Package 'graphTweets'

March 10, 2016

Type Package

Title Visualise Twitter Interactions

Version 0.3

Date 2016-02-26

Maintainer John Coene <jcoenep@gmail.com>

Description Allows building an edge table from data frame of tweets, also provides function to build vertices (meta-data).

License MIT + file LICENSE

Depends R (>= 3.0.0)

Imports reshape2,
dplyr,
igraph

RoxygenNote 5.0.1

URL https://github.com/JohnCoene/graphTweets

BugReports https://github.com/JohnCoene/graphTweets/issues

Suggests testthat

R topics documented:

`dynamise` *Make a dynamic graph*

Description

Create a dynamic graph from tweets and, optionally, open it in Gephi

Usage

```
dynamise(data, tweets, source, start.stamp, end.stamp = NULL,
  str.length = NULL, write = FALSE, format = "graphml",
  file.dir = getwd(), file.name = "graphTweets", open = FALSE)
```

Arguments

data	data.frame of tweets, typically returned by searchTwitter, required.
tweets	Column name of tweets within data, must be a character string, required.
source	User names or ID column of tweets author, must be a character string, required.
start.stamp	time.stamp to dynamise, typically a date or time, but may also be an interger or a factor, cannot be a character.
end.stamp	The end of the time stamp, or when edges are to leave the graph, defaults to NULL (edges never disappear). See details.
str.length	Defaults to NULL. Shorten length of @tags (see details in getEdges), to a maximum number of characters, optional.
write	if TRUE saves graph as file.
format	if write = TRUE set format of file, defaults to graphml (see details for valid formats).
file.dir	if write = TRUE directory where to save the file, defaults to working directory.
file.name	if write = TRUE name of file
open	if write = TRUE, open = TRUE opens file in https://gephi.org/.

Details

end.stamp: When the edges are to disappear, consider lifetime of a tweet, by default edges stay.

Valid values for format:

- edgelist
- pajek
- ncol
- lgl
- graphml
- dimacs
- gml
- dot
- leda

Author(s)

John Coene <jcoenep@gmail.com>

Examples

```
## Not run:
# load twitteR package to get data
library(twitteR)

# replace with your details
setup_twitter_oauth(consumer_key, consumer_secret, access_token,
                    access_secret)

# fetch tweets on rstats
tweets <- searchTwitter("rstats", n = 200)
tweets <- twListToDF(tweets)

# create dynamic graph
dyn <- dynamise(tweets, tweets = "text", source = "screenName",
                time.stamp = "created")

# create dynamic graph and open in Gephi
dyn <- dynamise(tweets, tweets = "text", source = "screenName",
                time.stamp = "created", write = TRUE, open = TRUE)

## End(Not run)
```

`getEdges` *Build list of edges from tweets*

Description

Builds a table of edges (source, target) from a list of tweets by subsetting @tags from the text.

Usage

```
getEdges(data, tweets, source, str.length = NULL, ...)
```

Arguments

`data`	`data.frame` of tweets, typically returned by `searchTwitter`, required.
`tweets`	Column name of tweets within `data`, must be a `character` string, required.
`source`	User names or ID column of `tweets` author, must be a `character` string, required.
`str.length`	Defaults to `NULL`. Shorten length of @tags (see details), to a maximum number of characters.
`...`	Any other columns to be passed on to the edges.

Details

The edges function takes in a data frame of tweets, typically obtained from the twitter Search or Streaming API, scrapes the content of tweets to subset the @tags subsequently forming a table of edges. @tags are subsets of regular expressions between at-signs (@) and first space (" "). Note that the table of edges returned is meant for a directed graph. Node labels can be shortened using the `str.length` parameters. This is useful for non-latin alphabet where nodes may be wrongly identified (i.e.: Chinese Sina Weibo data).

Author(s)

John Coene <john.coene@gmail.com>

See Also

twitteR and streamR packages wherefrom the data (data) can be obtained.

Examples

```
## Not run:
# load twitteR package to get data
library(twitteR)

# replace with your details
setup_twitter_oauth(consumer_key, consumer_secret, access_token,
                    access_secret)

# fetch tweets on rstats
tweets <- searchTwitter("rstats", n = 200)
tweets <- twListToDF(tweets)

# get edges
edges <- getEdges(data = tweets, tweets = "text", source = "screenName")

# get edges with coordinates
edges <- getEdges(data = tweets, tweets = "text", source = "screenName",
                  "longitude", "latitude")

# load igraph
library(igraph)

# plot
g <- graph.data.frame(edges, directed=TRUE)

plot(g)

## End(Not run)
```

---

getNodes *Build node table from edges*

Description

Get nodes from a data.frame of edges as typically returned by getEdges

Usage

```
getNodes(edges, source = "source", target = "target", ...)
```

Arguments

edges	data.frame of edges as typically returned by getEdges
source	Column of source nodes in edges, must be a character string, defaults to source.
target	Column of target nodes in edges, must be a character string, required.
...	Any other columns to be passed on to the source nodes - will not be applied to target nodes.

Details

One must keep in mind that nodes need to be unique therefore duplicate values (...) are dropped. Also, the meta-data (...), only applies to the source of edges; NAs are generated for target nodes.

Author(s)

John Coene <jcoenep@gmail.com>

Examples

```
## Not run:
# load twitteR package to get data
library(twitteR)

# replace with your details
setup_twitter_oauth(consumer_key, consumer_secret, access_token,
                    access_secret)

# fetch tweets on rstats
tweets <- searchTwitter("rstats", n = 200)
tweets <- twListToDF(tweets)

# get edges
edges <- getEdges(data = tweets, tweets = "text", source = "screenName")

# get nodes
nodes <- getNodes(edges)

# load igraph
library(igraph)

# plot
g <- graph.data.frame(edges, directed = TRUE, vertices = nodes)

plot(g)

## End(Not run)
```

graphTweets **graphTweets** *visualise Twitter Interactions.*

Description

- getEdges - get edges from tweets
- getNodes - add meta-data to vertices
- dynamise - create dynamic graphs

Examples

```
## Not run:
# authenticate
token <- twitteR::setup_twitter_oauth(consumer_key, consumer_secret,
                                      access_token, access_secret)

# search tweets
tweets <- twitteR::searchTwitter("rstats", n = 200)

# unlist to data.frame
tweets <- twitteR::twListToDF(tweets)

# load graphTweets
library(graphTweets)

# get edges
edges <- getEdges(data = tweets, tweets = "text", source = "screenName")

# load igraph
library(igraph)

# plot
g <- graph.data.frame(edges, directed=TRUE)
plot(g)

# add attributes to vertices
edges <- getEdges(data = tweets, tweets = "text", source = "screenName",
                  "retweetCount")
nodes <- getNodes(edges, source = "source", target = "target",
                  "retweetCount")

g <- graph.data.frame(edges, directed=TRUE, vertices = nodes)

plot(g, vertex.size = V(g)$retweetCount)

# create dynamic graph and open in Gephi
dyn <- dynamise(tweets, tweets = "text", source = "screenName",
                time.stamp = "created", write = TRUE, open = TRUE)

## End(Not run)
```

Index