

# Package: RNewsflow (via r-universe)

August 31, 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

**Repository** <https://kasperwelbers.r-universe.dev>

**RemoteUrl** <https://github.com/kasperwelbers/rnewsflow>

**RemoteRef** HEAD

**RemoteSha** d6909a9c8a971f1e9725159996f50155ab802ce9

## Contents

as_document_network . . . . .	2
compare_documents . . . . .	3
create_document_network . . . . .	6

create_queries . . . . .	7
delete_duplicates . . . . .	9
directed_network_plot . . . . .	10
docnet . . . . .	12
document_network_plot . . . . .	12
filter_window . . . . .	14
get_doc_terms . . . . .	15
get_overlap_terms . . . . .	15
hourdiff_range_thresholds . . . . .	16
network_aggregate . . . . .	17
newsflow_compare . . . . .	18
only_first_match . . . . .	20
rnewsflow_dfm . . . . .	21
show_window . . . . .	21
tcrossprod_sparse . . . . .	22
term_char_sim . . . . .	25
term_day_dist . . . . .	26
term_innovation . . . . .	27
term_intersect . . . . .	28
term_union . . . . .	29

## Index 30

---

as\_document\_network    *Create a document similarity network*

---

### Description

This function can be used to structure the output of the [compare\\_documents](#) function as an igraph network.

### Usage

```
as_document_network(e1)
```

### Arguments

`e1`                    An RNewsflow\_edgelist object, as created with [compare\\_documents](#).

### Value

A network/graph in the [igraph](#) class

### Examples

```
dtm = quanteda::dfm_tfidf(rnewsflow_dfm)
e1 = compare_documents(dtm, date_var='date', hour_window = c(0.1, 36))

g = as_document_network(e1)
g
```

---

compare_documents	<i>Compare the documents in a dtm</i>
-------------------	---------------------------------------

---

## 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

```
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 <a href="#">dfm</a> . Note that it is common to first weight the dtm(s) before calculating document similarity, For this you can use quanteda's <a href="#">dfm_tfidf</a> and <a href="#">dfm_weight</a>
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 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 <a href="#">dfm_tfidf</a> or <a href="#">dfm_weight</a> 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)
e1 = compare_documents(dtm, date_var='date', hour_window = c(0.1, 36))

d = data.frame(text = c('a b c d e',
                        '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 <a href="#">as.POSIXct</a> , 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

**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**

```
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 quantified [dfm](#)

ref_dtm	Optionally, another quanteda <i>dfm</i> . 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 terminolog 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.



**Value**

a list with a query dtm and ref\_dtm. Designed for use in [compare\\_documents](#) using the special ‘query\_lookup’ measure

**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	<i>Delete duplicate (or similar) documents from a document term matrix</i>
-------------------	--

---

**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**

```
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 = F
)
```

**Arguments**

dtm	A quanteda <a href="#">dfm</a> .
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 asymmetrical 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

<code>g</code>	A network/graph in the <a href="#">igraph</a> class
<code>weight_var</code>	The edge attribute that is used to specify the edges
<code>weight_thres</code>	A threshold for weight. Edges below the threshold are ignored
<code>delete_isolates</code>	If TRUE, isolates (i.e. vertices without edges) are ignored.
<code>vertex.size</code>	The size of the vertices/nodes. Defaults to 30. Can be a vector with values per vertex.
<code>vertex.color</code>	Color of vertices/nodes. Default is lightblue. Can be a vector with values per vertex.
<code>vertex.label.color</code>	Color of labels for vertices/nodes. Defaults to black. Can be a vector with values per vertex.
<code>vertex.label.cex</code>	Size of the labels for vertices/nodes. Defaults to 0.7. Can be a vector with values per vertex.
<code>edge.color</code>	Color of the edges. Defaults to grey. Can be a vector with values per edge.
<code>show.edge.labels</code>	Logical. Should edge labels be displayed? Default is TRUE.
<code>edge.label.color</code>	Color of the edge labels. Defaults to black. Can be a vector with values per edge.
<code>edge.label.cex</code>	Size of the edge labels. Defaults to 0.6. Can be a vector with values per edge.
<code>edge.arrow.size</code>	Size of the edge arrows. Defaults to 1. Can only be set globally (igraph might update this at some point)
<code>layout</code>	The igraph layout used to plot the network. Defaults to <a href="#">layout.davidson.harel</a>
<code>...</code>	Arguments to be passed to the <a href="#">plot.igraph</a> 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	<i>Document similarity network for one news agency, and the print and online editions of two newspapers</i>
--------	---

---

**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 [news-flow\\_compare](#).

---

document_network_plot	<i>Visualize (a subcomponent) of the document similarity network</i>
-----------------------	--

---

**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**

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

**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	<i>Filter edges from the document similarity network based on hour difference</i>
---------------	---

---

### 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)
```

### Arguments

g	A document similarity network, as created with <a href="#">newsflow_compare</a> or <a href="#">create_document_network</a>
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, <code>c(-10, 36)</code> 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 ‘ <code>V(g)\$sourcetype == "newspaper"</code> ’ 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 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**

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

**Arguments**

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

**Value**

Nothing... just plots



---

network_aggregate	<i>Aggregate the edges of a network by vertex attributes</i>
-------------------	--

---

### 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

<code>g</code>	A network/graph in the <a href="#">igraph</a> class
<code>by</code>	A character string indicating the vertex attributes by which the edges will be aggregated.
<code>by_from</code>	Optionally, specify different vertex attributes to aggregate the ‘from’ side of edges
<code>by_to</code>	Optionally, specify different vertex attributes to aggregate the ‘to’ side of edges
<code>edge_attribute</code>	Select an edge attribute to aggregate using the function specified in ‘agg_FUN’. Defaults to ‘weight’
<code>agg_FUN</code>	The function used to aggregate the edge attribute
<code>return_df</code>	Optional. If TRUE, the results are returned as a data.frame. This can in particular be convenient if <code>by_from</code> and <code>by_to</code> are used.
<code>keep_isolates</code>	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 <a href="#">dfm</a> . Note that it is common to first weight the dtm(s) before calculating document similarity, For this you can use quanteda's <a href="#">dfm_tfidf</a> and <a href="#">dfm_weight</a>
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 <a href="#">dfm_tfidf</a> or <a href="#">dfm_weight</a> 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 <a href="#">compare_documents</a> .

**Value**

An igraph network.

**Examples**

```
dtm = quanteda::dfm_tfidf(rnewsflow_dfm)
e1 = newsflow_compare(dtm, date_var='date', hour_window = c(0.1, 36))
```

---

only_first_match	<i>Transform document network so that each document only matches the earliest dated matching document</i>
------------------	---

---

### 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 [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

```
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	<i>quanteda dfm for RNewsflow vignette demo</i>
---------------	---

---

**Description**

quanteda dfm for RNewsflow vignette demo

**Usage**

```
rnewsflow_dfm
```

**Format**

```
dfm
```

---

show_window	<i>Show time window of document pairs</i>
-------------	---

---

**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 <a href="#">newsflow_compare</a> or <a href="#">create_document_network</a>
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      *tcrossprod with benefits, for people that like parameters*

---

## 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 things efficiently. While it's 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

```
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`

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 usefull 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 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).

## 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	<i>Find terms with similar spelling</i>
---------------	---

---

### 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 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 <a href="#">dfm</a> . 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	<i>Experimental: Convert dtm scores to a term innovation score, based on changes in term use over time</i>
-----------------	--

---

### 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

<i>m</i>	A <code>CsparseMatrix</code>
<i>date</i>	a character vector that specifies a date for each row in <i>m</i> . If given, only pairs of rows within a given date range (see <i>lwindow</i> , <i>rwindow</i> and <i>date_unit</i> ) are calculated.
<i>m2</i>	Optionally, use a different matrix for calculating the innovation scores. For example, if <i>m</i> is a DTM of press releases, <i>m2</i> can be a DTM of news articles, to see if term usage increased in the news after the press release.
<i>date2</i>	If <i>m2</i> is used, <i>date2</i> has to be used to specify the date for the rows in <i>m2</i> (otherwise date will be ignored)
<i>lwindow</i>	If <i>date</i> (and <i>date2</i> ) are used, <i>lwindow</i> 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 [ <i>date_units</i> ] before.
<i>rwindow</i>	Like <i>lwindow</i> , but for the right side. e.g. an <i>lwindow</i> of -1 and <i>rwindow</i> of 1, with <i>date_unit</i> is "days", means that only rows are matched for which the dates are within a 1 day distance
<i>date_unit</i>	The date unit used in <i>lwindow</i> and <i>rwindow</i> . 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)
<i>min_chi</i>	The minimum chi-square value

min_ratio	The minimum ratio (rwindow score / lwindow score)
smooth	The smoothing factor (prevents -Inf/Inf ratio)

**Value**

A CsparseMatrix

---

term_intersect	<i>Combine terms in a dtm</i>
----------------	-------------------------------

---

**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**

dtm	A quanteda <a href="#">dfm</a> or a CsparseMatrix.
simmat	A similarity matrix in CsparseMatrix format. For instance, created with <a href="#">term_char_sim</a>
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

---

term_union	<i>Combine terms in a dtm</i>
------------	-------------------------------

---

**Description**

Given a dtm and a similarity (adjacency) matrix, group clusters of similar terms ( $\text{simmat} > 0$ ) 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 <a href="#">dfm</a> or a CsparseMatrix.
simmat	A similarity matrix in CsparseMatrix format. For instance, created with <a href="#">term_char_sim</a>
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 ( <code>term_union</code> ) and AND ( <code>term_intersect</code> ) 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](#)