weight score so that the highest value is 1. If "max" is used, max is the highest possible value. "doc_max" uses the highest value within each document, and "dtm_max" uses the highest observed value in the dtm.

`min_obs_exp` The minimum ratio of the observed and expected frequency of a term combination

`union_sim_thres`
If given, a number between 0 and 1, used as the cosine similarity threshold for combining clusters of terms

`union_sim_thres2`
Like union_sim_thres, but after combining terms

`combine_all` If True, combine all terms. If False (default), terms that are included as unigrams (i.e. that are within the min_docfreq and max_docprob) are not combined with other terms.

`only_dtm_combs`
Only include term combinations that occur in dtm. This makes sense (and saves a lot of memory) if you are only interested in assymetric similarity measures based on the query

`use_dtm_and_ref`
if a ref_dtm is used, both the dtm and ref_dtm are used to compute the docfreq and docprob values used for filtering and weighting. If use_dtm_and_ref is set o FALSE, only the ref_dtm is used.

`verbose` If true, report progress

Details

The query dtm will contain the weighted term scores of the queries, and the lookup dtm will contain binary values for whether or not terms occured. This is designed to be used with the document.compare or newsflow.compare functions to compare the query matrix to the lookup matrix, using the special 'query_lookup' similarity measure.

Performs two operations. First, clusters of very similar columns (high cosine similarity) can be merged into a single column. This is an OR (union) combination, meaning that if at least one column is nonzero, the value will be one. Second, all columns will be combined to get the co-occurences (AND, or intersect).

To keep the vocabulary size manageable, only terms with at least min_docfreq (minimum document frequency) and max_docprob (max document probability) are returned. If a ref_dtm is given, both the dtm and ref_dtm will be used to compute the docfreq and docprob values, used for filtering and weighting (unless use_dtm_and_ref = F).

Value

a list with a query dtm and lookup dtm.

Examples

```
q = create_queries(rnewsflow_dfm, min_docfreq = 2, union_sim_thres = 0.9,
                   max_docprob = 0.05, verbose = FALSE)
head(colnames(q$query_dtm),100)
```

`delete.duplicates` *Delete duplicate (or similar) documents from a document term matrix*

Description

Delete duplicate (or similar) documents from a document term matrix. Duplicates are defined by: having high content similarity, occuring within a given time distance and being published by the same source.

Usage

```
delete.duplicates(dtm, meta = NULL, date.var = "date",
  hour.window = c(-24, 24), group.var = NULL, measure = c("cosine",
  "overlap_pct"), similarity = 1, keep = "first", tf.idf = FALSE,
  dup_csv = NULL, verbose = F)
```

Arguments

`dtm`	A quanteda dfm. Alternatively, a DocumentTermMatrix from the tm package can be used, but then the meta parameter needs to be specified manually
`meta`	If dtm is a quanteda dfm, docvars(meta) is used by default (meta is NULL) to obtain the meta data. Otherwise, the meta data.frame has to be given by the user, with the rows of the meta data.frame matching the rows of the dtm (i.e. each row is a document)
`date.var`	The name of the column in meta that specifies the document date. default is "date". The values should be of type POSIXlt or POSIXct
`hour.window`	A vector of length 2, in which the first and second value determine the left and right side of the window, respectively. For example, c(-10, 36) will compare each document to all documents between the previous 10 and the next 36 hours.
`group.var`	Optionally, The name of the column in meta that specifies a group (e.g., source, sourcetype). If given, only documents within the same group will be compared.
`measure`	the measure that should be used to calculate similarity/distance/adjacency. Currently supports the symmetrical measure "cosine" (cosine similarity), and the assymetrical measures "overlap_pct" (percentage of term scores in the document that also occur in the other document).
`similarity`	a threshold for similarity. Documents of which similarity is equal or higher are deleted
`keep`	A character indicating whether to keep the 'first' or 'last' published of duplicate documents.

tf.idf	if TRUE, weight the dtm with tf.idf before comparing documents. The original (non-weighted) DTM is returned.
dup_csv	Optionally, a path for writing a csv file with the duplicates edgelist. For each duplicate pair it is noted if "from" or "to" is the duplicate, or if "both" are duplicates (of other documents)
verbose	if TRUE, report progress

Details

Note that this can also be used to delete "updates" of articles (e.g., on news sites, news agencies). This should be considered if the temporal order of publications is relevant for the analysis.

Value

A dtm with the duplicate documents deleted

Examples

```
## example with very low similarity threshold (normally not recommended!)
dtm2 = delete.duplicates(rnewsflow_dfm, similarity = 0.5, keep='first', tf.idf = TRUE)
```

---

directed.network.plot

A wrapper for plot.igraph for visualizing directed networks.

Description

This is a convenience function for visualizing directed networks with edge labels using plot.igraph. It was designed specifically for visualizing aggregated document similarity networks in the RNewsflow package, but works with any network in the igraph class.

Usage

```
directed.network.plot(g, weight.var = "from.Vprop",
  weight.thres = NULL, delete.isolates = FALSE, vertex.size = 30,
  vertex.color = "lightblue", vertex.label.color = "black",
  vertex.label.cex = 0.7, edge.color = "grey",
  show.edge.labels = TRUE, edge.label.color = "black",
  edge.label.cex = 0.6, edge.arrow.size = 1,
  layout = igraph::layout.davidson.harel, ...)
```

Arguments

g	A network/graph in the igraph class
weight.var	The edge attribute that is used to specify the edges
weight.thres	A threshold for weight. Edges below the threshold are ignored

`delete.isolates`
: If TRUE, isolates (i.e. vertices without edges) are ignored.

`vertex.size`
: The size of the verticex/nodes. Defaults to 30. Can be a vector with values per vertex.

`vertex.color`
: Color of vertices/nodes. Default is lightblue. Can be a vector with values per vertex.

`vertex.label.color`
: Color of labels for vertices/nodes. Defaults to black. Can be a vector with values per vertex.

`vertex.label.cex`
: Size of the labels for vertices/nodes. Defaults to 0.7. Can be a vector with values per vertex.

`edge.color`
: Color of the edges. Defaults to grey. Can be a vector with values per edge.

`show.edge.labels`
: Logical. Should edge labels be displayed? Default is TRUE.

`edge.label.color`
: Color of the edge labels. Defaults to black. Can be a vector with values per edge.

`edge.label.cex`
: Size of the edge labels. Defaults to 0.6. Can be a vector with values per edge.

`edge.arrow.size`
: Size of the edge arrows. Defaults to 1. Can only be set globally (igraph might update this at some point)

`layout`
: The igraph layout used to plot the network. Defaults to layout.davidson.harel

`...`
: Arguments to be passed to the plot.igraph function.

Value

Nothing

Examples

```
data(docnet)
aggdocnet = network.aggregate(docnet, by='source')
directed.network.plot(aggdocnet, weight.var = 'to.Vprop', weight.thres = 0.2)
```

---

docnet *Document similarity network for one news agency, and the print and online editions of two newspapers*

Description

Document similarity network for one news agency, and the print and online editions of two newspapers

Format

docnet: A network/graph in the igraph class as created with document.network or newsflow.compare.

`document.network` *Create a document similarity network*

Description

Combines document similarity data (d) with document meta data (meta) into an igraph network/graph.

Usage

```
document.network(d, meta, id.var = "document_id", date.var = "date",
  min.similarity = NA)
```

Arguments

`d`	A data.frame with three columns, that represents an edgelist with weight values. The first and second column represent the names/ids of the 'from' and 'to' documents/vertices. The third column represents the similarity score. Column names are ignored
`meta`	A data.frame where rows are documents and columns are document meta information. Should at least contain 2 columns: the document name/id and date. The name/id column should match the document names/ids of the edgelist, and its label is specified in the 'id.var' argument. The date column should be intepretable with as.POSIXct, and its label is specified in the 'date.var' argument.
`id.var`	The label for the document name/id column in the 'meta' data.frame. Default is "document_id"
`date.var`	The label for the document date column in the 'meta' data.frame . default is "date"
`min.similarity`	For convenience, ignore all edges where the weight is below 'min.similarity'.

Details

This function is mainly offered to mimic the output of the newsflow.compare function when using imported document similarity data. This way the functions for transforming, aggregating and visualizing the document similarity data can be used.

Value

A network/graph in the igraph class

Examples

```
d = data.frame(x = c(1,1,1,2,2,3),
               y = c(2,3,5,4,5,6),
               v = c(0.3,0.4,0.7,0.5,0.2,0.9))

meta = data.frame(document_id = 1:8,
```

```
                    date = seq.POSIXt(from = as.POSIXct('2010-01-01 12:00:00'),
                           by='hour', length.out = 8),
                    medium = c(rep('Newspapers', 4), rep('Blog', 4)))

g = document.network(d, meta)

igraph::get.data.frame(g, 'both')
igraph::plot.igraph(g)
```

`document.network.plot` *Visualize (a subcomponent) of the document similarity network*

Description

Visualize (a subcomponent) of the document similarity network

Usage

```
document.network.plot(g, date.attribute = "date",
  source.attribute = "source", subcomp_i = NULL, dtm = NULL,
  sources = NULL, only.outer.date = FALSE,
  date.format = "%Y-%m-%d %H:%M", margins = c(5, 8, 1, 13),
  isolate.color = NULL, source.loops = TRUE, ...)
```

Arguments

`g`	A document similarity network, as created with newsflow.compare or document.network
`date.attribute`	The label of the vertex/document date attribute. Default is "date"
`source.attribute`	The label of the vertex/document source attribute. Default is "source"
`subcomp_i`	Optional. If an integer is given, the network is decomposed into subcomponents (i.e. unconnected components) and only the i-th component is visualized.
`dtm`	Optional. If a document-term matrix that contains the documents in g is given, a wordcloud with the most common words of the network is added.
`sources`	Optional. Use a character vector to select only certain sources
`only.outer.date`	If TRUE, only the labels for the first and last date are reported on the x-axis
`date.format`	The date format of the date labels (see format.POSIXct)
`margins`	The margins of the network plot. The four values represent bottom, left, top and right margin.
`isolate.color`	Optional. Set a custom color for isolates
`source.loops`	If set to FALSE, all edges between vertices/documents of the same source are ignored.
`...`	Additional arguments for the network plotting function plot.igraph

Value

Nothing.

Examples

```
docnet = docnet
dtm = rnewsflow_dfm

docnet_comps = igraph::decompose.graph(docnet) # get subcomponents

# subcomponent 1
document.network.plot(docnet_comps[[1]])

# subcomponent 2 with wordcloud
document.network.plot(docnet_comps[[2]], dtm=dtm)

# subcomponent 3 with additional arguments passed to plot.igraph
document.network.plot(docnet_comps[[3]], dtm=dtm, vertex.color='red')
```

---

documents.compare *Compare the documents in two corpora/dtms*

Description

Compare the documents in corpus dtm.x with reference corpus dtm.y.

Usage

```
documents.compare(dtm, dtm.y = NULL, measure = c("cosine",
  "overlap_pct", "overlap", "crossprod", "softcosine", "query_lookup",
  "query_lookup_pct"), min.similarity = 0, n.topsim = NULL,
  pvalue = c("none", "normal", "lognormal", "nz_normal", "nz_lognormal",
  "disparity"), simmat = NULL, simmat_thres = NULL)
```

Arguments

dtm	A quanteda dfm. Alternatively, a DocumentTermMatrix from the tm package can be used.
dtm.y	Optional. If given, documents from dtm will only be compared to the documents in dtm.y
measure	the measure that should be used to calculate similarity/distance/adjacency. Currently supports the symmetrical measure "cosine" (cosine similarity), the assymetrical measures "overlap_pct" (percentage of term scores in the document that also occur in the other document), "overlap" (like overlap_pct, but as the sum of overlap instead of the percentage) and the symmetrical soft cosine measure (experimental). The regular crossprod (inner product) is also supported. If the dtm's are prepared with the create_queries function, the special "query_lookup" and "query_lookup_pct" can be used.

`min.similarity`
: a threshold for similarity. lower values are deleted. Set to 0 by default.

`n.topsim`
: An alternative or additional sort of threshold for similarity. Only keep the [n.topsim] highest similarity scores for x. Can return more than [n.topsim] similarity scores in the case of duplicate similarities.

`pvalue`
: If used, transform the similarity score to a p-value. The value is reversed, so that higher means more similar (and thus the min.similarity still makes sense). Currently supports "normal" and "lognormal" distribution, and the uniform distribution used in the "disparity" filter (see Serrano et al.[1]). Also "nz_normal" and "nz_lognormal" can be used to only consider the nonzero values.

`simmat`
: If softcosine is used, a symmetrical matrix with the similarity scores of terms. If NULL, the cosine similarity of terms in dtm will be used

`simmat_thres`
: If softosine is used, a threshold for the similarity scores of terms

Details

The calculation of document similarity is performed using a vector space model approach. Inner-product based similarity measures are used, such as cosine similarity. It is recommended to weight the DTM beforehand, for instance using Term frequency-inverse document frequency (tf.idf)

Value

A data frame with pairs of documents and their similarities.

Examples

```
rnewsflow_dfm

comp = documents.compare(rnewsflow_dfm, min.similarity=0.4)
head(comp)
```

`filter.window` *Filter edges from the document similarity network based on hour difference*

Description

The ‘filter.window‘ function can be used to filter the document pairs (i.e. edges) using the ‘hour.window‘ parameter, which works identical to the ‘hour.window‘ parameter in the ‘newsflow.compare‘ function. In addition, the ‘from.vertices‘ and ‘to.vertices‘ parameters can be used to select the vertices (i.e. documents) for which this filter is applied.

Usage

```
filter.window(g, hour.window, to.vertices = NULL, from.vertices = NULL)
```

[1] https://www.pnas.org/content/106/16/6483.full

Arguments

g	A document similarity network, as created with newsflow.compare or document.network
hour.window	A vector of length 2, in which the first and second value determine the left and right side of the window, respectively. For example, c(-10, 36) will compare each document to all documents between the previous 10 and the next 36 hours.
to.vertices	A filter to select the vertices 'to' which an edge is filtered. For example, if 'V(g)$sourcetype == "newspaper"' is used, then the hour.window filter is only applied for edges 'to' newspaper documents (specifically, where the sourcetype attribute is "newspaper").
from.vertices	A filter to select the vertices 'from' which an edge is filtered. Works identical to 'to.vertices'.

Details

It is recommended to use the show.window function to verify whether the hour windows are correct according to the assumptions and focus of the study.

Value

A network/graph in the igraph class

Examples

```
data(docnet)
show.window(docnet, to.attribute = 'source') # before filtering

docnet = filter.window(docnet, hour.window = c(0.1,24))

docnet = filter.window(docnet, hour.window = c(6,36),
                       to.vertices = V(docnet)$sourcetype == 'Print NP')

show.window(docnet, to.attribute = 'sourcetype') # after filtering per sourcetype
show.window(docnet, to.attribute = 'source') # after filtering per source
```

---

get_doc_terms *View term scores for a given document*

Description

View term scores for a given document

Usage

```
get_doc_terms(dtm, docname = NULL, doc_i = NULL)
```

Arguments

dtm	A quanteda dfm
docname	name of document to select
doc_i	alternatively, select document by index

Value

A named vector with terms (names) and scores

Examples

```
get_doc_terms(rnewsflow_dfm, doc_i=1)
```

---

get_overlap_terms *View overlapping terms for a given pair of documents*

Description

View overlapping terms for a given pair of documents

Usage

```
get_overlap_terms(dtm, doc.x, doc.y, dtm.y = dtm)
```

Arguments

dtm	A quanteda dfm
doc.x	The name of the first document in dtm
doc.y	The name of the second document in dtm (or dtm.y)
dtm.y	Optionally, a second dtm (for when the documents occur in separate dtm's)

Value

A character vector

Examples

```
get_overlap_terms(rnewsflow_dfm,
                  quanteda::docnames(rnewsflow_dfm)[1],
                  quanteda::docnames(rnewsflow_dfm)[5])
```

hourdiff_range_thresholds *Inspect effects of thresholds on matches over time*

Description

If it can be assumed that matches should only occur within a given time range (e.g., event data should match news items after the event occured) a low effort validation can be obtained by looking at whether the matches only occur within this time range. This function plots the percentage of matches within a given time range (hourdiff) for different thresholds of the weight column. This can be used to determine a good threshold.

Usage

```
hourdiff_range_thresholds(g, breaks = 20, hourdiff_range = c(0, Inf),
  min_weight = NA, min_hourdiff = NA, max_hourdiff = NA)
```

Arguments

g	The output of newsflow.compare (either as "igraph" or "edgelist")
breaks	The number of breaks for the weight threshold
hourdiff_range	The time period (hourdiff range) in which the match 'should' occur.
min_weight	Optionally, filter out all value below the given weight
min_hourdiff	the lowest possible hourdiff value. This is used to estimate noise. If not specified, will be estimated based on data.
max_hourdiff	the highest possible hourdiff value.

Value

Nothing... just plots

network.aggregate *Aggregate the edges of a network by vertex attributes*

Description

This function offers a versatile way to aggregate the edges of a network based on the vertex attributes. Although it was designed specifically for document similarity networks, it can be used for any network in the igraph class.

Usage

```
network.aggregate(g, by = NULL, by.from = by, by.to = by,
  edge.attribute = "weight", agg.FUN = mean, return.df = FALSE,
  keep_isolates = T)
```

Arguments

g	A network/graph in the igraph class
by	A character string indicating the vertex attributes by which the edges will be aggregated.
by.from	Optionally, specify different vertex attributes to aggregate the ‘from‘ side of edges
by.to	Optionally, specify different vertex attributes to aggregate the ‘to‘ side of edges
edge.attribute	Select an edge attribute to aggregate using the function specified in ‘agg.FUN‘. Defaults to ’weight’
agg.FUN	The function used to aggregate the edge attribute
return.df	Optional. If TRUE, the results are returned as a data.frame. This can in particular be convenient if by.from and by.to are used.
keep_isolates	if True, also return scores for isolates

Details

The first argument is the network (in the ‘igraph‘ class). The second argument, for the ‘by‘ parameter, is a character vector to indicate one or more vertex attributes based on which the edges are aggregated. Optionally, the ‘by‘ parameter can also be specified separately for ‘by.from‘ and ‘by.to‘.

By default, the function returns the aggregated network as an igraph class. The edges in the aggregated network have five standard attributes. The ‘edges‘ attribute counts the number of edges from the ‘from‘ group to the ‘to‘ group. The ‘from.V‘ attribute shows the number of vertices in the ‘from‘ group that matched with a vertex in the ‘to‘ group. The ‘from.Vprop attribute shows this as the proportion of all vertices in the ‘from‘ group. The ‘to.V‘ and ‘to.Vprop‘ attributes show the same for the ‘to‘ group.

In addition, one of the edge attributes of the original network can be aggregated with a given function. These are specified in the ‘edge.attribute‘ and ‘agg.FUN‘ parameters.

Value

A network/graph in the igraph class, or a data.frame if return.df is TRUE.

Examples

```
data(docnet)
aggdocnet = network.aggregate(docnet, by='sourcetype')
igraph::get.data.frame(aggdocnet, 'both')

aggdocdf = network.aggregate(docnet, by.from='sourcetype', by.to='source', return.df = TRUE)
head(aggdocdf)
```

`newsflow.compare` *Compare the documents in a dtm with a sliding window over time*

Description

Given a document-term matrix (DTM) with dates for each document, calculates the document similarities over time using with a sliding window.

Usage

```
newsflow.compare(dtm, dtm.y = NULL, meta = NULL, meta.y = NULL,
  date.var = "date", hour.window = c(-24, 24), group.var = NULL,
  measure = c("cosine", "overlap_pct", "overlap", "crossprod",
  "softcosine", "query_lookup", "query_lookup_pct"), min.similarity = 0,
  n.topsim = NULL, only.from = NULL, only.to = NULL,
  only.complete.window = TRUE, pvalue = c("none", "normal",
  "lognormal", "nz_normal", "nz_lognormal", "disparity"),
  return_as = c("igraph", "edgelist", "matrix"), batchsize = 1000,
  simmat = NULL, simmat_thres = NULL, verbose = FALSE)
```

Arguments

`dtm`	A quanteda dfm. Alternatively, a DocumentTermMatrix from the tm package can be used, but then the meta parameter needs to be specified manually
`dtm.y`	optionally, another dtm. If given, the documents in dtm will be compared to the documents in dtm.y. This cannot be combined with only.from and only.to
`meta`	If dtm is a quanteda dfm, docvars(meta) is used by default (meta is NULL) to obtain the meta data. Otherwise, the meta data.frame has to be given by the user, with the rows of the meta data.frame matching the rows of the dtm (i.e. each row is a document)
`meta.y`	like meta, but for dtm.y (only necessary if dtm.y is used)
`date.var`	The name of the column in meta that specifies the document date. default is "date". The values should be of type POSIXct
`hour.window`	A vector of length 2, in which the first and second value determine the left and right side of the window, respectively. For example, c(-10, 36) will compare each document to all documents between the previous 10 and the next 36 hours.
`group.var`	Optionally, The name of the column in meta that specifies a group (e.g., source, sourcetype). If given, only documents within the same group will be compared.
`measure`	the measure that should be used to calculate similarity/distance/adjacency. Currently supports the symmetrical measure "cosine" (cosine similarity), the assymetrical measures "overlap_pct" (percentage of term scores in the document that also occur in the other document), "overlap" (like overlap_pct, but as the sum of overlap instead of the percentage) and the symmetrical soft cosine measure (experimental). The regular crossprod (inner product) is also supported. If the dtm's are prepared with the create_queries function, the special "query_lookup" and "query_lookup_pct" can be used.

`min.similarity`
: a threshold for similarity. lower values are deleted. Set to 0.1 by default.

`n.topsim`
: An alternative or additional sort of threshold for similarity. Only keep the [n.topsim] highest similarity scores for x. Can return more than [n.topsim] similarity scores in the case of duplicate similarities.

`only.from`
: A vector with names/ids of documents (dtm rownames), or a logical vector that matches the rows of the dtm. Use to compare only these documents to other documents.

`only.to`
: A vector with names/ids of documents (dtm rownames), or a logical vector that matches the rows of the dtm. Use to compare other documents to only these documents.

`only.complete.window`
: if True, only compare articles (x) of which a full window of reference articles (y) is available. Thus, for the first and last [window.size] days, there will be no results for x.

`pvalue`
: If used, transform the similarity score to a p-value. The value is reversed, so that higher means more similar (and thus the min.similarity still makes sense). Currently supports "normal" and "lognormal" distribution, and the uniform distribution used in the "disparity" filter (see Serrano et al.[2]). Also "nz_normal" and "nz_lognormal" can be used to only consider the nonzero values.

`return_as`
: Detemine whether output is returned as an "edgelist", "igraph" network or sparse "matrix'.

`batchsize`
: If group and/or date are used, size of batches.

`simmat`
: If softcosine is used, a symmetrical matrix with the similarity scores of terms. If NULL, the cosine similarity of terms in dtm will be used

`simmat_thres`
: If softosine is used, a threshold for the similarity scores of terms

`verbose`
: If TRUE, report progress

Details

The calculation of document similarity is performed using a vector space model approach. Inner-product based similarity measures are used, such as cosine similarity. It is recommended to weight the DTM beforehand, for instance using Term frequency-inverse document frequency (tf.idf)

Value

A network/graph in the igraph class

Examples

```
rnewsflow_dfm

dtm = quanteda::dfm_tfidf(rnewsflow_dfm)
g = newsflow.compare(dtm, hour.window = c(0.1, 36))
```

[2] https://www.pnas.org/content/106/16/6483.full

```
vcount(g) # number of documents, or vertices
ecount(g) # number of document pairs, or edges

head(igraph::get.data.frame(g, 'vertices'))
head(igraph::get.data.frame(g, 'edges'))
```

`only.first.match` *Transform document network so that each document only matches the earliest dated matching document*

Description

Transforms the network so that a document only has an edge to the earliest dated document it matches within the specified time window[^duplicate].

Usage

```
only.first.match(g)
```

Arguments

`g`	A document similarity network, as created with newsflow.compare or document.network

Details

If there are multiple earliest dated documents (that is, having the same publication date) then edges to all earliest dated documents are kept.

Value

A network/graph in the igraph class

Examples

```
data(docnet)

subcomp1 = igraph::decompose.graph(docnet)[[2]]
subcomp2 = only.first.match(subcomp1)

igraph::get.data.frame(subcomp1)
igraph::get.data.frame(subcomp2)

graphics::par(mfrow=c(2,1))
document.network.plot(subcomp1, main='All matches')
document.network.plot(subcomp2, main='Only first match')
graphics::par(mfrow=c(1,1))
```

rnewsflow_dfm *quanteda dfm for RNewsflow vignette demo*

Description

quanteda dfm for RNewsflow vignette demo

Usage

```
rnewsflow_dfm
```

Format

dfm

show.window *Show time window of document pairs*

Description

This function aggregates the edges for all combinations of attributes specified in 'from.attribute' and 'to.attribute', and shows the minimum and maximum hour difference for each combination.

Usage

```
show.window(g, to.attribute = NULL, from.attribute = NULL)
```

Arguments

g	A document similarity network, as created with newsflow.compare or document.network
to.attribute	The vertex attribute to aggregate the 'to' group of the edges
from.attribute	The vertex attribute to aggregate the 'from' group of the edges

Details

The filter.window function can be used to filter edges that fall outside of the intended time window.

Value

A data.frame showing the left and right edges of the window for each unique group.

Examples

```
data(docnet)
show.window(docnet, to.attribute = 'source')
show.window(docnet, to.attribute = 'sourcetype')
show.window(docnet, to.attribute = 'sourcetype', from.attribute = 'sourcetype')
```

---

tcrossprod_sparse *The tcrossprod function for sparse matrices, for people that like drowning in parameters*

Description

tcrossprod with benefits. Enables limiting row combinations to within specified groups and date windows, and filters results that do not pass the threshold on the fly. To achieve this, options for similarity measures are included in the function. For example, to get the cosine similarity, you can normalize with "l2" and use the "prod" (product) function for the

Usage

```
tcrossprod_sparse(m, m2 = NULL, min_value = NULL, max_value = NULL,
  only_upper = F, diag = T, top_n = NULL, rowsum_div = F,
  pvalue = c("none", "normal", "lognormal", "nz_normal", "nz_lognormal",
  "disparity"), normalize = c("none", "l2", "softl2"),
  crossfun = c("prod", "min", "softprod", "maxproduct"), group = NULL,
  group2 = NULL, date = NULL, date2 = NULL, lwindow = -1,
  rwindow = 1, date_unit = c("days", "hours", "minutes", "seconds"),
  simmat = NULL, simmat_thres = NULL, batchsize = 1000,
  verbose = F)
```

Arguments

m	A dgCMatrix
m2	A dgCMatrix
min_value	Optionally, a numerical value, specifying the threshold for including a score in the output.
max_value	Optionally, a numerical value for the upper limit for including a score in the output.
only_upper	if true, only the upper triangle of the matrix is returned. Only possible for symmetrical output (m and m2 have same number of columns)
diag	if false, the diagonal of the matrix is not returned. Only possible for symmetrical output (m and m2 have same number of columns)
top_n	an integer, specifying the top number of strongest scores for each column in m
rowsum_div	if true, divide crossproduct by column sums of m. (this has to happen within the loop for min_value and top_n filtering)

pvalue	If used, transform the similarity score to a p-value. The value is reversed, so that higher means more similar (and thus the min.similarity still makes sense). Currently supports "normal" and "lognormal" distribution, and the uniform distribution used in the "disparity" filter (see Serrano et al.[3]). Also "nz_normal" and "nz_lognormal" can be used to only consider the nonzero values.
normalize	normalize rows by a given norm score. Default is 'none' (no normalization). 'l2' is the l2 norm (use in combination with 'prod' crossfun for cosine similarity). 'l2soft' is the adaptation of l2 for soft similarity (use in combination with 'softprod' crossfun for soft cosine)
crossfun	The function used in the vector operations. Normally this is the "prod", for product (dot product). Here we also allow the "min", for minimum value. We use this in our document overlap_pct score. In addition, there is the (experimental) softprod, that can be used in combination with softl2 normalization to get the soft cosine similarity. And, the "maxproduct" is a special case used in the query_lookup measure, that uses product but only returns the score of the strongest matching term.
group	Optionally, a character vector that specifies a group (e.g., source) for each row in m. If given, only pairs of rows with the same group are calculated.
group2	If m2 and group are used, group2 has to be used to specify the groups for the rows in m2 (otherwise group will be ignored)
date	Optionally, a character vector that specifies a date for each row in m. If given, only pairs of rows within a given date range (see lwindow, rwindow and date_unit) are calculated.
date2	If m2 and date are used, date2 has to be used to specify the date for the rows in m2 (otherwise date will be ignored)
lwindow	If date (and date2) are used, lwindow determines the left side of the date window. e.g. -10 means that rows are only matched with rows for which date is within 10 [date_units] before.
rwindow	Like lwindow, but for the right side. e.g. an lwindow of -1 and rwindow of 1, with date_unit is "days", means that only rows are matched for which the dates are within a 1 day distance
date_unit	The date unit used in lwindow and rwindow. Supports "days", "hours", "minutes" and "seconds". Note that refers to the time distance between two rows ("days" doesn't refer to calendar days, but to a time of 24 hours)
simmat	if softcos is used, a symmetric matrix with terms that indicates the similarity of terms (i.e. adjacency matrix). If NULL, a cosine similarity matrix will be created on the go
simmat_thres	if softcos is used, a threshold for the term similarity.
batchsize	If group and/or date are used, size of batches.
verbose	if TRUE, report progress

[3] https://www.pnas.org/content/106/16/6483.full

Details

This function is called by the document comparison functions (documents.compare, newsflow.compare, delete.duplicates). We only expose it here for additional flexibility, and because it could be usefull outside of the purpose of this package.

Value

A dgCMatrix

Examples

```
set.seed(1)
m = Matrix::rsparsematrix(5,10,0.5)
tcrossprod_sparse(m, min_value = 0, only_upper = FALSE, diag = TRUE)
tcrossprod_sparse(m, min_value = 0, only_upper = FALSE, diag = FALSE)
tcrossprod_sparse(m, min_value = 0, only_upper = TRUE, diag = FALSE)
tcrossprod_sparse(m, min_value = 0.2, only_upper = TRUE, diag = FALSE)
tcrossprod_sparse(m, min_value = 0, only_upper = TRUE, diag = FALSE, top_n = 1)
```

---

term.day.dist *Calculate statistics for term occurence across days*

Description

Calculate statistics for term occurence across days

Usage

```
term.day.dist(dtm, meta = NULL, date.var = "date")
```

Arguments

dtm	A quanteda dfm. Alternatively, a DocumentTermMatrix from the tm package can be used, but then the meta parameter needs to be specified manually
meta	If dtm is a quanteda dfm, docvars(meta) is used by default (meta is NULL) to obtain the meta data. Otherwise, the meta data.frame has to be given by the user, with the rows of the meta data.frame matching the rows of the dtm (i.e. each row is a document)
date.var	The name of the meta column specifying the document date. default is "date". The values should be of type POSIXlt or POSIXct

Value

A data.frame with statistics for each term.

- freq: The number of times a term occurred
- doc.freq: The number of documents in which a term occured
- days.n: The number of days on which a term occured
- days.pct: The percentage of days on which a term occured
- days.entropy: The entropy of the distribution of term frequency across days
- days.entropy.norm: The normalized days.entropy, where 1 is a discrete uniform distribution

Examples

```
tdd = term.day.dist(rnewsflow_dfm, date.var='date')
head(tdd)
tail(tdd)
```

---

term_char_sim *Find terms with similar spelling*

Description

A quick, language agnostic way for finding terms with similar spelling. Calculates similarity as percentage of a terms bigram's or trigram's that also occur in the other term. The percentage has to be above the given threshold for both terms (unless allow_asym = T)

Usage

```
term_char_sim(voc, type = c("tri", "bi"), min_overlap = 2/3,
  max_diff = 4, pad = F, as_lower = T, same_start = 1,
  drop_non_alpha = T, min_length = 5, allow_asym = F, verbose = T)
```

Arguments

voc	A character vector that gives the vocabulary (e.g., colnames of a dtm)
type	Either "bi" (bigrams) or "tri" (trigrams)
min_overlap	The minimal overlap percentage. Works together with max_diff to determine required overlap
max_diff	The maximum number of bi/tri-grams that is different
pad	If True, pad the left size (ls) and right side (rs) of bi/tri-grams. So, trigrams for "pad" would be: "ls_ls_p", "ls_p_a", "p_a_d", "a_d_rs", "d_rs_rs".
as_lower	If True, ignore case
same_start	Should terms start with the same character(s)? Given as a number for the number of same characters. (also greatly speeds up calculation)

drop_non_alpha
: If True, ignore non alpha terms (e.g., numbers, punctuation). They will appear in the output matrix, but only with zeros.

min_length
: The minimum number of characters in a term. Terms with fewer characters are ignored. They will appear in the output matrix, but only with zeros.

allow_asym
: If True, the match only needs to be true for at least one term. In practice, this means that "America" would match perfectly with "Southern-America".

verbose
: If True, report progress

Value

A similarity matrix in the dgCMatrix format

Examples

```
dfm = quanteda::dfm(c('That guy Gadaffi','Do you mean Kadaffi?',
                      'Nah more like Gadaffel','What Gargamel?'))
simmat = term_char_sim(colnames(dfm), same_start=0)
term_union(dfm, simmat, verbose = FALSE)
```

---

term_innovation *Experimental: Convert dtm scores to a term innovation score, based on changes in term use over time*

Description

For each term in m, the usage before and after the document date is compared (with a chi2 test) to see whether usage increased.

Usage

```
term_innovation(m, date, m2 = NULL, date2 = NULL, lwindow = -7,
  rwindow = 7, date_unit = c("days", "hours", "minutes", "seconds"),
  min_chi = 5.024, min_ratio = 2, smooth = 1)
```

Arguments

m
: A dgCMatrix

date
: a character vector that specifies a date for each row in m. If given, only pairs of rows within a given date range (see lwindow, rwindow and date_unit) are calculated.

m2
: Optionally, use a different matrix for calculating the innovation scores. For example, if m is a DTM of press releases, m2 can be a DTM of news articles, to see if term usage increased in the news after the press release.

date2
: If m2 is used, date2 has to be used to specify the date for the rows in m2 (otherwise date will be ignored)

lwindow	If date (and date2) are used, lwindow determines the left side of the date window. e.g. -10 means that rows are only matched with rows for which date is within 10 [date_units] before.
rwindow	Like lwindow, but for the right side. e.g. an lwindow of -1 and rwindow of 1, with date_unit is "days", means that only rows are matched for which the dates are within a 1 day distance
date_unit	The date unit used in lwindow and rwindow. Supports "days", "hours", "minutes" and "seconds". Note that refers to the time distance between two rows ("days" doesn't refer to calendar days, but to a time of 24 hours)
min_chi	The minimum chi-square value
min_ratio	The minimum ratio (rwindow score / lwindow score)
smooth	The smoothing factor (prevents -Inf/Inf ratio)

Value

A dgCMatrix

---

term_intersect *Combine terms in a dtm*

Description

Given a dtm and a similarity (adjacency) matrix, create a new column for each nonzero cell in the similarity matrix. For the term combinations (everything except the diagonal) the column names will be pasted together with a "&" separator (read as AND)

Usage

```
term_intersect(dtm, simmat, as_dfm = T, verbose = F, par = NA,
  sep = " & ")
```

Arguments

dtm	A quanteda dfm or a dgCMatrix.
simmat	A similarity matrix in dgCMatrix format. For instance, created with term_char_sim
as_dfm	If True, return as quanteda dfm
verbose	If True, report progress
par	argument needing description
sep	The separator used for pasting the terms

Value

A dgCMatrix or quanteda dfm

`term_union` *Combine terms in a dtm*

Description

Given a dtm and a similarity (adjacency) matrix, group clusters of similar terms (simmat > 0) into a single column. Column names will be concatenated, with a "|" seperator (read as OR)

Usage

```
term_union(dtm, simmat, as_dfm = T, verbose = F, sep = "|",
  par = NA)
```

Arguments

`dtm`	A quanteda dfm or a dgCMatrix.
`simmat`	A similarity matrix in dgCMatrix format. For instance, created with term_char_sim
`as_dfm`	If True, return as quanteda dfm
`verbose`	If True, report progress
`sep`	The separator used for pasting the terms
`par`	argument needing description

Value

A dgCMatrix or quanteda dfm

Examples

```
dfm = quanteda::dfm(c('That guy Gadaffi','Do you mean Kadaffi?',
                      'Nah more like Gadaffel','What Gargamel?'))
simmat = term_char_sim(colnames(dfm), same_start=0)
term_union(dfm, simmat, verbose = FALSE)
```