Package ‘RNewsflow’

April 3, 2024

Type Package
Title Tools for Comparing Text Messages Across Time and Media
Version 1.2.8
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 GPL-3
Depends R (>= 3.3.0), igraph (>= 1.3.4), Matrix (>= 1.5)
Imports stringi (>= 1.7.8), scales (>= 1.2.1), wordcloud (>= 2.6),
data.table (>= 1.10.4), methods, quanteda (>= 3.2.3), Rcpp (>= 0.12.12)
LinkingTo Rcpp (>= 1.0.9), RcppEigen (>= 0.3.3.9.2), RcppProgress (>= 0.4.2)
LazyData true
RoxygenNote 7.3.1
Suggests knitr (>= 1.40), rmarkdown (>= 2.16)
VignetteBuilder knitr
Encoding UTF-8
NeedsCompilation yes
Repository CRAN
Date/Publication 2024-04-03 11:03:02 UTC

R topics documented:
as_document_network ........................................ 2
compare_documents ........................................... 3
create_document_network .................................... 6
create_queries ............................................... 7
as_document_network

Description

This function can be used to structure the output of the compare_documents function as an igraph network.

Usage

as_document_network(el)

Arguments

el  An RNewsflow_edgelist object, as created with compare_documents.

Value

A network/graph in the igraph class

Examples

dtm = quanteda::dfm_tfidf(rnewsflow_dfm)
el = compare_documents(dtm, date_var='date', hour_window = c(0.1, 36))
g = as_document_network(el)
g
Description

This function calculates document similarity scores using a vector space approach. The most important benefit is that it includes options for limiting the number of comparisons that need to be made and filtering the results, that are efficiently implemented in a custom inner product calculation. This makes it possible to compare a huge number of documents, especially for cases where only documents within a given time window need to be compared.

Usage

```r
compare_documents(
  dtm,
  dtm_y = NULL,
  date_var = NULL,
  hour_window = c(-24, 24),
  group_var = NULL,
  measure = c("cosine", "overlap_pct", "overlap", "dot_product", "softcosine",
             "cp_lookup", "cp_lookup_norm"),
  tf_idf = F,
  min_similarity = 0,
  n_topsim = NULL,
  only_complete_window = T,
  copy_meta = T,
  backbone_p = 1,
  simmat = NULL,
  simmat_thres = NULL,
  batchsize = 1000,
  verbose = FALSE
)
```

Arguments

dtm A quanteda dfm. Note that it is common to first weight the dtm(s) before calculating document similarity. For this you can use quanteda’s `dfm_tfidf` and `dfm_weight`
dtm_y Optionally, another dtm. If given, the documents in dtm will be compared to the documents in dtm_y.
date_var Optionally, the name of the column in docvars that specifies the document date. The values should be of type POSIXct, or coercable with as.POSIXct. If given, the hour_window argument is used to only compare documents within a time window.
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. It is possible to specify time windows down to the level of seconds by using fractions (hours / 60 / 60).

group_var
Optionally, The name of the column in docvars 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 asymmetrical 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 dot product (dot_product) is also supported.

tf_idf
If TRUE, weigh the dtm (and dtm_y) by term frequency - inverse document frequency. For more control over weighting, we recommend using quanteda’s dfm_tfidf or dfm_weight on dtm and dtm_y.

min_similarity
A threshold for similarity. lower values are deleted. For all available similarity measures zero means no similarity.

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_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.

copy_meta
If TRUE, copy the dtm docvars to the from_meta and to_meta data.tables

backbone_p
Apply backbone filtering with a "disparity" filter (see Serrano et al., DOI: 10.1073/pnas.0808904106). It is different from the original disparity filter algorithm in that it only looks at outward edges. Also, the outward degree k is measured as all possible edges (within a window), not just the non-zero edges.

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
A large, dense simmat can lead to memory problems and slows down computation. A pragmatig (though not mathematically pure) solution is to use a threshold to prune small similarities.

batchsize
For internal use (testing)

verbose
If TRUE, report progress

Details
By default, the function performs a regular tcrossprod of the dtm (with itself or with dtm_y). The following parameters can be set to limit comparisons and filter output:

- If the ‘date_var’ is specified. The given hour_window is used to only compare documents within the specified time distance.
- If the ‘group_var’ is specified, only documents for which the group is identical will be compared.
With the `min_similarity` argument, the output can be filtered with a minimum similarity threshold. For the inner product of two DTMs the size of the output matrix is often the main bottleneck for comparing many documents, because it generally increases exponentially with the number of documents in the DTMs. Even a low similarity threshold can greatly reduce the size of the output.

As an alternative or additional filter, you can limit the results for each row in dtm to the highest top_n similarity scores.

Margin attributes are also included in the output in the from_meta and to_meta data.tables (see details). If copy_meta = TRUE, The dtm docvars are also included in from_meta and to_meta.

Margin attributes are added to the meta data. The reason for including this is that some values that are normally available in a similarity matrix are missing if certain filter options are used. If group or date is used, we don’t know how many columns a rows has been compared to (normally this is all columns). If a min/max or top_n filter is used, we don’t know the true row sums (and thus row means). The meta data therefore includes the "row_n", "row_sum", "col_n", and "col_sum". In addition, there are "lag_n" and "lag_sum". this is a special case where row_n and row_sum are calculated for only matches where the column date < row date (lag). This can be used for more refined calculations of edge probabilities before and after a row document.

Value

A S3 class for RNewsflow_edgelist, which is a list with the edgelist, from_meta and to_meta data.tables.

Examples

dtm = quanteda::dfm_tfidf(rnewsflow_dfm)
el = compare_documents(dtm, date_var='date', hour_window = c(0.1, 36))

d = data.frame(text = c('a b c d e',
                      'f g h i j k',
                      'a b c'),
          date = as.POSIXct(c('2010-01-01', '2010-01-01', '2012-01-01')),
          stringsAsFactors=FALSE)
corp = quanteda::corpus(d, text_field='text')
dtm = quanteda::tokens(corp) |> quanteda::dfm()

g = compare_documents(dtm)
g

g = compare_documents(dtm, measure = 'overlap_pct')
g
create_document_network

Create a document similarity network

Description

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

Usage

create_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 interpretable 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 as_document_network 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
create_queries

**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 = create_document_network(d, meta)

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

---

**create_queries**  
*Automatically infer queries from combinations of terms in a dtm*

**Description**

This function was designed for the task of matching short event descriptions to news articles, but can more generally be used for document matching tasks. However, it should be noted that it will require exponentially more memory for dtms with more unique terms, which is why it is less suitable for matching larger documents. This only applies to the dtm, not the ref_dtm. Thus, if your goal is to match smaller documents such as event descriptions to news, this function might be useful.

**Usage**

```r
create_queries(
  dtm,
  ref_dtm = NULL,
  min_docfreq = 2,
  max_docprob = 0.01,
  weight = c("tfidf", "binary"),
  norm_weight = c("max", "doc_max", "dtm_max", "none"),
  min_obs_exp = NA,
  union_sim_thres = NA,
  combine_all = T,
  only_dtm_combs = T,
  use_dtm_and_ref = F,
  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, or of both the dtm and ref dtm if use_dtm_and_ref is T). Default is "binary" (does/does not occur). "tfidf" uses common tf-idf weighting (actually just idf, since scores are 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.

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, the weight is computed based only on the document frequencies in the ref dtm. If use_dtm_and_ref is set to TRUE, both the dtm and ref dtm are used.

verbose

If true, report progress.

Details

The main purpose of the function is that it intersects the terms in a dtm based to increase sparsity. This can improve certain document matching tasks, but at the cost of creating a bigger dtm. If all terms are combined this would be a quadratic increase of columns. However, only term combinations that occur in dtm (not ref_dtm) will be used. This is not a problem as long as the similarity of the documents in dtm to documents in dtm_y is calculated as an assymetric similarity measure (i.e. in which the sum of terms in dtm_y is not used).

To emphasize that this feature preparation step is geared towards the task of 'looking up' documents, we use the terminology of a 'query'. The output of the function is a list of two dtm: query_dtm and ref_dtm. Both dtms have the exact same columns that contain the query terms. The values in query_dtm are by default tfidf weighted, and the values in ref_dtm are binary.

Several options are given to only create term combinations that are informative. Firstly, a minimum and maximum document frequency of term combinations can be defined. Secondly, a minimum observed/expected ratio can be given. The expected probability of a combination of term A and term B is the joint probability. If the observed probability is not higher, the combination is not more informative than chance. Thirdly, before intersecting terms, one can first cluster very similar terms together as single columns to reduce the number of possible combinations.
**delete_duplicates**

**Value**

a list with a query dtm and ref_dtm. Designed for use in `compare_documents` using the special ‘query_lookup’ measure

**Examples**

```r
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, occurring within a given time distance and being published by the same source.

**Usage**

```r
delete_duplicates(
  dtm,
  date_var = NULL,
  hour_window = c(-24, 24),
  group_var = NULL,
  measure = c("cosine", "overlap_pct"),
  similarity = 1,
  keep = "first",
  tf_idf = FALSE,
  dup_csv = NULL,
  verbose = FALSE
)
```

**Arguments**

- `dtm` A quanteda `dfm`
- `date_var` The name of the column in `docvars(dtm)` that specifies the document 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, column name in `docvars(dtm)` 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**
```r
## 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**
```r
directed_network_plot(  
g,  
weight_var = "from.Vprop",  
weight_thres = NULL,  
delete_isolates = FALSE,  
vertex.size = 30,
```
directed_network_plot

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 vertices/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

```r
data(docnet)
aggdocnet = network_aggregate(docnet, by='source')
directed_network_plot(aggdocnet, weight_var = 'to.Vprop', weight_thres = 0.2)
```

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 create_document_network or newsflow_compare.

document_network_plot  Visualize (a subcomponent) of the document similarity network

Description

Visualize (a subcomponent) of the document similarity network

Usage

```r
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 `create_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')
**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**

```r
filter_window(g, hour_window, to_vertices = NULL, from_vertices = NULL)
```

**Arguments**

- `g`: A document similarity network, as created with `newsflow_compare` or `create_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**

```r
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')
```
The `get_doc_terms` function is used to view term scores for a given document. The function takes the following arguments:

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

The function returns a named vector with terms (names) and scores.

Example usage:

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

The `get_overlap_terms` function is used to view overlapping terms for a given pair of documents. The function takes the following 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)

The function returns a named vector with terms (names) and scores.

Example usage:

```r
get_overlap_terms(dtm, doc.x, doc.y)
```
Value

A character vector

Examples

```r
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 occurred) 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

```r
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

```r
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  Create a network of document similarities over time

Description

This is a wrapper for the compare_documents function, specialised for the case of analyzing documents over time. The difference is that using date_var is mandatory, and the output is returned as an igraph network (using as_document_network).

Usage

newsflow_compare(
  dtm,
  dtm_y = NULL,
  date_var = "date",
  hour_window = c(-24, 24),
  group_var = NULL,
  measure = c("cosine", "overlap_pct", "overlap", "dot_product", "softcosine"),
  tf_idf = F,
  min_similarity = 0,
  n_topsim = NULL,
  only_complete_window = T,
  ...
)
Arguments

- **dtm**: A quanteda dfm. Note that it is common to first weight the dtm(s) before calculating document similarity. For this you can use quanteda’s dfm_tfidf and dfm_weight.

- **dtm_y**: Optionally, another dtm. If given, the documents in dtm will be compared to the documents in dtm_y.

- **date_var**: The name of the column in meta that specifies the document date. default is "date". The values should be of type POSIXct, or coercable with as.POSIXct. If given, the hour_window argument is used to only compare documents within a time window.

- **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. It is possible to specify time windows down to the level of seconds by using fractions (hours / 60 / 60).

- **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 dot product (dot_product) is also supported.

- **tf_idf**: If TRUE, weigh the dtm (and dtm_y) by term frequency - inverse document frequency. For more control over weighting, we recommend using quanteda’s dfm_tfidf or dfm_weight on dtm and dtm_y.

- **min_similarity**: A threshold for similarity. lower values are deleted. For all available similarity measures zero means no similarity.

- **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_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.

- **...**: Other arguments passed to compare_documents.

Value

An igraph network.

Examples

dtm = quanteda::dfm_tfidf(rnewsflow_dfm)
el = newsflow_compare(dtm, date_var='date', hour_window = c(0.1, 36))
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

```r
only_first_match(g)
```

Arguments

- `g`: A document similarity network, as created with `newsflow_compare` or `create_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

```r
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))
```
**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 `create_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')
Description

This function (including the underlying cpp function batched_tcrossprod_cpp) is the workhorse of the RNewsflow package. It has unnervingly many arguments for a tcrossprod because it needs to be able to do many thing efficiently. While its mostly a backend function, we expose it because it has applications outside of RNewsflow, but we make no excuses for the fact that readability is very much sacrificed here for the convenience of being able to keep adding features that we need for RNewsflow.

Usage

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

Arguments

- `m` A CsparseMatrix
ttcrossprod_sparse

m2
A CsparseMatrix

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 similarities per row. So, for each row in m at most top_n scores are returned.

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).

max_p
A threshold for maximum p value.

pvalue
If max_p < 1, edges are removed based on a p value. For each document in dtm, a p value is calculated over its outward edges. Default is the p-value based on uniform distribution, akin to a "disparity" filter (see Serrano et al., DOI: 10.1073/pnas.0808904106) but without filtering on inward edges.

normalize
Normalize rows by a given norm score (before calculating similarity). 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. 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. The "cp_lookup" and "cp_lookup_norm" are special cases for conditional probability sensitive lookup.

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 POSIXct vector (or a vector that can be converted to as.POSIXct) 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.

row_attr: If TRUE, add the "row_n" and "row_sum" elements to the "margin" attribute.

col_attr: Like row_attr, but adding "col_n" and "col_sum" to the "margin" attribute.

lag_attr: If TRUE, adds "lag_n" and "lag_sum" to the "margin" attribute. These are the margin scores for rows, where the date of the column is before (lag) the date of the row. Only possible if date argument is given.

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

verbose: If TRUE, report progress.

Details:

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

This function is called by the document comparison functions (newsflow_compare, delete_duplicates). We only expose it here for additional flexibility, and because it could be useful outside of the purpose of this package.

The output matrix also has an attribute "margin", which contains margin scores (e.g., row_sum) if the row_attr or col_attr arguments are used. The reason for including this is that some values that are normally available in the output of a cross product are broken if certain filter options are used.

If group or date is used, we don’t know how many columns a row has been compared to (normally this is all columns). If a min/max or top_n filter is used, we don’t know the true row sums (and thus row means).

Value

A CsparseMatrix

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_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 term's 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("bi", "tri"),
  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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>voc</td>
<td>A character vector that gives the vocabulary (e.g., colnames of a dtm)</td>
<td></td>
</tr>
<tr>
<td>type</td>
<td>Either &quot;bi&quot; (bigrams) or &quot;tri&quot; (trigrams)</td>
<td></td>
</tr>
<tr>
<td>min_overlap</td>
<td>The minimal overlap percentage. Works together with max_diff to determine required overlap</td>
<td></td>
</tr>
<tr>
<td>max_diff</td>
<td>The maximum number of bi/tri-grams that is different</td>
<td></td>
</tr>
<tr>
<td>pad</td>
<td>If True, pad the left size (ls) and right side (rs) of bi/tri-grams. So, trigrams for &quot;pad&quot; would be: &quot;ls_ls_p&quot;, &quot;ls_p_a&quot;, &quot;p_a_d&quot;, &quot;a_d_rs&quot;, &quot;d_rs_rs&quot;.</td>
<td></td>
</tr>
<tr>
<td>as_lower</td>
<td>If True, ignore case</td>
<td></td>
</tr>
<tr>
<td>same_start</td>
<td>Should terms start with the same character(s)? Given as a number for the number of same characters. (also greatly speeds up calculation)</td>
<td></td>
</tr>
<tr>
<td>drop_non_alpha</td>
<td>If True, ignore non alpha terms (e.g., numbers, punctuation). They will appear in the output matrix, but only with zeros.</td>
<td></td>
</tr>
<tr>
<td>min_length</td>
<td>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.</td>
<td></td>
</tr>
<tr>
<td>allow_asym</td>
<td>If True, the match only needs to be true for at least one term. In practice, this means that &quot;America&quot; would match perfectly with &quot;Southern-America&quot;.</td>
<td></td>
</tr>
<tr>
<td>verbose</td>
<td>If True, report progress</td>
<td></td>
</tr>
</tbody>
</table>
Value

A similarity matrix in the CsparseMatrix format

Examples

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

---

term_day_dist

Calculate statistics for term occurrence across days

Description

Calculate statistics for term occurrence 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.

Examples

tdd = term_day_dist(rnewsflow_dfm, date.var='date')
head(tdd)
tail(tdd)
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 CsparseMatrix
- **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
term_intersect

\[
\begin{align*}
\text{min\_ratio} & \quad \text{The minimum ratio (rwindow score / lwindow score)} \\
\text{smooth} & \quad \text{The smoothing factor (prevents -Inf/Inf ratio)}
\end{align*}
\]

**Value**

A CsparseMatrix

---

**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, sep = " & ", par = NA)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dtm</td>
<td>A quanteda dfm or a CsparseMatrix.</td>
</tr>
<tr>
<td>simmat</td>
<td>A similarity matrix in CsparseMatrix format. For instance, created with term_char_sim</td>
</tr>
<tr>
<td>as_dfm</td>
<td>If True, return as quanteda dfm</td>
</tr>
<tr>
<td>verbose</td>
<td>If True, report progress</td>
</tr>
<tr>
<td>sep</td>
<td>The separator used for pasting the terms</td>
</tr>
<tr>
<td>par</td>
<td>If TRUE, add parentheses to colnames before combining. This is mainly for internal use, as it allows specification if OR (term_union) and AND (term_intersect) operations are combined. If NA, this is based on whether parentheses are present.</td>
</tr>
</tbody>
</table>

**Value**

A CsparseMatrix 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 "|" separator (read as OR).

Usage

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

Arguments

dtm
A quanteda dfm or a CsparseMatrix.
simmat
A similarity matrix in CsparseMatrix 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
If TRUE, add parentheses to colnames before combining. This is mainly for internal use, as it allows specification if OR (term_union) and AND (term_intersect) operations are combined. If NA, this is based on whether parentheses are present.

Value
A CsparseMatrix or quanteda dfm

Examples

dfm = quanteda::tokens(c('That guy Gadaffi','Do you mean Kadaffi?','Nah more like Gadaffel','Not Kadaffel?')) |>
quanteda::dfm()
simmat = term_char_sim(colnames(dfm), same_start=0)
term_union(dfm, simmat, verbose = FALSE)
Index

* datasets
  docnet, 12
  rnewsflow_dfm, 21
as.POSIXct, 6
as_document_network, 2, 6, 18
compare_documents, 2, 3, 9, 18, 19
create_document_network, 6, 12–14, 20, 21
create_queries, 7
delete_duplicates, 9
dfm, 3, 7–9, 19, 26, 28, 29
dfm_tfidf, 3, 4, 19
dfm_weight, 3, 4, 19
directed_network_plot, 10
docnet, 12
document_network_plot, 12
filter_window, 14, 21
format.POSIXct, 13
get_doc_terms, 15
get_overlap_terms, 15
hourdiff_range_thresholds, 16
igraph, 2, 6, 10–12, 14, 17, 18, 20
layout.davidson.harel, 11
network_aggregate, 17
newsflow_compare, 12–14, 18, 20, 21
only_first_match, 20
plot.igraph, 10, 11, 13
rnewsflow_dfm, 21
show_window, 14, 21
tcrossprod_sparse, 22
term_char_sim, 25, 28, 29
term_day_dist, 26
term_innovation, 27
term_intersect, 28
term_union, 29