rOpenSci tools for accessing research literature for text mining

Scott Chamberlain*,[a]

[a]*rOpenSci, Museum of Paleontology, University of California, Berkeley, CA, USA*

Abstract

Corresponding Author:

Scott Chamberlain

rOpenSci, Museum of Paleontology, University of California, Berkeley, CA, USA

Email address: myrmecocystus@gmail.com

[*] Corresponding author
Email address: `myrmecocystus(at)gmail.com` (Scott Chamberlain)

April 8, 2019

Background. xxxx.

Methods. xxxx.

Results. xxxx.

Discussion. xxxx.

Introduction

There's more than 100 million articles published (source: Crossref API), representing an enormous amount of knowledge. In addition to simply reading these articles, they contain a vast trove of information of interest to researchers for machine aided questions.

For example, many researchers are interested in statistical outcomes of articles: questions about P-values, about effect sizes, and more. With regard to effect sizes, these are of particular interest, as they are often combined in meta-analyses to draw broad conclusions about a particular question.

Text-mining is the broad term associated with pulling information out of articles. Given the importance of text-mining, good text-mining tools are needed to make it easier for researchers. In particular, the R programming language is used widely throughout many academic fields and thus tools in R for text mining are of particular importance.

Here, we present an overview of text-mining tools in the R programming language. We do not cover analysis tools per se, but rather those tools for searching for, acquiring, and "mashing up" text.

Digital articles: technical aspects

Those articles that are digital can be split into two groups: easily machine readable and non-machine readable.

The machine readable articles are those in XML, JSON, or plain text format. The former two, XML and JSON, are ideal for the machine readable types because they are structured data, whereas plain text has no structure - it's simply a set of characters with line breaks and spaces in between.

Of the non-machine readable kind, there's PDFs. These can be broken out into two groups: text based PDFs and scanned PDFs. The former are converted from digital versions of various kinds (MS Word, OpenOffice, markdown, etc.), while the latter are PDFs created by scanning in print articles for which there is no digital version.

Digital articles: the access landscape

Acces to full-text is the holy grail in text-mining. Some use cases can get by with article metadata (authors, title, etc.), some with abstracts, but many use cases need full-text.

The landscape of access to full-text is extremely hetergeous, with the majority of variation along the publisher axis. The major hurdle is paywalls. The majority of articles are published by the big three publishers - Wiley, Springer, Elsevier - and the majority of their articles are behind paywalls.

A promising sign is an increasing number of open access publishers, yet these represent a very small portion of the total articles (XXXXX) (ref.).

With respect to paywalled articles, access varies by institution, depending on what each institution decided to pay for. In addition, some users may not realize access varies with IP address so that access from campus vs. from home (if not on a VPN) will drastically differ.

We can not end this section without mentioning SciHub. This is a last resort option for many probably, providing access to full text of articles that are normally paywalled. No tools in this manuscript provide access to SciHub.

The discovery problem

xxx

xxx

Data sources

There is increasing open access scientific literature content available online. However, only a small proportion of scientific journals provide access to their full content; whereas, most publishers provide open access to their metadata only (most often through Crossref; Table 1). The following is a synopsis of the major data sources and associated R tools.

Table 1. Sources of scientific literature, their content type provided via web services, whether rOpenSci has an R packages for the service, and where to find the API documentation.

Data Provider	Content Type	rOpenSci Package	Documentation
Crossref	Metadata only	rcrossref/crminer	[1]
DataCite	Metadata only	rdatacite	[2]
Biodiversity Heritage Library	Full content/Metadata	rbhl	[3]
Public Library of Science (PLoS)	Full text/altmetrics	rplos	[4]
Scopus (Elsevier)	Full content/Metadata	fulltext	[5]
arXiv	Full content/Metadata	aRxiv	[6]
Biomed Central (via Springer)	Full content/Metadata	fulltext	[7]
bioRxiv	Full content/Metadata	fulltext	[8]
PMC/Pubmed (via Entrez)	Full content/Metadata	rentrez	[9]
Europe PMC	Full content/Metadata	europepmc	[10]
Microsoft Academic Search	Metadata	fulltext/microdemic	[11]
Directory of Open Access Journals	Metadata	jaod	[12]
JSTOR Data for Research	Full content	jstor	[13]
ORCID	Metadata	rorcid	[14]
Wikimedia's Citoid	Citations	rcitoid	[15]
Open Citation Corpus	Citations	citecorp	[16]

[1] https://api.crossref.org

[2] https://support.datacite.org/docs/api

[3] http://bit.ly/KYQ1Rd

[4] http://api.plos.org/solr

[5] http://bit.ly/J9S616

[6] https://arxiv.org/help/api/index

[7] https://dev.springer.com/

[8] http://www.biorxiv.org/

[9] https://www.ncbi.nlm.nih.gov/books/NBK25500

[10] https://azure.microsoft.com/en-us/services/cognitive-services

[11] https://dev.labs.cognitive.microsoft.com/docs/services/56332331778daf02acc0a50b/operations/565d9001ca73072048922d97

[12] https://doaj.org/api/v1/docs

[13] https://www.jstor.org/dfr/

[14] https://pub.orcid.org/

[15] https://en.wikipedia.org/api/rest_v1/#/Citation/getCitation

[16] http://opencitations.net/

Crossref/Datacite

Crossref is a non-profit that creates (or “mints”) Digital Object Identifiers (DOIs). In addition, they maintain metadata associated with each DOI. The metadata ranges from simple (including author, title, dates, DOI, type, publisher) to including number of citations to the article, as well as references in the article, and even abstracts. At the time of writing they hold 100 million DOIs.

One can search by DOI or search citation data to get citations. In addition, Crossref has a text-mining opt-in program for publishers. The result of this is that some publishers provide URLs for full text content of their articles. The majority of these links are pay-walled, while some are open access. Using any of the various tools for working with Crossref data, you can filter your search to get only articles with full text links, and further to get only articles with full text links that are open access.

The main interfaces for Crossref in R are rcrossref and crminer. Similar interfaces are available in Ruby (serrano) and Python (habanero).

Datacite is similar to Crossref, but focuses on datasets instead of articles. The main interface for Datacite in R is rdatacite.

Biodiversity Heritage Library

The Biodiversity Heritage Library (BHL) houses scans of biodiversity books, and provides web interfaces and APIs to query and fetch those data. They also provide text of the scanned pages. The main R interace to BHL is through rbhl.

Public Library of Science

The Public Library of Science (PLOS) is one of the largest open access only publishers. They as of this writing have published 2.1 million articles. One of the strongs advantages of PLOS is that they provide an API to their Solr instance, which is a very flexible way to search their articles. The main R interace to PLOS is through rplos.

Elsevier/Scopus

Elsevier is one of the largest publishers. Most of their articles are not open access. However, they have a numbrer of advantages if you have access to their articles: they are one of the few publishers to provide machine readable XML (many publishers do have XML versions of articles, but do not provide it); they are one of the few (two) publishers part of Crossref's text and data mining program. The packages fulltext and crminer can be used to access Elsevier articles through Crossref's TDM program. There's an interface to Scopus article search within fulltext.

arXiv/bioRxiv

arXiv and bioRxiv are preprint publishers, the former in existence for many years, and the latter new on the scene. You can access articles from these publishers through fulltext. arXiv does provide a web API that we hook into; bioRxiv does not, but we can get you articles nonetheless.

Pubmed/PMC/Europe PMC

Pubmed/PMC is a corpus/website of NIH funded research in the United States; while Europe PMC is an equivalent for the European Union. You can access articles from Pubmed/PMC through fulltext, and for Europe PMC through europepmc.

Microsoft Academic Research

Microsoft Academic Research (MAR) is a search engine for research articles. You can use their GUI web interface to search, and they provide APIs for programmatic access. The R interface for MAR is microdemic; and fulltext hooks into `microdemic` as well for article search and abstract retrieval.

Directory of Open Access Journals

xxxxx

JSTOR

xxxxx

ORCID

xxxxx

Citoid/Open Citation Corpus

xxx

fulltext: a swiss army knife for text mining in R

fulltext is a general purpose R package for the data part of text-mining: search for articles, get links to articles, get article abstracts, and fetch full text of articles. The `fulltext` package is always adding additional data sources as time allows (See Table 1). Starting from searching for articles, the outputs of search can be fed into a function to get links to those articles, or to get abstracts for those articles, or to fetch their full text.

The following is a breakdown of the major distinct functional parts of `fulltext`.

Search

`ft_search()` provides search access to nine different data sources (PLOS, BMC, Crossref, Entrez, arXiv, bioRxiv, Europe PMC, Scopus, Microsoft Academic), creating a mostly unified interface to all data sources. The parts of each data source that are common are mostly factored into the parameters of the `ft_search()` function, and we also allow the user to pass on data source specific options as needed.

xxxxx

Links

`ft_links()` provides two pathways to get links (URLs) for articles, with a choice of four different data sources (PLOS, BMC, Crossref, Entrez). First, you can use `ft_search()`, then pass the output of that function to `ft_links()`. Second, you can pass DOIs directly to `ft_links()`. Both end up at the same point, links for each article, if they could be found for the user selected data source.

The biggest caveat with `ft_links()` is that we can't gaurantee that the links will work. Link rot is one way in which the links may not work: link rot is when the URL does not point to the original content anymore, or fails altogether. Additionally, with Crossref, publishers can deposit URLs for articles, but they make change the URLs at some later date but not update the URLs with Crossref.

Abstracts

`ft_abstract()` provides access to article abstracts from four different data sources (PLOS, Scopus, Microsoft Academic Research, Crossref). The only way to use the function is to pass article identifiers, which are for the most DOIs.

The advantage of abstracts over full text is that abstracts can often be retrieved even for paywalled articles. That is, you can have much broader coverage of the articles you're targeting relative to full text.

If you are after abstracts, and you are already getting or already have full text, and if the articles are in XML format, then you can use pubchunks to extract out the abstracts.

Fetch full text

`ft_get()` fetchs full text of articles from many different data sources. From the DOIs that are passed in to the function, we detect the publisher, and there are specific plugins for certain publishers:

- aaas
- aip
- amersocclinoncol
- amersocmicrobiol
- arxiv
- biorxiv
- bmc
- copernicus
- crossref
- elife
- elsevier
- entrez
- frontiersin
- ieee
- informa

- instinvestfil
- jama
- microbiology
- peerj
- pensoft
- plos
- pnas
- royalsocchem
- sciencedirect
- scientificsocieties
- wiley

If there's no built-in plugin for the publisher already, we use the FTDOI API (https://ftdoi.org) to try to get the link for the full text of the article. If the FTDOI API doesn't bear fruit, we search Crossref for a link to the full text. If Crossref doesn't have any full text links, we give up.

Since users can go through a lot of article requests, we cache successfully downloaded articles, and keep that knowledge consistent across R sessions; all subsequent requests for the same article just use the cached version. Additionally, all errors in `ft_get()` are collected in a tidy data.frame in the output of the function to help the user quickly determine what went wrong.

How to text mine from R: Three case studies

Case study 1: Citation mining

In this example, xxxx

Load libraries

```r
library("rcrossref")
library("rplos")
library("rorcid")
library("rcitoid")
library("citecorp")
```

rcrossref

Using `rcrossref` for Crossref data:

```r
x <- cr_works(query="NSF")
head(x$data)
#> # A tibble: 6 x 32
#>   alternative.id container.title created deposited published.print doi
#>   <chr>          <chr>           <chr>   <chr>     <chr>           <chr>
#> 1 S106352031630~ Applied and Co~ 2016-0~ 2019-02-~ 2018-03         10.1~
#> 2 <NA>           Biogeosciences~ 2017-0~ 2017-07-~ <NA>            10.5~
#> 3 <NA>           Global Biogeoc~ 2018-0~ 2019-01-~ 2018-10         10.1~
#> 4 <NA>           IEEE Communica~ 2016-1~ 2017-12-~ 2017            10.1~
#> 5 S002178241400~ Journal de Mat~ 2014-0~ 2018-10-~ 2014-10         10.1~
#> 6 123            Light: Science~ 2019-0~ 2019-01-~ 2019-12         10.1~
#> # ... with 26 more variables: indexed <chr>, issn <chr>, issue <chr>,
#> #   issued <chr>, member <chr>, page <chr>, prefix <chr>, publisher <chr>,
#> #   reference.count <chr>, score <chr>, ...
```

Case study 2: Abstract mining

Sometimes you just need abstracts for your research question. The benefit of only needing abstracts, and not need full text, is that there's many more articles that will have abstracts available than have their full text available.

As an example, let's say you xxxx

```r
library("fulltext")
```

xxxxx

Using `fulltext`:

```r
res <- ft_search("ecology", from = "crossref",
  crossrefopts = list(filter = c(has_abstract = TRUE)))
ids <- res$crossref$data$doi
out <- ft_abstract(x = ids, from = "crossref")
abstracts <- vapply(out$crossref, "[[", "", "abstract")
```

Using quanteda, read the abstracts into a corpus

```r
library("quanteda")
corp <- corpus(abstracts)
docvars(corp) <- ids
```

Get a summary of the abstracts

```r
summary(corp)
#> Corpus consisting of 10 documents:
#>
#>    Text Types Tokens Sentences                    V1
#>   text1   143    262        10   10.2458/v22i1.21112
#>   text2   117    244         6   10.2458/v17i1.21696
#>   text3    75    118         4   10.2458/v25i1.23119
#>   text4     5      8         1    10.2458/v1i1.21154
#>   text5   105    171         7   10.1155/2011/868426
#>   text6   112    181         6   10.1155/2012/273413
#>   text7   117    240         8 10.5194/we-13-91-2013
#>   text8   140    245         9 10.5194/we-13-95-2013
#>   text9   107    202         7   10.1155/2014/198707
#>  text10   118    224         6   10.5402/2011/897578
```

```
#>
#> Source: /Users/sckott/github/ropensci/textmine/use-cases/* on x86_64 by sckott
#> Created: Fri Apr  5 11:36:04 2019
#> Notes:
```

Use the `kwic()` function to see a word in context across the abstracts

```
kwic(corp, pattern = "ecology")
#>
#>   [text1, 33] knowledge production within critical political | ecology |
#>   [text1, 50]              in scientific articles on dryland | ecology |
#>  [text1, 204]                 to equilibrium models in range | ecology |
#>  [text1, 246]    communal areas.Keywords: Critical political | ecology |
#>  [text1, 255]                 , scientific models, rangeland | ecology |
#>    [text2, 5]                            < jats:p> Political | ecology |
#>   [text2, 23]        manifestations of political economy and | ecology |
#>   [text2, 45]                      I try to extend political | ecology |
#>  [text2, 149]                   , in dialogue with political | ecology |
#>  [text2, 177]            people and resources that political | ecology |
#>  [text2, 229]      indigeneity scholars.Key words: political | ecology |
#>   [text3, 71]                   an analysis from a political | ecology |
#>  [text3, 114]                system, supermarkets, political | ecology |
#>  [text6, 134]                was observed when allopatry and | ecology |
#>  [text7, 167]             ecosystem should be considered for | ecology |
#>  [text7, 185]                       the" four-color issue of | ecology |
#>  [text7, 201]             step toward advancing knowledge in | ecology |
#>  [text9, 195]         or for theoretical studies integrating | ecology |
#>
#>  . This article is a
#>  , and investigates the functions
#>  , and the fence-line photographs
#>  , fence-line photography, scientific
#>  , Southern Africa</
#>  has expanded in multiple new
#>  in the" problem"
#>  to engage with ethnic studies
#>  approaches to better understand the
#>  focuses on cannot be adequately
#>  , coloniality, Maidu,
#>  standpoint allows a different interpretation
#>  </ jats:p>
#>  act together, leading to
#>  "? Here, I
#>  ", and propose that
#>  and conservation biology. In
#>  and biogeography.</
```

Case study 3: Full text mining

In this example, xxxx

```
library("fulltext")
# library("crminer")
```

Search for articles

Search for the term *ecology* in PLOS journals.

```
(res1 <- ft_search(query = 'ecology', from = 'plos'))
#> Query:
#>   [ecology]
#> Found:
#>   [PLoS: 47337; BMC: 0; Crossref: 0; Entrez: 0; arxiv: 0; biorxiv: 0; Europe PMC: 0; Scopus:
#> Returned:
#>   [PLoS: 10; BMC: 0; Crossref: 0; Entrez: 0; arxiv: 0; biorxiv: 0; Europe PMC: 0; Scopus: 0;
```

Each publisher/search-engine has a slot with metadata and data

```
res1$plos
#> Query: [ecology]
#> Records found, returned: [47337, 10]
#> License: [CC-BY]
#>                               id
#> 1  10.1371/journal.pone.0001248
#> 2  10.1371/journal.pone.0059813
#> 3  10.1371/journal.pone.0155019
#> 4  10.1371/journal.pone.0080763
#> 5  10.1371/journal.pone.0208370
#> 6  10.1371/journal.pone.0150648
#> 7  10.1371/journal.pcbi.1003594
#> 8  10.1371/journal.pone.0102437
#> 9  10.1371/journal.pone.0175014
#> 10 10.1371/journal.pone.0166559
```

Get full text

Using the results from `ft_search()` we can grab full text of some articles

```
(out <- ft_get(res1))
#> <fulltext text>
#> [Docs] 10
#> [Source] ext - /Users/sckott/Library/Caches/R/fulltext
#> [IDs] 10.1371/journal.pone.0001248 10.1371/journal.pone.0059813
#>      10.1371/journal.pone.0155019 10.1371/journal.pone.0080763
#>      10.1371/journal.pone.0208370 10.1371/journal.pone.0150648
#>      10.1371/journal.pcbi.1003594 10.1371/journal.pone.0102437
#>      10.1371/journal.pone.0175014 10.1371/journal.pone.0166559 ...
```

Extract text from pdfs

Ideally for text mining you have access to XML or other text based formats. However, sometimes you only have access to PDFs. In this case you want to extract text from PDFs. `fulltext` can help with that.

You can extract from any pdf from a file path, like:

```
path <- system.file("examples", "example1.pdf", package = "fulltext")
ft_extract(path)
#> <document>/Library/Frameworks/R.framework/Versions/3.5/Resources/library/fulltext/examples/ex
#>   Title: Suffering and mental health among older people living in nursing homes---a mixed-met
#>   Producer: pdfTeX-1.40.10
#>   Creation date: 2015-07-17
```

Extract text chunks

Requires the pubchunks library. Here, we'll search for some PLOS articles, then get their full text, then extract various parts of each article with `pub_chunks()`.

```
library("pubchunks")
res <- ft_search(query = "ecology", from = "plos", limit = 3)
x <- ft_get(res)
x %>% ft_collect() %>% pub_chunks(c("doi", "history")) %>% pub_tabularize()
#> $plos
#> $plos$`10.1371/journal.pone.0001248`
#>                            doi history.received history.accepted
#> 1 10.1371/journal.pone.0001248       2007-07-02       2007-11-06
#>   .publisher
#> 1       plos
#>
#> $plos$`10.1371/journal.pone.0059813`
#>                            doi history.received history.accepted
#> 1 10.1371/journal.pone.0059813       2012-09-16       2013-02-19
#>   .publisher
#> 1       plos
```

```
#>
#> $plos$`10.1371/journal.pone.0155019`
#>                               doi history.received history.accepted
#> 1 10.1371/journal.pone.0155019       2015-09-22       2016-04-22
#>   .publisher
#> 1       plos
```

Future directions

xxxx

Acknowledgments

xxxx

Data Accessibility

All scripts and data used in this paper can be found in the permanent data archive Zenodo under the digital object identifier (DOI). This DOI corresponds to a snapshot of the GitHub repository at https://github.com/ropensci/textmine. Software can be found at https://github.com/ropensci/xxx, xxxx, all under MIT licenses.

References