

# R tools for accessing research literature for text mining

Scott Chamberlain<sup>\*,a</sup>

<sup>a</sup>*rOpenSci, Museum of Paleontology, University of California, Berkeley, CA, USA*

## Abstract

Text mining is a powerful method for answering research questions. However, getting texts to extract information can be a daunting and complicated task. The primary reason for this is the diversity of publisher technologies. There are thousands of different publishers, each with their own licenses, URL patterns, access options, and more. Layered on top of that is the varied access each user has based on their institutional affiliation. Here, I introduce a suite of software packages in the R programming language for fetching texts. The tapestry of different publishers, access levels, and other factors requires a patchwork of approaches for getting texts to users. The flagship R package called `fulltext` attempts to simplify search and retrieval of texts for text mining by serving as an interface to the varied and complex publishers. The `fulltext` package, along with many others, make acquiring texts easier than ever, facilitating answering research questions with text mining.

---

\*Corresponding author

Email address: `myrmecocystus(at)gmail.com` (Scott Chamberlain)

## 5 Introduction

There are more than 100 million research articles published (Crossref API: <https://github.com/CrossRef/rest-api-doc>), representing an enormous amount of knowledge. In addition to simply reading these articles, the articles contain a vast trove of information of interest to researchers for machine aided questions (Kong & Gerstein, 2018; Usai et al., 2018). For example, many researchers are interested in statistical outcomes of articles that can be extracted from numeric results: P-values, effect sizes, means, and more. In addition, researchers are often interested in words in articles, their use through time, and the contexts they are found in.

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 to do. Graphical user interface (GUI) based text mining tools are available (e.g., Ba & Bossy, 2016; Cañada et al., 2017; Muñoz, Kissling & Loon, 2019) and some research papers have used them (Chaix et al., 2019), but given the urgent recent call to action for more reproducible research (Open Science Collaboration, 2015; Camerer et al., 2016, 2018), we must move away from GUI based tools as fast as possible. A number of examples of programmatic tools can be found in the literature. For example, Sinclair et al. (2016) present a tool in Python called seqenv for the domain specific task of linking sequences to environments through text mining.

Using programmatic tools for text mining is important for a variety of reasons. First, using programmatic tools makes it easier for yourself. You can re-run code to get updated results as data changes, and results can always be re-created. Second, code can be shared whereas GUI driving workflows can not; colleagues, reviewers and other researchers can reproduce and check your work. (FIXME: re-work this; citations)

Most recent text mining papers do not use programmatic approaches, highlighting the need for more programmatic text mining tools, and increased discussion of those tools to increase awareness. For example, many papers search Web of Science using their web interface, and downloading papers manually (Ding, Li & Fan, 2018; McCallen et al., 2019). Many of these papers doing GUI based searching and paper downloads are using R or Python downstream for analysis; replacing GUI based data acquisition with programmatic approaches will improve research.

The R programming language is free of cost, and is used widely throughout many academic fields; tools in R for text mining are of particular importance because they can be adopted by academics rapidly.

35 Here, I present an overview of text mining tools in the R programming language, not for text mining  
36 analysis, but rather those tools for searching for, acquiring, and extracting parts of texts (e.g., title,  
37 abstract, authors). Most of the packages presented here are part of the rOpenSci suite ([https://ropensci.  
38 org/](https://ropensci.org/)).

## 39 **Digital articles: technical aspects**

40 Those articles that are digital (which in theory includes all articles) can be split into two groups:  
41 machine readable and non-machine readable.

42 The machine readable articles are those in XML<sup>1</sup>, JSON<sup>2</sup>, or plain text format. The former two, XML  
43 and JSON, are the best machine readable types because they are structured data<sup>3</sup>, whereas plain text  
44 has no structure - it's simply a set of characters with line breaks and spaces in between.

45 Of the non-machine readable types, the most notable is the Portable Document Format (PDF)<sup>4</sup>. These  
46 can be broken out into two groups: text based PDFs and scanned PDFs (images of text). The former  
47 are converted from digital versions of various kinds (MS Word, OpenOffice, LaTeX, markdown, etc.),  
48 while the latter are created by scanning print articles to a PDF format. Text-based PDFs are much  
49 better for text mining purposes as plain text can be extracted easily in R with [pdftools](#), a binding to  
50 [libpoppler](#). However, with scanned PDFs, text must be extracted using Optimal Character Recognition  
51 (OCR; see R package [tesseract](#)), which isn't always a clean solution, especially compared to true text  
52 based PDFs.

53 The reality in scholarly publishing is all publishers, if they provide any access to their articles, only  
54 provide PDF format. Very few publishers, with some quite large (Elsevier, Pensoft, PLOS), provide  
55 XML format. Although most publishers most likely have the XML behind each of their articles, they for  
56 some indefensible reason do not share it - making text mining more difficult. Some provide plain text  
57 (Elsevier). I only know of one publisher that provides full text as JSON (PLOS). Thus, text mining, in  
58 most cases, will require extracting text from PDFs.

---

<sup>1</sup><https://www.w3.org/TR/xml/>

<sup>2</sup><https://tools.ietf.org/html/rfc7159>

<sup>3</sup>[https://en.wikipedia.org/wiki/Data\\_model](https://en.wikipedia.org/wiki/Data_model)

<sup>4</sup><https://en.wikipedia.org/wiki/PDF>

## 59 **Digital articles: the access landscape**

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

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

65 A promising sign is an increasing number of open access articles, yet open access articles represent a  
66 small percent of all articles: an estimate in 2018 said that 28% of the scholarly literature was open  
67 access (Piwowar et al., 2018).

68 With respect to paywalled articles, access varies by institution, depending on each institution's publisher  
69 contracts. MORE ABOUT THIS ...

70 Some may not realize access to articles varies with IP address so that access from campus vs. from  
71 home (if not on a VPN) will drastically differ. Sometimes a VPN is required, and this can provide a  
72 significant technical hurdle to users attempting to do text mining work.

73 One final hurdle in text mining comes unsurprisingly from Elsevier. They use so-called "fences" for  
74 programmatic access. That is, even if a person trying to get an article programmatically their institution  
75 has access to and they have access to, and they are on the correct IP address, they may still not get  
76 access to an Elsevier article. Elsevier puts in place these fences and only if you contact their technical  
77 team directly can you get these fences removed, and only then on a per institution basis.

78 I can not end this section without mentioning SciHub. This is a last resort option for many probably  
79 (or possibly first, depending on your level of access), providing access to full text of articles that are  
80 normally paywalled. No tools in this manuscript provide access to SciHub.

## 81 **The discovery problem**

82 A text mining project starts with a question. From that question, researchers then attempt to acquire  
83 scholarly articles for text mining. Finding appropriate articles is not altogether straight-forward.

84 Some of the discovery difficulty relates to the fact that there are so many places to search for articles;  
85 a non-exhaustive list: Google Scholar, Microsoft Academic Research, Scopus, ScienceDirect, Web of

86 Science, Pubmed/Entrez, Europe PMC, Directory of Open Access Journals, Open Knowledge Maps,  
87 CORE, Fatcat, and more. It's probably difficult to know where the best place is to search. Some of  
88 these are paywalled (e.g., Web of Science), and some are not.

89 The most important aspect about any source for article search with respect to reproducible research is  
90 being able to use the data source programmatically. Of those listed above, the following can be used  
91 programmatically: Microsoft Academic Research, Scopus, ScienceDirect, Pubmed/Entrez, Europe PMC,  
92 and Directory of Open Access Journals. All of these are included in the R package [fulltext](#), discussed  
93 further below.

94 On top of the vast array of different data sources is the varied ways that search is implemented in each  
95 source. Most sources are probably using Solr or Elasticsearch under the hood, though we can't know  
96 this for sure as most do not make their software infrastructure public knowledge. Nonetheless, data  
97 sources differ in how search works from the user perspective. For example, some provide wild card/fuzzy  
98 search (i.e., 'appl\*' includes results for 'apple', 'application', etc.) and some do not. Some sources  
99 are searching full text of articles, while others only search metadata (i.e., title, authors, abstract). In  
100 addition, each source has a different set of metadata/full text available. In brief, the same search against  
101 different sources produces different results. Some text mining research articles perform the same search  
102 against many different sources (refs), while others choose just one source.

## 103 **Data sources**

104 There is increasing open access scientific literature content available online. However, only a small  
105 proportion of scientific journals provide access to their full text; whereas, most publishers provide open  
106 access to their metadata only (most often through Crossref; Table 1). The following is a synopsis of the  
107 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	rcrossref/crminer	<a href="#">5</a>
DataCite	Metadata	rdatacite	<a href="#">6</a>
Biodiversity Heritage Library	Full text/Metadata	rbhl	<a href="#">7</a>
Public Library of Science (PLOS)	Full text (pdf/xml)/Metadata	rplos	<a href="#">8</a>
Scopus (Elsevier)	Full text (pdf/xml)/Metadata	fulltext	<a href="#">9</a>
arXiv	Full text (pdf)/Metadata	aRxiv	<a href="#">10</a>
Biomed Central (via Springer)	Full text (pdf)/Metadata	fulltext	<a href="#">11</a>
bioRxiv	Full text (pdf)/Metadata	fulltext	<a href="#">12</a>
PMC/Pubmed (via Entrez)	Full text (pdf/xml)/Metadata	rentrez	<a href="#">13</a>
Europe PMC	Full text (pdf/xml)/Metadata	europemc	<a href="#">14</a>
Microsoft Academic Search	Metadata	microdemic	<a href="#">15</a>
Directory of Open Access Journals	Metadata	jaod	<a href="#">16</a>
JSTOR Data for Research	Full text	jstor	<a href="#">17</a>
ORCID	Metadata	rorcid	<a href="#">18</a>
Wikimedia's Citoid	Citations	rcitoid	<a href="#">19</a>
Open Citation Corpus	Citations	citecorp	<a href="#">20</a>

<sup>5</sup><https://api.crossref.org>

<sup>6</sup><https://support.datacite.org/docs/api>

<sup>7</sup><http://bit.ly/KYQ1Rd>

<sup>8</sup><http://api.plos.org/solr>

<sup>9</sup><http://bit.ly/J9S616>

<sup>10</sup><https://arxiv.org/help/api/index>

<sup>11</sup><https://dev.springer.com/>

<sup>12</sup><http://www.biorxiv.org/>

<sup>13</sup><https://www.ncbi.nlm.nih.gov/books/NBK25500>

<sup>14</sup><https://azure.microsoft.com/en-us/services/cognitive-services>

<sup>15</sup><https://dev.labs.cognitive.microsoft.com/docs/services/56332331778daf02acc0a50b/operations/>

<sup>16</sup><https://doi.org/10.2196/565d9001ca73072048922d97>

<sup>17</sup><https://doaj.org/api/v1/docs>

<sup>18</sup><https://www.jstor.org/df/>

<sup>19</sup><https://pub.orcid.org/>

<sup>20</sup>[https://en.wikipedia.org/api/rest\\_v1/#/Citation/getCitation](https://en.wikipedia.org/api/rest_v1/#/Citation/getCitation)

<sup>21</sup><http://opencitations.net/>

Data Provider	Content Type	rOpenSci Package	Documentation
Fatcat	Metadata	none	<a href="#">21</a>
SHERPA/RoMEO	Journal Level Metadata	rromeo	<a href="#">22</a>
CORE	Full text (pdf)/Metadata	rcoreoa	<a href="#">23</a>
Dissemin	Metadata	dissemr	<a href="#">24</a>

## Crossref and 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](#). [rcrossref](#) is a complete client for the public facing Crossref web services including metadata, whereas [crminer](#) focuses only on retrieving full text of articles. Similar interfaces to [rcrossref](#) 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](#).

<sup>21</sup><https://fatcat.wiki/>

<sup>22</sup><https://www.sherpa.ac.uk/romeo/apimanual.php?la=en&flDnum=%7C&mode=simple>

<sup>23</sup><https://core.ac.uk/>

<sup>24</sup><https://dissemin.readthedocs.io/en/latest/api.html>

## 129 *Public Library of Science*

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

## 134 *Elsevier/Scopus*

135 Elsevier is one of the largest publishers. Scopus is one of their products, a searchable database of  
136 articles. Most of their articles are not open access. However, they have a number of advantages if you  
137 have access to their articles: they are one of the few publishers to provide machine readable XML (many  
138 publishers do have XML versions of articles, but do not provide it, as mentioned in the Introduction);  
139 they are one of the few (two) publishers that take part in Crossref's text and data mining program. The  
140 packages [fulltext](#) and [crminer](#) can be used to access Elsevier articles through Crossref's TDM program.  
141 There's an interface to Scopus article search within [fulltext](#).

## 142 *arXiv/bioRxiv*

143 arXiv and bioRxiv are preprint publishers, the former in existence for many years, and the latter new  
144 on the scene. Preprints are scholarly articles that are generally not peer-reviewed, but that for the most  
145 part will later be published in a different peer-review publication. You can access articles from these  
146 publishers through [fulltext](#). arXiv does provide a web API that we hook into; bioRxiv does not, but we  
147 can get you articles nonetheless.

## 148 *Pubmed/PMC/Europe PMC*

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

## 152 *Microsoft Academic Research*

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



## 156 *Directory of Open Access Journals*

157 Directory of Open Access Journals (DOAJ) maintains data on open access journals, as well as some  
158 portion of the articles in those journals. Thus, you can search for journals as well as articles with DOAJ.  
159 The R interface for DOAJ is [jaod](#).

## 160 *JSTOR*

161 JSTOR's Data for Research program gives institutions with access to JSTOR, access to full text of  
162 articles within JSTOR. There is no way however to make the interaction with JSTOR completely  
163 programmatic, thus making reproducible research very difficult. Nonetheless, there is an R package  
164 ([jstor](#)) for using data from JSTOR's Data for Research.

## 165 *ORCID*

166 ORCID (<https://orcid.org/>) is an organization keeping track of identifiers and metadata for researchers  
167 around the world. Individuals can optionally maintain metadata on their scholarly works connected to  
168 their account with ORCID. Thus, across all of ORCID, a significant cache of metadata is accruing on  
169 scholarly works, their funding amounts, collaborators, etc., useful for bibliometrics research and more.  
170 The R interface for ORCID is [rorcid](#).

## 171 *Citoid/Open Citation Corpus*

172 The Open Citation Corpus (<http://opencitations.net/>) holds records of which articles cite which other  
173 articles, allowing for all important research on the scholarly web of citation. Citation data has been  
174 very closely guarded until recently, but the largest publishers are still not contributing to the Open  
175 Citation Corpus. The R interface to the Open Citation Corpus is [rcitoid](#).

## 176 *Fatcat*

177 Fatcat is a project from Ben Newbold of the Internet Archive Labs. It is a “versioned, publicly-editable  
178 catalog of research publications: journal articles, conference proceedings, pre-prints, blog posts”. Fatcat  
179 currently does not have an R client, but is used inside of the [fulltext](#) package.

## 180 *SHERPA/RoMEO*

181 SHERPA/RoMEO (<http://sherpa.mimas.ac.uk/romeo/index.php>) aggregates and analyses publisher  
182 open access policies and provides summaries of self-archiving permissions and conditions of rights given  
183 to authors. The `[rromeo]` is an R interface to SHERPA/RoMEO.

## 184 *CORE*

185 CORE (<https://core.ac.uk/>) touts itself as the world's largest collection of open access research articles,  
186 providing metadata on journals and articles, as well as access to the full text of articles. The `rcoreoa` R  
187 package interfaces with the CORE API.

## 188 *Dissemin*

189 Dissemin (<https://dissem.in/>) detects papers behind pay-walls and invites authors to upload them  
190 to an open repository. Dissemin provides metadata including links to open versions of articles. The  
191 `[dissemr]` R package interfaces with the Dissemin API.

## 192 *Learn more*

193 To learn how to use the above packages, follow the links to get to documentation, and how to report  
194 issues. The rOpenSci forum (<https://discuss.ropensci.org/>) is a good place to ask questions about these  
195 packages.

## 196 **fulltext: a toolset for text mining in R**

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

202 *Search*

203 `ft_search()` provides search access to nine different data sources (PLOS, BMC, Crossref, Entrez, arXiv,  
204 bioRxiv, Europe PMC, Scopus, Microsoft Academic), creating a mostly unified interface to all data  
205 sources. The parts of each data source that are common are for the most part factored out into the  
206 parameters of the `ft_search()` function: query term(s), pagination (number of results, result number  
207 to start at). In addition, we allow the user to pass on data source specific options to refine the search  
208 per data source.

209 With `ft_search()`, you can query any combination of the nine data sources at once. The returned  
210 object is a list, with access to results of each data source by its name (e.g., `$plos`, or `$crossref`). For  
211 each data source, the returned object does vary because the returned data from each data source widely  
212 varies; for the most part `data.frame`'s are returned. For those data sources not queried, their slot is  
213 empty.

214 One important aspect of the research result we highlight is the licenses in the returned data for each  
215 data source.

```
x <- ft_search(query = 'ecology', from = c("plos", "crossref"))
```

216 The results for this PLOS search have all CC-BY licenses

```
x$plos
#> Query: [ecology]
#> Records found, returned: [47257, 10]
#> License: [CC-BY]
#>
#>          id
#> 1 10.1371/journal.pone.0001248
#> 2 10.1371/journal.pone.0059813
#> 3 10.1371/journal.pone.0080763
#> 4 10.1371/journal.pone.0155019
#> 5 10.1371/journal.pone.0175014
#> 6 10.1371/journal.pone.0150648
#> 7 10.1371/journal.pone.0208370
#> 8 10.1371/journal.pcbi.1003594
```

```
#> 9 10.1371/journal.pone.0102437
#> 10 10.1371/journal.pone.0166559
```

217 Whereas the results for this Crossref search have mixed licenses

```
x$crossref
#> Query: [ecology]
#> Records found, returned: [164657, 10]
#> License: [variable, see individual records]
#>   archive          container.title    created deposited
#> 1 Portico          Ecology 2006-05-03 2018-08-04
#> 2 Portico          Ecology 2006-05-03 2018-08-04
#> 3      NA          Ecology 2006-05-03 2018-08-04
#> 4      NA          Ecology 2006-05-03 2018-08-04
#> 5      NA          Ecology 2006-05-03 2018-08-04
#> 6      NA          Ecology 2006-05-03 2018-08-04
#> 7      NA          Ecology 2006-05-09 2018-08-01
#> 8 Portico          Ecology 2017-04-26 2019-03-08
#> 9      NA Trends in Ecology & Evolution 2002-07-25 2017-06-14
#> 10     NA Journal of Industrial Ecology 2014-11-21 2017-06-23
#> Variables not shown: published.print (chr), published.online (chr), doi
#>   (chr), indexed (chr), issn (chr), issue (chr), issued (chr), member
#>   (chr), page (chr), prefix (chr), publisher (chr), reference.count
#>   (chr), score (chr), source (chr), title (chr), type (chr), url (chr),
#>   volume (chr), author (list), link (list), license (list), subject
#>   (chr), alternative.id (chr), subtitle (chr), reference (list)
```

218 You can dig into the license field for each article, with URLs holding information on each license

```
vapply(x$crossref$data$license, function(w) w$URL[1], "")
#> [1] "http://doi.wiley.com/10.1002/tdm_license_1.1"
#> [2] "http://doi.wiley.com/10.1002/tdm_license_1.1"
#> [3] "http://doi.wiley.com/10.1002/tdm_license_1"
```

```
#> [4] "http://doi.wiley.com/10.1002/tdm_license_1.1"
#> [5] "http://doi.wiley.com/10.1002/tdm_license_1"
#> [6] "http://doi.wiley.com/10.1002/tdm_license_1"
#> [7] "http://doi.wiley.com/10.1002/tdm_license_1"
#> [8] "http://doi.wiley.com/10.1002/tdm_license_1.1"
#> [9] "http://www.elsevier.com/tdm/userlicense/1.0/"
#> [10] "http://doi.wiley.com/10.1002/tdm_license_1.1"
```

## 219 *Links*

220 `ft_links()` provides two pathways to get links (URLs) for articles, with a choice of four different data  
 221 sources (PLOS, BMC, Crossref, Entrez). First, you can use `ft_search()`, then pass the output of that  
 222 function to `ft_links()`.

```
out <- ft_search(query = "ecology", from = "entrez")
ft_links(out)
#> <fulltext links>
#> [Found] 6
#> [IDs] ID_30964001 ID_30962485 ID_30962432 ID_30952928 ID_30674747
#> ID_30674743 ...
```

223 Second, you can pass DOIs directly to `ft_links()`. Both end up at the same point, links for each  
 224 article, if they could be found for the user selected data source.

```
ft_links(out$entrez$data$doi)
```

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

## 229 *Abstracts*

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

232 which are for the most DOIs.

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

236 If you are after abstracts, and you are already getting or already have full text, and if the articles are in  
237 XML format, then you can use [pubchunks](#) to extract out the abstracts.

### 238 *Fetch full text*

239 `ft_get()` fetches full text of articles from many different data sources. From the DOIs that are passed  
240 in to the function, we detect the publisher, and there are specific plugins for certain publishers: AAAS,  
241 American Institute of Physics, American Society of Clinical Oncology, American Society for Microbiology,  
242 arXiv, bioRxiv, BiomedCentral, Copernicus, Crossref, Elife, Elsevier, Pubmed/PMC via NCBI’s Entrez,  
243 Frontiers, IEEE, Informa, Instituto de Investigaciones Filologicas, American Medical Association,  
244 Microbiology Society, PeerJ, Pensoft, PLOS, PNAS, Royal Society of Chemistry, ScienceDirect, Scientific  
245 Societies, and Wiley.

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

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

## 253 **Text mining in R: Use Cases**

### 254 *Use cases in the literature*

255 At least three papers have used the fulltext package. Bauer, Barbera & Munzert (2016) explored a  
256 way to think about the qualitative impact of citations rather than simple counts of citations; they did  
257 not use fulltext, but pointed it out as a good option for collecting papers. Walsh & Cherbuin (2018)  
258 used fulltext and [roadoi](#), an rOpenSci R package, to “map the literature” on nutritional interventions in

259 cognitive health. Piper et al. (2019) in a review of DNA metabarcoding for high-throughput insect  
260 surveillance used fulltext, along with [rentrez](#) (an rOpenSci package), to collect articles for text mining.

### 261 *Three case studies*

262 The following three case studies demonstrate in brief how one could use fulltext and related packages to  
263 do a few different text mining tasks.

### 264 *Case study 1: Citation mining*

265 In this example, xxxx

### 266 *Load libraries*

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

### 267 *rcrossref*

268 Using `rcrossref` for Crossref data:

```
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>, ...
```

## 269 Case study 2: Abstract mining

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

273 As an example, let's say you xxxx

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

274 *xxxxx*

275 Using `fulltext`:

```
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")
```

276 Using `quanteda`, read the abstracts into a corpus

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

277 Get a summary of the abstracts

```
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: Thu Apr 11 11:20:19 2019
#> Notes:

```

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

```

kwic(corp, pattern = "ecology", window = 2)
#>
#> [text1, 33] critical political | ecology | . This
#> [text1, 50] on dryland | ecology | , and
#> [text1, 204] in range | ecology | , and
#> [text1, 246] Critical political | ecology | , fence-line
#> [text1, 255] , rangeland | ecology | , Southern
#> [text2, 5] > Political | ecology | has expanded
#> [text2, 23] economy and | ecology | in the
#> [text2, 45] extend political | ecology | to engage
#> [text2, 149] with political | ecology | approaches to
#> [text2, 177] that political | ecology | focuses on
#> [text2, 229] : political | ecology | , coloniality
#> [text3, 71] a political | ecology | standpoint allows
#> [text3, 114] , political | ecology | </

```

```
#> [text6, 134]      allopatry and / ecology / act together
#> [text7, 167]      considered for / ecology / "?"
#> [text7, 185]      issue of / ecology / ",
#> [text7, 201]      knowledge in / ecology / and conservation
#> [text9, 195] studies integrating / ecology / and biogeography
```

279 *Case study 3: Full text mining*

280 In this example, xxxx

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

281 *Search for articles*

282 Search for the term *ecology* in PLOS journals.

```
(res1 <- ft_search(query = 'ecology', from = 'plos'))
#> Query:
#> [ecology]
#> Found:
#> [PLoS: 47272; BMC: 0; Crossref: 0; Entrez: 0; arxiv: 0; biorxiv: 0; Europe PMC: 0; Scopus: 0]
#> Returned:
#> [PLoS: 10; BMC: 0; Crossref: 0; Entrez: 0; arxiv: 0; biorxiv: 0; Europe PMC: 0; Scopus: 0; .]
```

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

```
res1$plos
#> Query: [ecology]
#> Records found, returned: [47272, 10]
#> License: [CC-BY]
#>
#> id
#> 1 10.1371/journal.pone.0001248
#> 2 10.1371/journal.pone.0059813
```

```
#> 3 10.1371/journal.pone.0080763
#> 4 10.1371/journal.pone.0155019
#> 5 10.1371/journal.pone.0175014
#> 6 10.1371/journal.pone.0150648
#> 7 10.1371/journal.pone.0208370
#> 8 10.1371/journal.pcbi.1003594
#> 9 10.1371/journal.pone.0102437
#> 10 10.1371/journal.pone.0166559
```

284 *Get full text*

285 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.0080763 10.1371/journal.pone.0155019
#>      10.1371/journal.pone.0175014 10.1371/journal.pone.0150648
#>      10.1371/journal.pone.0208370 10.1371/journal.pcbi.1003594
#>      10.1371/journal.pone.0102437 10.1371/journal.pone.0166559 ...
```

286 *Extract text from pdfs*

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

290 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
```

## 291 *Extract text chunks*

292 Requires the [pubchunks](#) library. Here, we'll search for some PLOS articles, then get their full text, then  
 293 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.0080763`
#>
#> doi history.received history.accepted
#> 1 10.1371/journal.pone.0080763      2013-08-15      2013-10-16
#> .publisher
#> 1      plos
```

## 294 *Further processing*

295 From here, you'll likely want to use R packages for removing stopwords, tokenizing, and more.

296 The Natural Language Processing CRAN Task View ([https://cloud.r-project.org/web/views/](https://cloud.r-project.org/web/views/NaturalLanguageProcessing.html)  
297 [NaturalLanguageProcessing.html](https://cloud.r-project.org/web/views/NaturalLanguageProcessing.html)) is a good place to start if you're not sure what packages to use.  
298 Some packages worth trying in this space are [antiword](#), [pdftools](#), [rtika](#), [tokenizers](#), and [textreuse](#).

## 299 **Future directions**

300 Text mining will always be a complex task given all the layers involved: often temporal time-span of  
301 research questions; varied permissions among researchers and their articles they're trying to access;  
302 varied approaches to getting full text (xml vs pdf vs plain text); and more.

303 Programmatic text mining is a first step towards making text mining easier. The R ecosystem is an  
304 especially good place to do text mining because there are many packages for text mining analysis, and  
305 endless packages for any required statistical analyses. In addition, rOpenSci and others are building  
306 up a set of packages in R for searching for and acquiring full text programatically to help make the  
307 research workflow as reproducible as possible.

308 Future work for `fulltext` includes:

- 309 1. Adding more publisher plugins
- 310 2. Fine tuned user control over publishers
- 311 3. Improve VPN/proxy controls
- 312 4. Incorporate more search engines to help resolve URLs for fulltext versions
- 313 5. Improve documentation

314 With respect to what publishers can do to make text mining easier, publishers should:

- 315 1. provide XML if they have it
- 316 2. not change URL patterns so often, or at all
- 317 3. maintain consistent URL patterns among journals, years, etc.
- 318 4. keep their Crossref metadata up to date
- 319 5. open up their citation data

## 320 **Acknowledgments**

321 This manuscript was greatly improved by comments from Maëlle Salmon.

## 322 Data Accessibility

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

## 327 References

- 328 Ba M., Bossy R. 2016. Interoperability of corpus processing work-flow engines: The case of alvisnlp/ml  
329 in openminted. In: *Proceedings of the workshop on cross-platform text mining and natural language*  
330 *processing interoperability (interop 2016) at Irec*. 15–18.
- 331 Bauer PC., Barbera P., Munzert S. 2016. The quality of citations: Towards quantifying qualitative  
332 impact in social science research. *SSRN Electronic Journal*.
- 333 Camerer CF., Dreber A., Forsell E., Ho T-H., Huber J., Johannesson M., Kirchler M., Almenberg  
334 J., Altmejd A., Chan T., Heikensten E., Holzmeister F., Imai T., Isaksson S., Nave G., Pfeiffer T.,  
335 Razen M., Wu H. 2016. Evaluating replicability of laboratory experiments in economics. *Science*  
336 351:1433–1436.
- 337 Camerer CF., Dreber A., Holzmeister F., Ho T-H., Huber J., Johannesson M., Kirchler M., Nave G.,  
338 Nosek BA., Pfeiffer T., Altmejd A., Buttrick N., Chan T., Chen Y., Forsell E., Gampa A., Heikensten  
339 E., Hummer L., Imai T., Isaksson S., Manfredi D., Rose J., Wagenmakers E-J., Wu H. 2018. Evaluating  
340 the replicability of social science experiments in nature and science between 2010 and 2015. *Nature*  
341 *Human Behaviour* 2:637–644.
- 342 Cañada A., Capella-Gutierrez S., Rabal O., Oyarzabal J., Valencia A., Krallinger M. 2017. LimTox: A  
343 web tool for applied text mining of adverse event and toxicity associations of compounds, drugs and  
344 genes. *Nucleic Acids Research* 45:W484–W489.
- 345 Chaix E., Deléger L., Bossy R., Nédellec C. 2019. Text mining tools for extracting information about  
346 microbial biodiversity in food. *Food Microbiology* 81:63–75.
- 347 Ding Z., Li Z., Fan C. 2018. Building energy savings: Analysis of research trends based on text mining.  
348 *Automation in Construction* 96:398–410.

349 Kong X., Gerstein MB. 2018. Text mining systems biology: Turning the microscope back on the  
 350 observer. *Current Opinion in Systems Biology* 11:117–122.

351 McCallen E., Knott J., Nunez-Mir G., Taylor B., Jo I., Fei S. 2019. Trends in ecology: Shifts in ecological  
 352 research themes over the past four decades. *Frontiers in Ecology and the Environment* 17:109–116.

353 Muñoz G., Kissling WD., Loon EE van. 2019. Biodiversity observations miner: A web application to  
 354 unlock primary biodiversity data from published literature. *Biodiversity Data Journal* 7.

355 Open Science Collaboration. 2015. Estimating the reproducibility of psychological science. *Science*  
 356 349:aac4716–aac4716.

357 Piper AM., Batovska J., Cogan NOI., Weiss J., Cunningham JP., Rodoni BC., Blacket MJ. 2019.  
 358 Prospects and challenges of implementing DNA metabarcoding for high-throughput insect surveillance.  
 359 *GigaScience* 8.

360 Piwowar H., Priem J., Larivière V., Alperin JP., Matthias L., Norlander B., Farley A., West J., Haustein  
 361 S. 2018. The state of OA: A large-scale analysis of the prevalence and impact of open access articles.  
 362 *PeerJ* 6:e4375.

363 Sinclair L., Ijaz UZ., Jensen LJ., Coolen MJ., Gubry-Rangin C., Chroňáková A., Oulas A., Pavloudi C.,  
 364 Schnetzer J., Weimann A., Ijaz A., Eiler A., Quince C., Pafilis E. 2016. **Seqenv**: Linking sequences to  
 365 environments through text mining. *PeerJ* 4:e2690.

366 Usai A., Pironti M., Mital M., Mejri CA. 2018. Knowledge discovery out of text data: A systematic  
 367 review via text mining. *Journal of Knowledge Management* 22:1471–1488.

368 Walsh E., Cherbuin N. 2018. Mapping the literature on nutritional interventions in cognitive health: A  
 369 data-driven approach. *Nutrients* 11:38.