Using API's from R: Twitter, Facebook, NY Times

Wouter van Atteveldt

Many Internet data sources such as Twitter and Facebook offer a public API (application programming interface) that can be used to easily (and legally) retrieve data from their site.

This tutorial will show how to use a selection of R client packages that are designed to query the APIs of Twitter, Facebook, and the NY Times. At the end, there will also be an example of querying an API without using a special R client.

Twitter

To access twitter we will use the twitteR packages. The following code will load these packages, installing them if needed. You can skip the 'install_github' steps after the first time.

```
install.packages("devtools") # only needed once
devtools::install_github("geoffjentry/twitteR") # only needed once
library(twitteR)
```

Connecting to Twitter

First, you need to get a number of tokens (a kind of passwords) from twitter:

1. Sign in to twitter at http://twitter.com
2. Go to https://apps.twitter.com/ and 'create a new app'
3. After filling in the required information, go to 'keys and access tokens'
4. Select 'create access token', and refresh
5. Create variables with the consumer key, consumer secret, access token, and access token secret:

```
tw_token = '...'
tw_token_secret = '...'
tw_consumer_key = "..."
tw_consumer_secret = "..."
```

Now you can connect using the setup_twitter_oauth function:

```
setup_twitter_oauth(tw_consumer_key, tw_consumer_secret, tw_token, tw_token_secret)
```

```
## [1] "Using direct authentication"
```

Searching twitter

Please see the documentation for the Twitter API and the twitteR package for all the possibilities of the API. As the following simple example shows, you can search for keywords and get a list or results

```
tweets = searchTwitteR("#Trump2016", resultType="recent", n = 10)


tweets[[1]]
```

```
## [1] "freedoms411: RT @starknightz: PLZ Share - RT\nLIVE Stream (6-29-2016) 4:00PMEDT\nDonald Trump ra
```

```
tweets[[1]]$text
```

```
## [1] "RT @starknightz: PLZ Share - RT\nLIVE Stream (6-29-2016) 4:00PMEDT\nDonald Trump rally 4:00PMED
```

To make it easier to manipulate the tweets, we can convert them from a list of `status` objects to a data.frame, for which we use the `ldply` (list-dataframe-ply) function from the plyr package, taking advantage of the fact that `as.data.frame` works on a single status object:

```
tweets = plyr::ldply(tweets, as.data.frame)
nrow(tweets)
```

```
## [1] 10
```

```
names(tweets)
```

```
##  [1] "text"          "favorited"     "favoriteCount" "replyToSN"
##  [5] "created"       "truncated"     "replyToSID"    "id"
##  [9] "replyToUID"    "statusSource"  "screenName"    "retweetCount"
## [13] "isRetweet"     "retweeted"     "longitude"     "latitude"
```

Facebook

For querying facebook, we can use Pable Barbera's `Rfacebook` package, which we install directly from github:

```
devtools::install_github("pablobarbera/Rfacebook", subdir="Rfacebook")
library(Rfacebook)
```

To get a permanent facebook oath token, there are a number of steps you need to take

1. Log on to facebook and go to https://developers.facebook.com/apps
2. Create an app with the 'basic settings'
3. Copy the the app id and app secret, and run fbOAth
4. This will prompt you to paste a (localhost) url into your app settings. Add this setting in facebook app settings under products -> facebook login.
5. Next, authenticate in your web browser, and accept the permissions.
6. Now you have a `fb_token` that you can use for authentication in the API, which you can save for reuse

```
fb_app_id = '...'
fb_app_secret = '...'
fb_token = fbOAuth(fb_app_id, fb_app_secret)
saveRDS(fb_token, "fb_token.rds")
```

Now, we can use the facebook API, e.g. to get all stories posted to the NY Times public facebook page:

```
p = getPage(page="nytimes", token=fb_token)
```

```
## 25 posts
```

```
head(p)
```

from_id	from_name	message
5281959998	The New York Times	It’s hard to believe that Simone Biles, who is already so good, is getting even better.
5281959998	The New York Times	The Supreme Court struck down a Texas law that would have forced dozens of aborti
5281959998	The New York Times	In a big win for pro-choice advocates, the Supreme Court struck down regulations in
5281959998	The New York Times	With the Brexit vote, Britons wanted to make their world smaller: “We will have few
5281959998	The New York Times	One finding from the survey: Blacks were more than 4 times as likely as whites to say
5281959998	The New York Times	The best of The New York Times Food.

We can also get all comments on a post, e.g. from the first post:

```
post = getPost(p$id[1], token=fb_token)
names(post$comments)
```

```
## [1] "from_id"      "from_name"    "message"      "created_time"
## [5] "likes_count"  "id"
```

NYTimes: package rtimes

For the NY Times, we can use the `rtimes` package. Like the other APIs, we first need to get a key, which you can request at

```
install.packages("rtimes")
library('rtimes')
nyt_api_key = '...'
options(nytimes_as_key = nyt_api_key)
```

Now, we can use the `as_search` command to search for articles

```
res <- as_search(q="trump", begin_date = "20160101", end_date = '20160501')
names(res)
```

```
## [1] "copyright" "meta"      "data"
```

```
res$meta
```

```
##   hits time offset
## 1 5332   26      0
```

This will have returned the first ‘page’ of 10 results, which we can convert to a data frame using `ldply` from the `plyr` package:

```
arts = plyr::ldply(res$data, function(x) c(headline=x$headline$main, date=x$pub_date))
head(arts)
```

headline	date
Donald Trump's Aging Air Fleet Gives His Bid, and His Brand, a Lift	2016-04-24T00:00:00Z
In Campaign and Company, Ivanka Trump Has a Central Role	2016-04-17T00:00:00Z
Donald Trump Settled a Real Estate Lawsuit, and a Criminal Case Was Closed	2016-04-06T00:00:00Z
If Not Trump, What?	2016-04-29T00:00:00Z
Protest Turns Rowdy as Donald Trump Appears at California G.O.P. Convention	2016-04-30T00:00:00Z
Run on a Ticket With Donald Trump? No, Thanks, Many Republicans Say	2016-05-01T00:00:00Z

APIs and rate limits

Most APIs limit how many requests you can make per minute, hour, or day. For example, twitter by default allows 180 search queries per 15 minutes, while NY Times allows 1000 requests per day.

Most APIs also have a way of checking how many queries you have 'left', for example for twitter you can use the following:

```
twitteR::getCurRateLimitInfo("search")
```

```
##         resource limit remaining               reset
## 1 /search/tweets   180       179 2016-06-29 09:11:41
```

The `twitteR` package has built-in functionality to retry if it reaches the rate limit, and will automatically divide large requests into smaller requests. For example, if you ask for 1000 results, it will do 10 requests of 100 results each (the maximum per request).

If such functionality is not available in the client library, you will need to work around these limits yourself (if needed). For example, the `rtimes` package only retrieves a single page per API call. To download all results for a call, we need to loop over the results ourselves.

The first step is finding out how many hits there are, for example for the front page articles mentioning Syria in January:

```
res <- as_search(q="syria", fq='section_name:Front Page', begin_date = "20160101", end_date = '20160131
res$meta
```

```
##   hits time offset
## 1   38   51      0
```

So, there are 39 hits, i.e. 4 pages. We can query all pages by using a for loop, adding the pages to a list:

```
npages = ceiling(res$meta$hits / 10)
results = res$data
for (p in 1:(npages-1)) {
  res <- as_search(q="syria", fq='section_name:Front Page', begin_date = "20160101", end_date = '201601
  results = c(results, res$data)
}
arts = plyr::ldply(results, function(x) c(headline=x$headline$main, date=x$pub_date))
nrow(arts)
```

```
## [1] 38
```

```r
tail(arts)
```

```
##                                                           headline
## 33             Transcript of the Democratic Presidential Debate
## 34 Tumultuous 1st Year for Saudi King Salman's 'Decisive' Reign
## 35       Deep in Colombian Jungle, Peace Looms at Rebel Hideout
## 36                 Transcript of Republican Presidential Debate
## 37        Transcript of the Main Republican Presidential Debate
## 38     Transcript of the Preliminary G.O.P. Presidential Debate
##                    date
## 33 2016-01-18T00:00:00Z
## 34 2016-01-23T12:14:59Z
## 35 2016-01-19T00:03:45Z
## 36 2016-01-15T00:00:00Z
## 37 2016-01-29T00:00:00Z
## 38 2016-01-29T00:00:00Z
```

(Note that appending to the list every iteration is not very efficient, but in this case the bottleneck is almost certainly the API call, so there is little to gain in optimizing this)

API access without client library

For many popular APIs, such as Twitter, Facebook, and NY Times, an R client library already exists. However, if this doesn't exist it is relatively easy to query an API directly using HTTP calls, for example using the r `httr` package.

The NY Times API is relatively easy, so it's a good case to show how to build an API client 'from scratch'. To build your own API client, the first step is to have a look at the API documentation for the NY Times Article Search API.

This tells us that we need to do a GET request to the articlesearch end point, specifying at least an `api-key` and a query `q`:

```r
library(httr)
url = 'https://api.nytimes.com/svc/search/v2/articlesearch.json'
r = httr::GET(url, query=list("api-key"=nyt_api_key, q="clinton"))
status_code(r)
```

```
## [1] 200
```

The status code 200 indicates "OK", other status codes generally indicate a problem, such as an invalid API key (search for 'HTTP Status codes' for an overview) The results are retrieved as a json-dictionary, which is accessible in R as a list through the `content` function in `httr`, which identifies the data type based on the headers and converts it. The API documentation linked above contains a list of these fields, but you can also inspect the list itself from R:

```r
result = content(r)
names(result)
```

```
## [1] "response"  "status"    "copyright"
```

```r
names(result$response$docs[[1]])
```

```
##  [1] "web_url"           "snippet"           "lead_paragraph"
##  [4] "abstract"          "print_page"        "blog"
##  [7] "source"            "multimedia"        "headline"
## [10] "keywords"          "pub_date"          "document_type"
## [13] "news_desk"         "section_name"      "subsection_name"
## [16] "byline"            "type_of_material"  "_id"
## [19] "word_count"        "slideshow_credits"
```

```r
result$response$docs[[1]]$headline
```

```
## $main
## [1] "Possible Conflict at Heart of Clinton Foundation"
##
## $content_kicker
## [1] "Letter From Washington"
##
## $kicker
## [1] "Letter From Washington"
```

We can create a data frame of all articles with the `ldply` function from the `plyr` package as above:

```r
arts = plyr::ldply(result$response$docs, function(x) c(headline=x$headline$main, date=x$pub_date))
head(arts)
```

headline	date
Possible Conflict at Heart of Clinton Foundation	2016-05-23T00:00:00Z
Clinton Responds to Benghazi Report	2016-06-28T18:00:20Z
‘Systemic’ Lapses Found in Escape of 2 Killers From Dannemora Prison	2016-06-07T00:00:00Z
18 Newly Implicated in Killers’ Escape. Zero Fired; Zero Prosecuted.	2016-06-24T00:00:00Z
5 Key Factors to the New York Prison Escape, in a Killer’s Words	2016-06-08T00:00:00Z
Bernie Sanders Vows Fight to Convention as Hillary Clinton Wins a Primary	2016-06-06T00:00:00Z