Collecting and Analyzing Social Media Data

Tiago Ventura | Center for Social Media and Politics | NYU

Big Data for Development and Governance

10/21/2022

About me

Tiago Ventura

Postdoc at Center for Social Media and Politics, NYU

Incoming Assistant Professor, McCourt School, Georgetown University

@TiagoVentura_ TiagoVentura venturat@umd.edu https://tiagoventura.rbind.io/

Plans for the Workshop

Twitter Data

- Collecting data using the Academic Access through academictwitteR.
- Quick introduction to network analysis with Twitter data

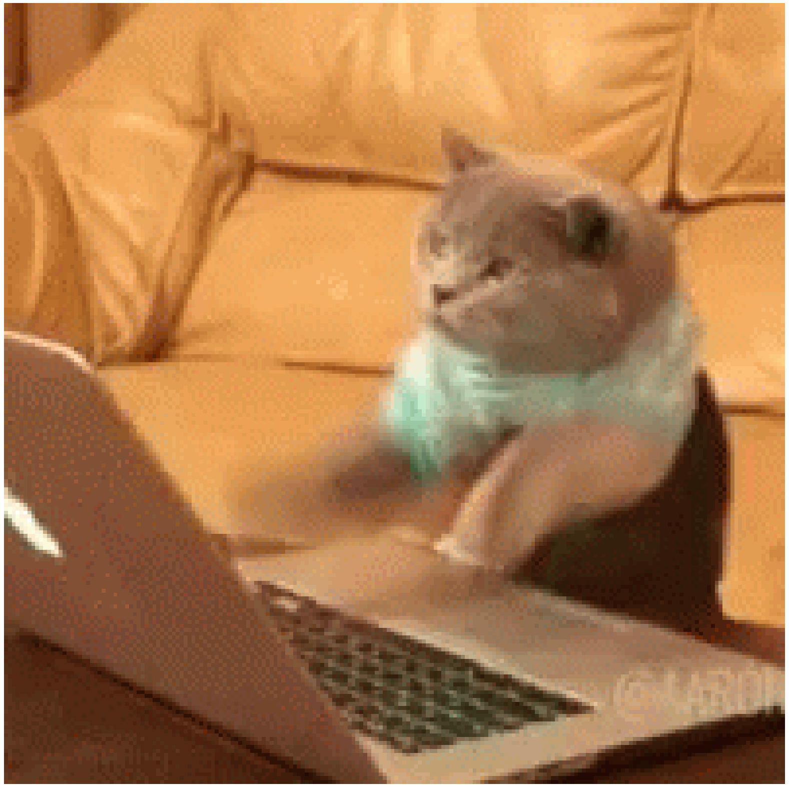
Youtube Data.

- Python Library developed by Megan Brown, Senior Engenieer at the Center for Social Media and Politics at NYU, and some other colleagues.

Telegram Data

- Python module Telethon.

One-hour workshop



Some assumptions

- Assume some knowledge of R and Python.
- We will not go through authentication with the APIs (instructions are provided for you to go through it later).
- You can follow the code in the notebooks. However, the best approach is just to run this later by yourselve, with the proper access to the APIs.
- Particularly for Youtube and Telegram, I will just showcase others' people library to access data from these platforms.

Logistics

All materials are available in the Github repo for the workshop: https://github.com/TiagoVentura/workshop_big_data_conference.

You can just clone all the files from there.

More:

- Slides: Twitter, Youtube, Telegram
- Notebooks: Twitter, Youtube, Telegram

Gathering and Analyzing Twitter Data

Getting Access to the Twitter APIs.

- Apply for a Twitter developer account.
- Apply to the academic research product track.
- Save your keys in a local file in your computer.

Standard Access

- Search for Tweets from the last 7 days.
- Stream Tweets in real-time
- Get Tweets from a user’s timeline (up to 3200 most recent Tweets)
- Build the full Tweet objects from a Tweet ID, or a set of Tweet IDs
- Look up follower relationships

Academic Research product track

- Ability to get historical Tweets.
- Cap of 10 million Tweets per month
- More advanced filter options to return relevant data.

Collecting Twitter Data

- **For R users:** academictwitteR package developed by Chris Barrie.
- **For Python User:** check the library Twarc.

Access tweets from the archive

Load Packages

```r
# Call packages using pacman
#install.packages("pacman")
pacman::p_load(here, jsonlite, tidyverse, academictwitteR)
```

Add your API Key

```r
# Using Academic Twitter to add yourkey
set_bearer()
```

get_all_tweets

```r
# Using Academic Twitter to add yourkey
# Collect data
tweets <-
  get_all_tweets(
    query = "(eleicoes2022 OR lula OR bolsonaro OR ciro OR tebet)",
    start_tweets = "2022-10-01T00:00:00Z", #start time
    end_tweets = "2022-10-04T00:00:00Z", #end time
    file = "br_elections", # file to save
    data_path = "data_br/", # folder where all data as jsons will be stores
    n = 200000, # number of tweets
    lang = "pt"
  )
```

get_all_tweets

```r
# Using Academic Twitter to add yourkey
# Collect data
tweets <-
  get_all_tweets(
    query = "(eleicoes2022 OR lula OR bolsonaro OR ciro OR tebet)",
    start_tweets = "2022-10-01T00:00:00Z", #start time
    end_tweets = "2022-10-04T00:00:00Z", #end time
    file = "br_elections", # file to save
    data_path = "data_br/", # folder where all data as jsons will be stores
    n = 200000, # number of tweets
    lang = "pt"
  )
```

get_all_tweets

```r
# Using Academic Twitter to add yourkey
# Collect data
tweets <-
  get_all_tweets(
    query = "(eleicoes2022 OR lula OR bolsonaro OR ciro OR tebet)",
    start_tweets = "2022-10-01T00:00:00Z", #start time
    end_tweets = "2022-10-04T00:00:00Z", #end time
    file = "br_elections",
    data_path = "data_br/",
    n = 200000, # number of tweets
    lang = "pt"
  )
```

get_all_tweets

```r
# Using Academic Twitter to add yourkey
# Collect data
tweets <-
  get_all_tweets(
    query = "(eleicoes2022 OR lula OR bolsonaro OR ciro OR tebet)",
    start_tweets = "2022-10-01T00:00:00Z", #start time
    end_tweets = "2022-10-04T00:00:00Z", #end time
    file = "br_elections", # file to save
    data_path = "data_br/", # folder where all data as jsons will be stores
    n = 200000,
    lang = "pt"
  )
```

Where does the data live?

bind_tweets: tidy

```r
# data processing
tweets_tidy <- bind_tweets("./data_br", output_format = "tidy")
tweets_tidy
```

```
## # A tibble: 6 × 31
##   tweet_id    user_…¹ text  possi…² conve…³ lang  source creat…⁴ autho…⁵ in_re…⁶
##   <chr>       <chr>   <chr> <lgl>   <chr>   <chr> <chr>  <chr>   <chr>   <chr>
## 1 1577066499… cainsw… "RT … FALSE   157706… pt    Twitt… 2022-1… 809471… <NA>
## 2 1577066498… nandam… "RT … FALSE   157706… pt    Twitt… 2022-1… 422691… <NA>
## 3 1577066498… fran51… "RT … FALSE   157706… pt    Twitt… 2022-1… 133561… <NA>
## 4 1577066498… juliam… "RT … FALSE   157706… pt    Twitt… 2022-1… 839520… <NA>
## 5 1577066497… Comerc… "@Au… FALSE   157706… pt    Twitt… 2022-1… 155826… 152441…
## 6 1577066497… caralh… "RT … FALSE   157706… pt    Twitt… 2022-1… 136448… <NA>
## # … with 21 more variables: user_name <chr>, user_created_at <chr>,
## #   user_location <chr>, user_verified <lgl>, user_description <chr>,
## #   user_protected <lgl>, user_pinned_tweet_id <chr>,
## #   user_profile_image_url <chr>, user_url <chr>, retweet_count <int>,
## #   like_count <int>, quote_count <int>, user_tweet_count <int>,
## #   user_list_count <int>, user_followers_count <int>,
## #   user_following_count <int>, sourcetweet_type <chr>, sourcetweet_id <chr>, …
## # ℹ Use `colnames()` to see all variable names
```

bind_tweets: json

```r
# examing the data
tweets_raw <- bind_tweets("./data_br",
                output_format = "raw")
str(tweets_raw, max.level=1)
```

```
## List of 27
##  $ tweet.entities.mentions           : tibble [215,630 × 5] (S3: tbl_df/tbl/data.frame)
##  $ tweet.entities.annotations        : tibble [386,283 × 6] (S3: tbl_df/tbl/data.frame)
##  $ tweet.entities.urls               : tibble [32,703 × 12] (S3: tbl_df/tbl/data.frame)
##  $ tweet.entities.hashtags           : tibble [10,405 × 4] (S3: tbl_df/tbl/data.frame)
##  $ tweet.entities.cashtags           : tibble [3 × 4] (S3: tbl_df/tbl/data.frame)
##  $ tweet.public_metrics.retweet_count : tibble [200,062 × 2] (S3: tbl_df/tbl/data.frame)
##  $ tweet.public_metrics.reply_count   : tibble [200,062 × 2] (S3: tbl_df/tbl/data.frame)
##  $ tweet.public_metrics.like_count    : tibble [200,062 × 2] (S3: tbl_df/tbl/data.frame)
##  $ tweet.public_metrics.quote_count   : tibble [200,062 × 2] (S3: tbl_df/tbl/data.frame)
##  $ tweet.attachments.media_keys      : tibble [200,062 × 2] (S3: tbl_df/tbl/data.frame)
##  $ tweet.attachments.poll_ids        : tibble [200,062 × 2] (S3: tbl_df/tbl/data.frame)
##  $ tweet.geo.place_id                : tibble [200,062 × 2] (S3: tbl_df/tbl/data.frame)
##  $ tweet.geo.coordinates             : tibble [200,062 × 3] (S3: tbl_df/tbl/data.frame)
##  $ tweet.withheld.country_codes      : tibble [200,062 × 2] (S3: tbl_df/tbl/data.frame)
##  $ tweet.withheld.copyright          : tibble [200,062 × 2] (S3: tbl_df/tbl/data.frame)
##  $ tweet.edit_history_tweet_ids      : tibble [200,062 × 2] (S3: tbl_df/tbl/data.frame)
##  $ tweet.referenced_tweets           : tibble [184,799 × 3] (S3: tbl_df/tbl/data.frame)
##  $ tweet.main                        :'data.frame':    200062 obs. of  9 variables:
##  $ user.public_metrics.followers_count: tibble [337,098 × 2] (S3: tbl_df/tbl/data.frame)
##  $ user.public_metrics.following_count: tibble [337,098 × 2] (S3: tbl_df/tbl/data.frame)
##  $ user.public_metrics.tweet_count    : tibble [337,098 × 2] (S3: tbl_df/tbl/data.frame)
##  $ user.public_metrics.listed_count   : tibble [337,098 × 2] (S3: tbl_df/tbl/data.frame)
```

Network Analysis with Twitter Data

Many different ways you can analyze Twitter data: the text, the images, the geolocation, links, among many other things.

A popular way is to to look at the user connections using some sort of network models. Not limited to Twitter data.

A network has two core elements: nodes and edges. On Twitter this means:

- Nodes are Twitter users
- Edges are any sort of connections these users make. A reply, a friendship, or the most common, a retweet.

igraph package to analyze network data in R.

Intro to Network Analysis in R

Step 1: Filter Nodes

Code

```
# Filter retweets
tweets_tidy_rt <- tweets_tidy %>%
                  filter(!is.na(sourcetweet_type))

# Visualize the dta
tweets_tidy_rt %>%
  select(user_username,
  sourcetweet_author_id) %>%
  head()
```

Output

```
## # A tibble: 6 × 2
##   user_username   sourcetweet_author_id
##   <chr>           <chr>
## 1 cainsworts      1534722153819643906
## 2 nandamattosbh   18880621
## 3 fran51995877    26752656
## 4 juliam3ndes     863806721696858112
## 5 caralho_modesti 2876592790
## 6 carolfcarneiro  44481447
```

Step 2: Create a edge list

Code

```
# Create a edge list
data <- cbind(tweets_tidy_rt$author_id,
        tweets_tidy_rt$sourcetweet_author_id)
```

Output

```
##      [,1]                  [,2]
## [1,] "809471355116781568"  "1534722153819643906"
## [2,] "42269111"            "18880621"
## [3,] "1335618427852124163" "26752656"
## [4,] "839520909807521793"  "863806721696858112"
## [5,] "136448124"           "2876592790"
## [6,] "108719485"           "44481447"
```

Step 3: Create your network structure

Code

```r
pacman::p_load(igraph)

# Create an empty network

net <- graph.empty()

# Add nodes
net <- add.vertices(net,
        length(unique(c(data))), # number of nodes
        name=as.character(unique(c(data)))) # unique names

# Add edges
net <- add.edges(net, t(data))

# summary
summary(net)
```

Step 3: Create your network structure

Code

```
pacman::p_load(igraph)

# Create an empty network

net <- graph.empty()

# Add nodes
net <- add.vertices(net,
        length(unique(c(data))), # number of nodes
        name=as.character(unique(c(data)))) # unique names

# Add edges
net <- add.edges(net, t(data))

# summary
summary(net)
```

Step 3: Create your network structure

Code

```
pacman::p_load(igraph)

# Create an empty network

net <- graph.empty()

# Add nodes
net <- add.vertices(net,
        length(unique(c(data))), # number of nodes
        name=as.character(unique(c(data)))) # unique names

# Add edges
net <- add.edges(net, t(data))

# summary
summary(net)
```

Output

```
## IGRAPH 4afcf56 DN-- 79886 154583 --
## + attr: name (v/c)
```

Step four: Add information to your network object

Edge level (`E(object)`) or Node leve (`V(object)`).

Code

```r
library(urltools)

# Edges
E(net)$text <- tweets_tidy_rt$text
E(net)$idauth <- tweets_tidy_rt$sourcetweet_author_id
E(net)$namehub <- tweets_tidy_rt$user_username


# Capturing hashtags
E(net)$hash <- str_extract_all(tweets_tidy_rt$text,
                               "#\\S+")
```

Accessing the raw

```r
# grab expanded and unwound_url
entities <- tweets_raw$tweet.entities.urls
entities
```

```
## # A tibble: 32,703 × 12
```

Network Statistics, Communities and Layout

Two very common concepts in network science are in-degree and out-degree.

- In-degree refers to how many links pointing to themselves the user has.
- Out-degree means how many edges originated at this user.

A user is called an authority when their in-degree is high.

- A user is called a hub when its out-degree is high, as this user retweets very often.

Degree Statistics

```
# Calculate in degree and out degree
V(net)$outdegree<-degree(net, mode="out")
V(net)$indegree<-degree(net, mode="in")
summary(net)
```

```
## IGRAPH 4afcf56 DN-- 79886 154583 --
## + attr: name (v/c), outdegree (v/n), indegree (v/n), text (e/c), idauth
## | (e/c), namehub (e/c), hash (e/x), domain (e/c)
```

Layout

```r
l <- layout_with_fr(net, grid = c("nogrid"))
#saveRDS(l, "layout.rds")
head(l)
```

```
##            [,1]       [,2]
## [1,] -102.96401  216.91269
## [2,] -178.82523  158.25089
## [3,]   52.34920   81.01076
## [4,]  -11.32539 -139.13169
## [5,]   51.89335  -56.94802
## [6,]   29.44408  -99.62837
```

Communities

```r
my.com.fast <- walktrap.community(net)
str(my.com.fast, max.level = 1)
```

```
## Class 'communities'  hidden list of 6
##  $ merges    : num [1:77186, 1:2] 58675 61627 60095 58720 58731 ...
##  $ modularity: num [1:79886] 0 -0.00127 -0.00127 -0.00126 -0.00125 ...
##  $ membership: num [1:79886] 1689 2004 11 169 175 ...
##  $ names     : chr [1:79886] "809471355116781568" "42269111" "1335618427852124163" "83952090980752
##  $ vcount    : int 79886
##  $ algorithm : chr "walktrap"
```

Add the layout and membership to your igraph object.

```
V(net)$l1 <- l[,1]
V(net)$l2 <- l[,2]
V(net)$membership <- my.com.fast$membership
```

What are the largest communities?

Code

```
comunidades<- data_frame(membership=V(net)$membership)

comunidades %>%
    count(membership) %>%
    ungroup() %>%
    mutate(total=sum(n),
           prop_community=n/total) %>%
    arrange(desc(n)) %>%
    top_n(5)
```

Output

```
## # A tibble: 5 × 2
##   membership     n
##        <dbl> <int>
## 1         11 18272
## 2          4 18077
## 3          8  7951
## 4         13  2923
## 5          2  1165
```

Who are the main authorities in each community?

Code

```
# Create an datafram for the authoritiew
authorities <- data_frame(name=V(net)$name,
                ind=V(net)$indegree,
                membership=V(net)$membership) %>%
                filter(membership==11|
                       membership==4|
                       membership==8) %>%
                group_by(membership) %>%
                arrange(desc(ind)) %>%
                slice(1:10)
```

Authorities names

```
# I will get only from the 100 most retweeted to save some time.
users_most_retweets <-authorities %>%
                      mutate(data_user=map(name,
                             get_user_profile)) %>%
                      unnest()
```

```
## Processing from 1 to 1
## Processing from 1 to 1
```

Who are the main authorities in each community?

ggplot code

```r
# Main Communities
ggplot(users_most_retweets %>%
         filter(membership=="4"),
         aes(x=reorder(username,
                ind,
             fill=membership),
                     y=ind)) +
    geom_histogram(stat="identity", width=.5, color="black") +
    coord_flip() +
    xlab("") + ylab("") +
    theme_minimal(base_size = 12) +
    theme(plot.title = element_text(size = 22, face = "bold"),
          axis.title=element_text(size=16),
          axis.text = element_text(size=12, face="bold")) +
    facet_grid(~membership)
```

Community I

Community II

Visualizing communities

Function to Plot Network

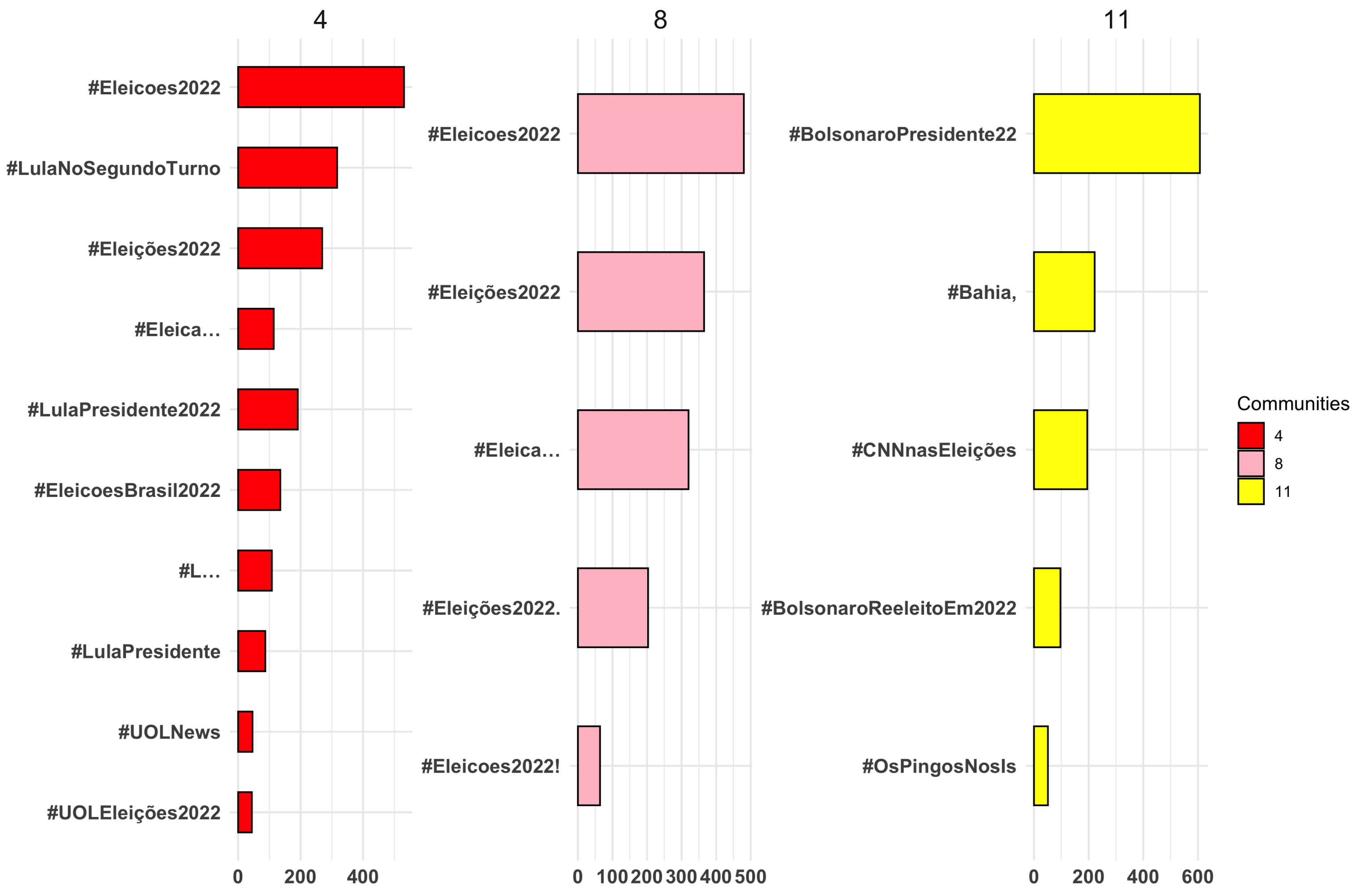
```
# A function with the density. Nice to visualize as well.
my.den.plot <- function(l=l,new.color=new.color, ind=ind, legend, color){
  library(KernSmooth)
  est <- bkde2D(l, bandwidth=c(10, 10))
  plot(l,cex=log(ind+1)/4, col=new.color, pch=16, xlim=c(-160,140),ylim=c(-140,160), xlab=""
   legend("topright", c(legend[1],legend[2], legend[3]), pch = 17:19, col=c(color[1], color[
  contour(est$x1, est$x2, est$fhat,  col = gray(.6), add=TRUE)
}
```

Function to Plot Network

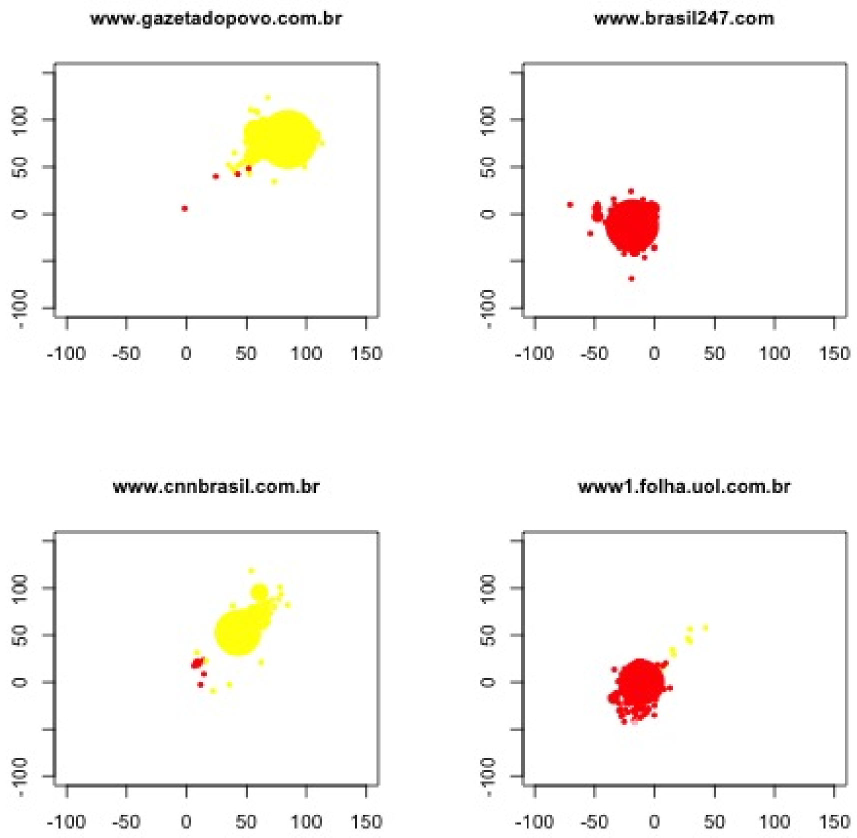
```
# Colors for each community

# Building a empty containes
temp <- rep(1,length(V(net)$membership))
new.color <- "white"
new.color[V(net)$membership==11] <- "Yellow"
new.color[V(net)$membership==8] <- "pink"
new.color[V(net)$membership==4] <- "red"

# Add color
V(net)$new.color <- new.color
```

Network Plot

Hashtags by communities



Sharing news on Twitter



From News Sharing, Gatekeeping, and Polarization: A Study of the #Bolsonaro Election

Other APIs endpoints

Most of our work with the Twitter API happens with the capacity to query the API with search terms. For this reason, the search (and filter for live data collection) endpoints are the most popular.

However, there are a few other endpoints from the Twitter API that can also be very useful for research puporses. Let's walk through them briefly.

Getting user id

Imagine a research in which you have the Twitter accounts of elites, and you want to collect their Twitter data. The first step is to collect their ids.

```r
# getting some Twitter Ids
pelosi <- get_user_id("SpeakerPelosi")
pelosi
```

```
## SpeakerPelosi
##    "15764644"
```

Getting whom a user follows

```
pelosi_network <- get_user_following(pelosi)
```

```
## Processing 15764644
## Total data points:  429
## This is the last page for  15764644 : finishing collection.
```

```
glimpse(pelosi_network)
```

```
## Rows: 429
## Columns: 14
## $ url               <chr> "https://t.co/v695zCnmxN", "https://t.co/4xG26ktTyt"…
## $ protected         <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL…
## $ public_metrics    <df[,4]> <data.frame[26 x 4]>
## $ description       <chr> "Representative for Ohio’s Eleventh Congressional…
## $ verified          <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE…
## $ created_at        <chr> "2021-11-04T22:01:33.000Z", "2010-06-16T17:20:23.000…
## $ username          <chr> "RepShontelBrown", "RepAlGreen", "RepJoeGarcia", "Re…
## $ id                <chr> "1456381091598700556", "156333623", "937801969", "11…
## $ name              <chr> "Rep. Shontel Brown", "Congressman Al Green", "Rep. …
## $ entities          <df[,2]> <data.frame[26 x 2]>
## $ pinned_tweet_id   <chr> "1463532439456952323", "1422610928756043778", NA, NA…
## $ profile_image_url <chr> "https://pbs.twimg.com/profile_images/14707589214…
## $ location          <chr> NA, "Houston, TX & Washington, DC", "Miami, Florida"…
## $ from_id           <chr> "15764644", "15764644", "15764644", "15764644", "157…
```

Estimate user ideology

Code

```r
#devtools::install_github("pablobarbera/twitter_ideology/pkg/tweetscores")
library(tweetscores)
results <- estimateIdeology("SpeakerPelosi", pelosi_network$id, verbose = FALSE)
```

Output

```r
plot(results)
```

User timeline

```r
pelosi_tl = get_user_timeline(pelosi,
                              start_tweets = "2022-01-01T00:00:00Z",
                               end_tweets = "2022-10-22T00:00:00Z",
                              n=100) #limit
```

```
## user:  15764644
## Total pages queried: 1 (tweets captured this page: 100).
## Total tweets captured now reach 100 : finishing collection.
```

```r
glimpse(pelosi_tl)
```

```
## Rows: 100
## Columns: 15
## $ context_annotations    <list> [<data.frame[7 x 2]>], [<data.frame[14 x 2]>],…
## $ referenced_tweets      <list> [<data.frame[1 x 2]>], <NULL>, <NULL>, <NULL>,…
## $ entities               <df[,4]> <data.frame[26 x 4]>
## $ id                     <chr> "1582556778608218118", "1582492989015805953"…
## $ text                   <chr> "Anna May Wong was a dazzling, trailblazing tal…
## $ source                 <chr> "Twitter for iPhone", "Twitter Web App", "Twitt…
## $ possibly_sensitive     <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE…
## $ conversation_id        <chr> "1582556778608218118", "1582492989015805953", "…
## $ created_at             <chr> "2022-10-19T02:18:33.000Z", "2022-10-18T22:05:0…
## $ lang                   <chr> "en", "en", "en", "en", "en", "en", "en", "en",…
## $ edit_history_tweet_ids <list> "1582556778608218118", "1582492989015805953", "…
## $ author_id              <chr> "15764644", "15764644", "15764644", "15764644"…
## $ public_metrics         <df[,4]> <data.frame[26 x 4]>
## $ attachments            <df[,1]> <data.frame[26 x 1]>
```

Tweets liked by an user

```
pelosi_likes = get_liked_tweets(pelosi)
```

```
## Processing 15764644
## Total data points:  11
## Total data points:  12
## This is the last page for  15764644 : finishing collection.
```

```
glimpse(pelosi_likes) # she mostly liked her own tweets
```

```
## Rows: 12
## Columns: 16
## $ id                      <chr> "1230592378790129664", "819738961887264768", "8…
## $ edit_history_tweet_ids <list> "1230592378790129664", "819738961887264768", "…
## $ created_at              <chr> "2020-02-20T20:37:41.000Z", "2017-01-13T02:52:5…
## $ public_metrics          <df[,4]> <data.frame[12 x 4]>
## $ text                    <chr> "That moment when you hear @presmccartney sa…
## $ entities                <df[,4]> <data.frame[12 x 4]>
## $ lang                    <chr> "en", "en", "en", "en", "en", "en", "en", "en",…
## $ context_annotations     <list> [<data.frame[5 x 2]>], <NULL>, <NULL>, [<dat…
## $ author_id               <chr> "17025399", "281593711", "39547629", "774337933…
## $ conversation_id         <chr> "1230592378790129664", "819738961887264768", "…
## $ source                  <chr> "Buffer", "Twitter Web Client", "Twitter for iP…
## $ possibly_sensitive      <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE…
## $ attachments             <df[,1]> <data.frame[12 x 1]>
## $ in_reply_to_user_id     <chr> NA, NA, NA, "15764644", NA, NA, NA, NA, NA, NA,…
## $ referenced_tweets       <list> <NULL>, <NULL>, <NULL>, [<data.frame[1 x 2]>], …
## $ from_id                 <chr> "15764644", "15764644", "15764644", "1576464…
```

Question?

That's a wrap for Twitter data

See here the link for the Youtube slides