

R tools for accessing research literature for text mining

Scott Chamberlain^{*,a}

^a*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's 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.

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 used widely throughout many academic fields; tools in R for text mining are of particular importance because they can be adopted by academics rapidly.

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

34 **Digital articles: technical aspects**

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

37 The machine readable articles are those in XML¹, JSON², or plain text format. The former two, XML
38 and JSON, are the best machine readable types because they are structured data³, whereas plain text
39 has no structure - it's simply a set of characters with line breaks and spaces in between.

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

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

53 **Digital articles: the access landscape**

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

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

¹<https://www.w3.org/TR/xml/>

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

³https://en.wikipedia.org/wiki/Data_model

⁴<https://en.wikipedia.org/wiki/PDF>

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

62 With respect to paywalled articles, access varies by institution, depending on each institution’s publisher
63 contracts. MORE ABOUT THIS ...

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

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

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

75 **The discovery problem**

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

78 Some of the discovery difficulty relates to the fact that there are so many places to search for articles;
79 a non-exhaustive list: Google Scholar, Microsoft Academic Research, Scopus, ScienceDirect, Web of
80 Science, Pubmed/Entrez, Europe PMC, Directory of Open Access Journals, Open Knowledge Maps,
81 CORE, Fatcat, and more. It’s probably difficult to know where the best place is to search. Some of
82 these are paywalled (e.g., Web of Science), and some are not.

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

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

97 **Data sources**

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

⁵<https://api.crossref.org>

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

⁷<http://bit.ly/KYQ1Rd>

⁸<http://api.plos.org/solr>

⁹<http://bit.ly/J9S616>

¹⁰<https://arxiv.org/help/api/index>

¹¹<https://dev.springer.com/>

¹²<http://www.biorxiv.org/>

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

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

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

¹⁶<https://doaj.org/api/v1/docs>

¹⁷<https://www.jstor.org/df/>

¹⁸<https://pub.orcid.org/>

¹⁹https://en.wikipedia.org/api/rest_v1/#/Citation/getCitation

²⁰<http://opencitations.net/>

Data Provider	Content Type	rOpenSci Package	Documentation
Fatcat	Metadata	none	21
SHERPA/RoMEO	Journal Level Metadata	rromeo	22
CORE	Full content/Metadata	rcoreoa	23
Dissemin	Metadata	dissemr	24

104 *Crossref/Datacite*

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

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

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

116 Datacite is similar to Crossref, but focuses on datasets instead of articles. The main interface for
 117 Datacite in R is [rdatacite](#).

118 *Biodiversity Heritage Library*

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

²¹<https://fatcat.wiki/>

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

²³<https://core.ac.uk/>

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

122 *Public Library of Science*

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

127 *Elsevier/Scopus*

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

134 *arXiv/bioRxiv*

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

138 *Pubmed/PMC/Europe PMC*

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

142 *Microsoft Academic Research*

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

146 *Directory of Open Access Journals*

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

150 *JSTOR*

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

155 *ORCID*

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

161 *Citoid/Open Citation Corpus*

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

166 *Fatcat*

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

170 *SHERPA/RoMEO*

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

174 *CORE*

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

178 *Dissemin*

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

182 **fulltext: a toolset for text mining in R**

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

188 *Search*

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

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

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

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

202 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
```

203 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)
```

204 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"
```

205 *Links*

206 `ft_links()` provides two pathways to get links (URLs) for articles, with a choice of four different data
207 sources (PLOS, BMC, Crossref, Entrez). First, you can use `ft_search()`, then pass the output of that
208 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 ...
```

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

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

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

215 *Abstracts*

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

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

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

224 *Fetch full text*

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

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

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

239 **How to text mine from R: Three case studies**

240 *Case study 1: Citation mining*

241 In this example, xxxx

242 *Load libraries*

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

243 *rcrossref*

244 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>, ...

```

245 Case study 2: Abstract mining

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

249 As an example, let's say you xxxx

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

250 *xxxxx*

251 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")

```

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

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

253 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:
```

254 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.</
```

255 *Case study 3: Full text mining*

256 In this example, xxxx

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

257 *Search for articles*

258 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:
#> Returned:
#> [PLOS: 10; BMC: 0; Crossref: 0; Entrez: 0; arxiv: 0; biorxiv: 0; Europe PMC: 0; Scopus: 0; .
```

259 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
```

260 *Get full text*

261 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 ...
```

262 *Extract text from pdfs*

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

266 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
```

267 *Extract text chunks*

268 Requires the [pubchunks](#) library. Here, we'll search for some PLOS articles, then get their full text, then
 269 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
```

270 **Future directions**

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

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

279 Future work for `fulltext` includes:

- 280 1. Adding more publisher plugins
- 281 2. Fine tuned user control over publishers
- 282 3. Improve VPN/proxy controls
- 283 4. Incorporate more search engines to help resolve URLs for fulltext versions
- 284 5. Improve documentation

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

- 286 1. provide XML if they have it
- 287 2. not change URL patterns so often, or at all
- 288 3. maintain consistent URL patterns among journals, years, etc.
- 289 4. keep their Crossref metadata up to date
- 290 5. open up their citation data

291 **Acknowledgments**

292 XXXX

293 Data Accessibility

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

298 References

- 299 Ba M., Bossy R. 2016. Interoperability of corpus processing work-flow engines: The case of alvisnlp/ml
300 in openminted. In: *Proceedings of the workshop on cross-platform text mining and natural language*
301 *processing interoperability (interop 2016) at Irec*. 15–18.
- 302 Camerer CF., Dreber A., Forsell E., Ho T-H., Huber J., Johannesson M., Kirchler M., Almenberg
303 J., Altmejd A., Chan T., Heikensten E., Holzmeister F., Imai T., Isaksson S., Nave G., Pfeiffer T.,
304 Razen M., Wu H. 2016. Evaluating replicability of laboratory experiments in economics. *Science*
305 351:1433–1436.
- 306 Camerer CF., Dreber A., Holzmeister F., Ho T-H., Huber J., Johannesson M., Kirchler M., Nave G.,
307 Nosek BA., Pfeiffer T., Altmejd A., Buttrick N., Chan T., Chen Y., Forsell E., Gampa A., Heikensten
308 E., Hummer L., Imai T., Isaksson S., Manfredi D., Rose J., Wagenmakers E-J., Wu H. 2018. Evaluating
309 the replicability of social science experiments in nature and science between 2010 and 2015. *Nature*
310 *Human Behaviour* 2:637–644.
- 311 Cañada A., Capella-Gutierrez S., Rabal O., Oyarzabal J., Valencia A., Krallinger M. 2017. LimTox: A
312 web tool for applied text mining of adverse event and toxicity associations of compounds, drugs and
313 genes. *Nucleic Acids Research* 45:W484–W489.
- 314 Chaix E., Deléger L., Bossy R., Nédellec C. 2019. Text mining tools for extracting information about
315 microbial biodiversity in food. *Food Microbiology* 81:63–75.
- 316 Ding Z., Li Z., Fan C. 2018. Building energy savings: Analysis of research trends based on text mining.
317 *Automation in Construction* 96:398–410.
- 318 Kong X., Gerstein MB. 2018. Text mining systems biology: Turning the microscope back on the
319 observer. *Current Opinion in Systems Biology* 11:117–122.

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

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

324 Open Science Collaboration. 2015. Estimating the reproducibility of psychological science. *Science*
325 349:aac4716–aac4716.

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

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

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