Softcite Dataset: A Dataset of Software Mentions in Biomedical and Economic Research Publications

Caifan Du (Primary contact)
cfdu@utexas.edu
TEL: 512-431-0178
1616 Guadalupe St, Austin, TX, 78701-1213, USA
University of Texas at Austin

Johanna Cohoon
jlcohoon@utexas.edu
TEL: 434-249-1991
1616 Guadalupe St, Austin, TX, 78701-1213, USA
University of Texas at Austin

Patrice Lopez
patrice.lopez@science-miner.com
SCIENCE-MINER, France

James Howison
jhowison@ischool.utexas.edu
TEL: 315-395-4056
1616 Guadalupe St, Austin, TX, 78701-1213, USA
University of Texas at Austin

Abstract

Software contributions to academic research are relatively invisible, especially to the formalized scholarly reputation system based on bibliometrics. In this paper we introduce a gold-standard dataset of software mentions from the manual annotation of 4,971 academic PDFs in biomedicine and economics. The dataset is intended to be used for automatic extraction of software mentions from PDF format research publications by supervised learning at scale. We provide a description of the dataset and an extended discussion of its creation process, including improved text conversion of academic PDFs. Finally, we reflect on our challenges and lessons learned during the dataset creation, in hope of encouraging more discussion about creating datasets for machine learning use.

Softcite Dataset: A Dataset of Software Mentions in Biomedical and Economic Research Publications

Introduction

Software is crucial to contemporary scholarship because it is a key element of the "knowledge infrastructure" supporting research (Edwards et al., 2013; Howison & Bullard, 2016; Mayernik et al., 2017). Yet, like other infrastructural work (Star & Ruhleder, 1996), the creation and maintenance of research software has been found to be relatively invisible, especially to the formalized scholarly reputation system built on bibliometrics (Howison et al., 2015; Howison & Herbsleb, 2011; Mayernik et al., 2017). This is because practices of mentioning software in publications are diverse, hard to observe, and hard to use for software creators to demonstrate their impact. While other measures of impact such as downloads or "forks" of code repositories are sometimes used, they are difficult to compare to bibliometrics and their relationship to actual use in research is less clear than mentions in publications.

Invisibility leads to the under-acknowledgement of software as meaningful academic contributions, and thus reduces incentives for researchers and funders to undertake software work. As a result the quality of software degrades and the effectiveness of scholarly funding and scholarship is undermined.

To support increased visibility of software in the research literature we present the *Softcite dataset*, a "gold-standard" corpus of software mentions intended for training and testing supervised machine learning models for software entity recognition at scale. The software mentions are annotated within full-text open access biomedical and economic academic articles converted directly from PDFs.

This article is organized in five sections. First, we review existing work on software visibility and entity extraction. Second, we describe our annotated corpus, introducing its data sources, creation, format, and content. Third, to enhance provenance, we provide details of our process. Fourth, we reflect on our project, pointing to challenges and lessons learned. Finally, we discuss future work.

Related Work

Existing work seeks to make software more visible in research literature by (1) changing authoring and publishing practices, and (2) identifying software in currently published literature using text processing and entity recognition.

Changing authoring and publishing practices involves disseminating clear standards for formal software citation as well as changing the behaviors of researchers and publishers. For example, leveraging FORCE11's experience in promoting data citation, the FORCE11 Software Citation Working Group[1] and the subsequent Software Citation Implementation Working Group[2] have been leading this effort (e.g., Chue Hong, Allen, Gonzalez-Beltran, et al., 2019; Chue Hong, Allen, Gonzalez-Beltran, et al., 2019; Smith et al., 2016).

Mozilla Science, GitHub, Zenodo, and Software Heritage facilitate archiving and identification of software releases (Di Cosmo, 2020; Di Cosmo, Gruenpeter, & Zacchiroli, 2019) to support new software citation practices. Digital repositories and registries such as HAL (Hyper articles en Ligne), Astrophysics Source Code Library (ASCL), and swMath catalogue software, provide metadata and links to software (Allen & Schmidt, 2014; Di Cosmo, Gruenpeter, Marmol, et al., 2019; Greuel & Sperber, 2014). swMath also tracks citations to the software (Greuel & Sperber, 2014). The Research Resource Identification initiative (RRID) provides unique identifiers for research resources including software (A. E. Bandrowski & Martone, 2016; A. Bandrowski et al., 2016). Innovative venues such as the Journal of Open Research Software (JORS), the Journal of Open Source Software (JOSS), and ACM Transactions on Mathematical Software, publish peer-reviewed "software papers".

A third approach to changing current practices is to show how to cite software. Software creators can declare the preferred citation requests using ecosystem specific forms (e.g., the R CITATION file), or general schema such as CodeMeta and CITATION.cff, or through free-form statements on websites. A specialized search engine, CiteAs.org, ("CiteAs.org", n.d.) helps users find these statements.

While these efforts aim to change citation practices moving forward, a second strategy seeks to work with existing publications and to identify software mentions within them.

Some efforts focus on formal citations. For example, bibliometric services like the Web of Science Data Citation Index (Park & Wolfram, 2019) have started to index software citations in addition to data. Yet empirical studies show that less than half of software mentions have formal citations (Howison & Bullard, 2016).

Accordingly, other efforts focus on identifying mentions in full-text. This approach is called "named entity recognition" (NER). NER does not require change by authors and publishers and thus complements efforts to disseminate new behaviors and new systems of software citation and indexing. Krüger and Schindler (2020) reviews studies using NER techniques to identify software, together with data and databases, categorizing 48 articles into four main approaches: term search (12 studies, 1 on software), manual extraction (12 studies, 6 on software), rule-based extraction (16 studies, 5 on software), and supervised learning (4 studies, 2 on software). We complement their review with newly identified studies focusing on software recognition. In total, we found 18 studies identifying software mentioned in research: term search (1 study), manual annotation (6 studies), rule-based extraction (7 studies), and supervised learning (4 studies).

Term search (e.g., Russell et al. (2018)) utilizes known identifiers as search terms, including URLs, DOIs, or software names. Results of term search have high precision, identifying software consistently mentioned by widely-known and recognizable names, but cannot discover previously unknown software entities, nor deal with variation or ambiguity in naming, making recall unknown but likely low.

Rule-based extraction relies on syntactic heuristics specifying patterns common to software mentions. For instance, leveraging the URL patterns specific to R package management system, Li and Yan (2018) achieved a precision of .84 and a recall of .87 in identifying mentions of R packages in 13,694 PLoS journal articles. Rules, once created, scale well, but are limited to specific domains or specific software ecosystems (e.g., Greuel and Sperber (2014), Hwang et al. (2019)).

Manual cataloging of mentions by humans performs better in identifying alternative phrasing that mentions software in literature. It produces data with both high recall and precision, but requires great commitment of time and engagement (Howison et al., 2015; McLennan & Kennell, 2010). Accordingly, existing manually extracted datasets of software

mentions are relatively small: 90 (Howison & Bullard, 2016), 40 (Nangia & Katz, 2017), 166 (Allen et al., 2018), and 85 articles (Duck et al., 2015).

Supervised machine learning is a promising solution to extend the usefulness of manually annotated data. For example, Krüger and Schindler (2020) used a deep learning model trained on the bootstrapped dataset generated by Duck et al. (2016) and achieved a precision of .90 and a recall of .94. But, as in many applications, broadly generalizable supervised learning models need large quantities of gold-standard data to build confidence in results (Halevy et al., 2009).

The relatively small manually annotated datasets mean that current applications of supervised learning in software NER often adopt so-called "weak" supervision strategies such as bootstrapping. Bootstrapping relies on whitelists of software names, usually drawn from external dictionaries, or small sets of "seed" mentions identified in a rule-based fashion (e.g., Duck et al. (2015), Duck et al. (2016), Ozyurt et al. (2016), Pan et al. (2016), Pan et al. (2015), Schindler et al. (2020)). These "seeds" are then used as examples to train supervised learning models able to identify software beyond the seeds.

Bootstrapping is promising, nevertheless it is costly to evaluate its quality because mentions are sparse within publications. While false positives can be identified by examining only the text of the identified mentions themselves, checking for false negatives requires manually reviewing large amounts of full-text. Sampling can help, checking randomly selected sections of full-text for false negatives, however the effort is still substantial, and at the limit approaches the cost of creating a gold-standard dataset. Bootstrapping tends to offer good precision, but relatively low recall: e.g., Pan et al. (2015) obtained an F-score of .58, and a recall of .42 assessed by re-coding a sample of full-text.

Combining approaches can yield improvements. For example, the Resource Disambiguator for the Web (RDW) text mining pipeline boosts bootstrapped software NER with regular expression search for URL and RRIDs, but lacking a true gold-standard dataset, uses post-hoc inspection to estimate high recall (.86) and precision (.94) (Hsu et al., 2019; Ozyurt et al., 2016).

Full supervision can significantly improve the scope and performance of NER, as well as provide benchmarks for evaluating bootstrapping and other weak supervision

approaches, but high cost is unavoidable. More full-text annotation of software mentions, especially across disciplines, will boost efforts throughout the community.

NER is also limited by the availability of usable full-text of academic articles. However the default format for scholarly article publishing is the PDF. PDF is good at rich content presentation but extremely challenging for machine processing. Increasingly publishers are providing XML versions of article text, and XML is much more suitable for machine processing, but coverage remains only partial and uneven across domains, especially in combination with open access licensing, requiring contract negotiations. Thus, existing NER in scholarly text are often limited to article abstracts or sections (e.g., Ozyurt et al., 2016; Schindler et al., 2020; Wei et al., 2020), rather than enabling full-scale "distant reading" for bibliometric and text analysis of the literature (Mehta et al., 2017). While XML is increasingly available from traditional publishers, we think it is unlikely that standardized XML will ever achieve full coverage, especially considering the growth of open access pre-print and institutional repositories. Further, different publishers provide different XML formats, requiring pipelines to manage differences between them. Given all above, we sought to create a gold-standard dataset of annotated software mentions directly from PDF academic publications.

The Softcite dataset

We created the Softcite dataset, which is presented as a TEI/XML file with paragraphs from articles automatically converted from PDF publications, as well as article level metadata. In these paragraphs are annotations of 4,093 software mentions, together with 2,541 annotations of their details including `publisher`, `version`, and `URL`. Below we describe the data sources and their annotation, dataset format and content, and validation through prototype NER training.

Annotation

To create this dataset we manually annotated 4,971 full-text research articles published during the period 2000-2010: 2,521 from biomedicine and 2,450 from economics. All articles were open access and obtained as PDFs.

We chose biomedicine as a field where software is known to have made a large and recent impact, including the formation of the new sub-field of bioinformatics. To increase the diversity of publishers and citation practices in our dataset, we chose economics as a contrast to biomedicine. Still more fields will be needed for enhancing cross-disciplinary NER.

Biomedical articles were randomly selected from the PubMed Central (PMC) Open Access (OA) Subset [3]. PMC is a quality source of OA publications in biomedicine with high coverage. We downloaded and randomly sorted the index of DOIs, enabling us to expand our sample if needed.

Economic articles were retrieved from Unpaywall ("Unpaywall", n.d.), which gives access to the largest collection of open access articles via CrossRef DOIs (Piwowar et al., 2018). We used the "subfield" attribute of the Classification of Scientific Journals (Eric, 2016) to identify a list of ISSNs that we queried in Unpaywall, pulling 5,000 random DOIs for which Unpaywall had found an open access PDF link. Both collections included some non-article PDFs (e.g., journal front-matter) which annotators skipped.

We employed 36 paid student annotators over two years. Annotators were trained and then assigned PDF articles to read, copying quotes containing software mentions into working files and applying our annotation scheme. As a group we first annotated the biomedical articles, then switched to economics approximately half-way through our annotation budget. Training effectiveness was assessed through a period of double annotation, after which articles were annotated by a single person.

The entire dataset underwent review and curation by an author of this paper, working in consultation with two other authors as expert annotators. Curation resolved inconsistencies, including conceptual inconsistencies (such as whether web services like Google were software) and fatigue inconsistencies (by inspecting all occurrences of identified software names).

Further details of annotation, training, alignment, and curation are documented later in this paper in detail sufficient to understand the provenance of the dataset (Gebru et al., 2018; Thomer et al., 2018).

The annotation scheme represented in the final Softcite dataset is shown in

Figure 1.

Figure 1

Final annotation scheme applied to the Softcite dataset

Annotation field	Definition	Example
software	Whether a mentioned string in the article text refers to a specific piece of software. The string value should be the name of the mentioned software. If the acronym of the software name is present, it should be included in the annotation as well.	R Statistical Package for the Social Sciences (SPSS) G Power PyMOL
publisher	The publisher of the mentioned software. It could be organizations or individuals. Only the name of the organization/individual is annotated, other information such as the geo-location of the organization or institutional affiliation of individuals are not included in the annotation.	R Development Core Team GraphPad Software Inc National Institutes of Health Indica Labs University of Geneva J. Dempster Robert Jones and Agnes Hunt Orthopaedic Hospital
version	The version of the software is mentioned in the text. It could be a version number, or the version release date. Prefixes such as "v.", "ver.", or "version" are not annotated.	9.2 2.2.25+ 2009 1.1 RC2 4.5 build 1416 2008.02 XD
url	The URL that provides access to the mentioned software.	http://www.R-project.org www.ebi.ac.uk/Tools/msa/clustalw2 http://bioweb2.pasteur.fr/projects/mobyle/downloads.html https://github.com/sylvainforet/libngs
certainty	The original annotator of the software mention self-rated an integer certainty score ranging from 1 to 10.	1 7 10

Publication as TEI/XML

The PDFs were converted to machine-readable text using GROBID ("GROBID", 2008–2020), which uses machine learning to perform "structure-aware" text extraction from PDFs ("Document engineering | science-miner", n.d.). GROBID produces TEI/XML representations of academic articles including title, author, DOI, section titles, figures, tables, and in-text citations and bibliography items. We used Python ("Python", n.d.) scripts to align the full-text quotes from the annotators with the GROBID produced

TEI/XML text.

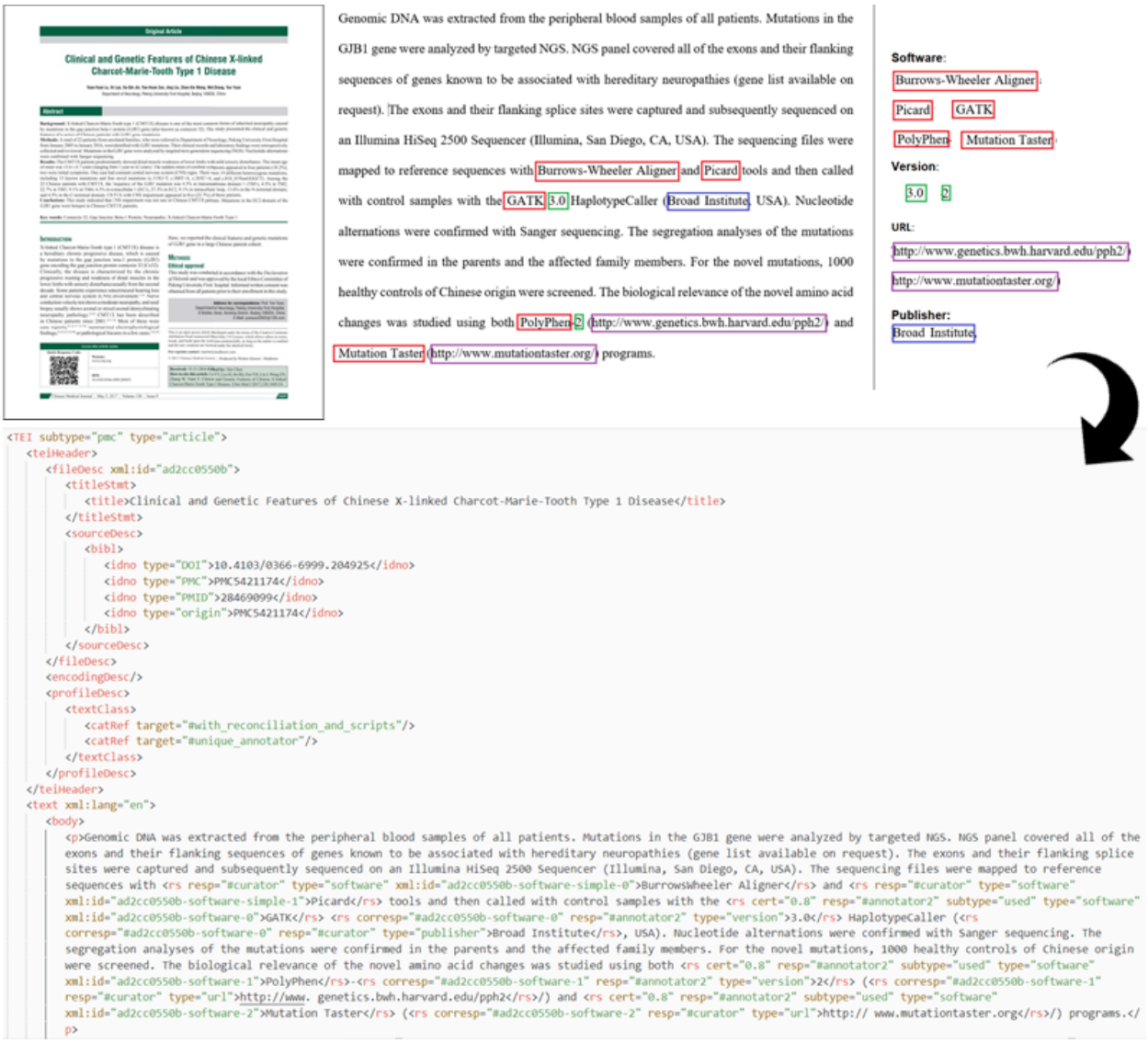
The corpus file contains one TEI entry for every scholarly publication reviewed by annotators, encoded following guidelines from the TEI consortium (TEI Consortium, 2020). Each article entry is a `<TEI>` element with metadata including title, article source (PMC or economics), and publication identifier (DOI, PMCID, PMID). Each paragraph containing at least one manually annotated software mention is encoded under the TEI `<body>` element. These paragraphs thus contain both positive examples of software mentions and nearby sentences without mentions that can be used as negative examples. Mentions of software and their attributes are encoded with `<rs>`, the TEI element standing for "referencing string" used for entity markup[4]. Each `<rs>` element has a `type` attribute indicating its category: `software` and details about the entity (`publisher`, `version`, and `url`). Annotators have a unique, anonymized, identifier in the `resp` attribute (if the annotation was modified during expert review, its `resp` attribute became `resp="#curator"`). Figure 2 shows how software mentions identified in an article are encoded as `<TEI>` elements in the Softcite TEI/XML corpus.

We provide clarity on the annotation and review status of each annotated article using the `<catRef>` element. While all the software mentions in the dataset were validated by expert reviewers to ensure they are gold-standard, originally all the articles were annotated by one or more annotators. Thus, we have `<catRef target="#multiple_annotator">` or `<catRef target="#unique_annotator">` to indicate article annotation status before expert review. Since all the included paragraphs were validated by expert review, articles that contain software mentions also have `<catRef target="with_reconciliation_and_scripts">`. Articles that did not have mentions did not undergo expert review and do not have this label.

While many articles we annotated are gold Open Access and would allow full redistribution for research use, to ensure the ability to expand the dataset on equal terms with sources beyond gold OA, we distribute paragraphs as short quotations from publications, instead of the publication full-text. For those interested in further text from the papers, inside the TEI/XML corpus we document the version and configuration of GROBID actually used to create the text in the corpus file.

Figure 2

Annotated software mentions in TEI/XML originally from academic PDF




```
<TEI subtype="pmc" type="article">
  <teiHeader>
    <fileDesc xml:id="ad2cc0550b">
      <titleStmt>
        <title>Clinical and Genetic Features of Chinese X-linked Charcot-Marie-Tooth Type 1 Disease</title>
      </titleStmt>
      <sourceDesc>
        <bibl>
          <idno type="DOI">10.4103/0366-6999.204925</idno>
          <idno type="PMC">PMC5421174</idno>
          <idno type="PMID">28469099</idno>
          <idno type="origin">PMC5421174</idno>
        </bibl>
      </sourceDesc>
    </fileDesc>
    <encodingDesc/>
    <profileDesc>
      <textClass>
        <catRef target="#with_reconciliation_and_scripts"/>
        <catRef target="#unique_annotator"/>
      </textClass>
    </profileDesc>
  </teiHeader>
  <text xml:lang="en">
    <body>
      <p>Genomic DNA was extracted from the peripheral blood samples of all patients. Mutations in the GJB1 gene were analyzed by targeted NGS. NGS panel covered all of the
      exons and their flanking sequences of genes known to be associated with hereditary neuropathies (gene list available on request). The exons and their flanking splice
      sites were captured and subsequently sequenced on an Illumina HiSeq 2500 Sequencer (Illumina, San Diego, CA, USA). The sequencing files were mapped to reference
      sequences with <rs resp="#curator" type="software" xml:id="ad2cc0550b-software-simple-0">BurrowsWheeler Aligner</rs> and <rs resp="#curator" type="software"
      xml:id="ad2cc0550b-software-simple-1">Picard</rs> tools and then called with control samples with the <rs cert="0.8" resp="#annotator2" subtype="used" type="software"
      xml:id="ad2cc0550b-software-0">GATK</rs> <rs corresp="#ad2cc0550b-software-0" resp="#annotator2" type="version">3.0</rs> HaplotypeCaller (<rs
      corresp="#ad2cc0550b-software-0" resp="#curator" type="publisher">Broad Institute</rs>, USA). Nucleotide alternations were confirmed with Sanger sequencing. The
      segregation analyses of the mutations were confirmed in the parents and the affected family members. For the novel mutations, 1000 healthy controls of Chinese origin
      were screened. The biological relevance of the novel amino acid changes was studied using both <rs cert="0.8" resp="#annotator2" subtype="used" type="software"
      xml:id="ad2cc0550b-software-1">PolyPhen</rs>-<rs corresp="#ad2cc0550b-software-1" resp="#annotator2" type="version">2</rs> (<rs corresp="#ad2cc0550b-software-1"
      resp="#curator" type="url">http://www. genetics.bwh.harvard.edu/pph2</rs>/) and <rs cert="0.8" resp="#annotator2" subtype="used" type="software"
      xml:id="ad2cc0550b-software-2">Mutation Taster</rs> (<rs corresp="#ad2cc0550b-software-2" resp="#curator" type="url">http:// www.mutationtaster.org</rs>/) programs.</
      p>
```

Dataset content

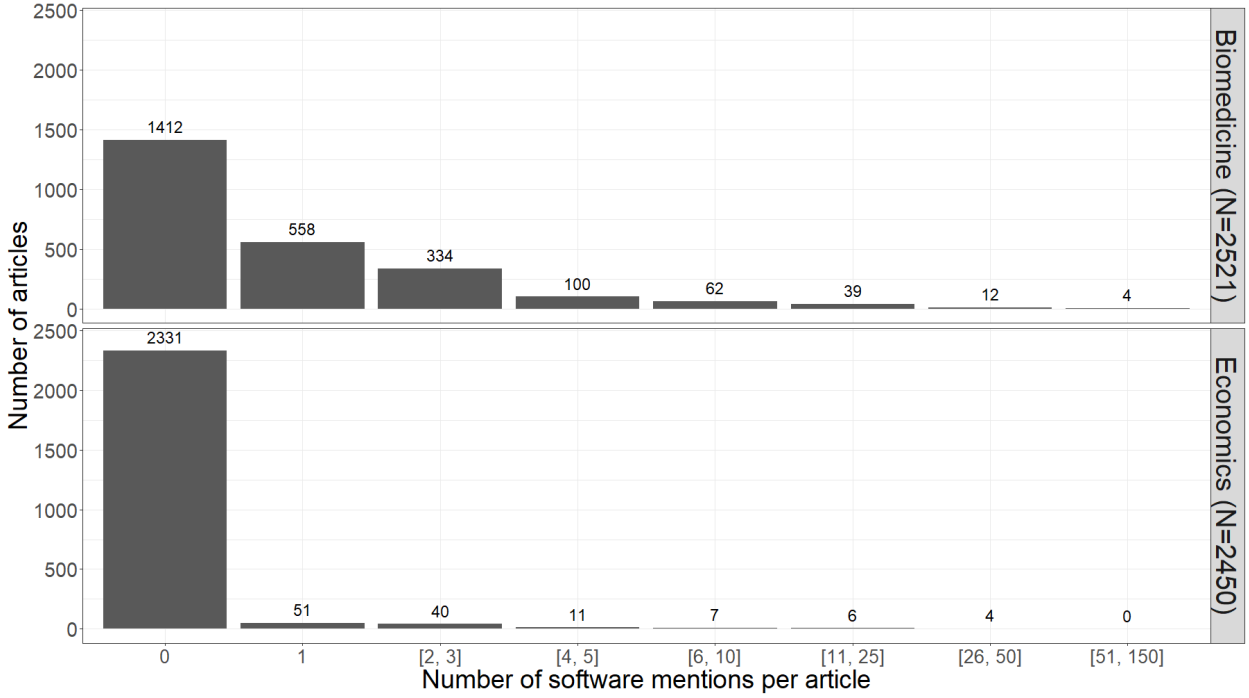
Mention density of articles

Software mentions are disproportionately distributed across the annotated articles. 71% (N=3,743) of the articles in our dataset do not mention any software at all; 94% of the articles have fewer than ten software mentions. The farthest outlier is a biomedical article containing 152 software mentions. Figure 3 shows numbers of articles with varying levels of mention density (number of mentions per article). Overall, as Figure 4 suggests, biomedical

articles have far more software mentions than economic articles in our dataset. (Figure 3 to Figure 4 were produced using R ("The R Project for Statistical Computing", n.d.) package ggplot2 by Wickham (2016))

Figure 3

How many articles have software mentions in two annotated article sets

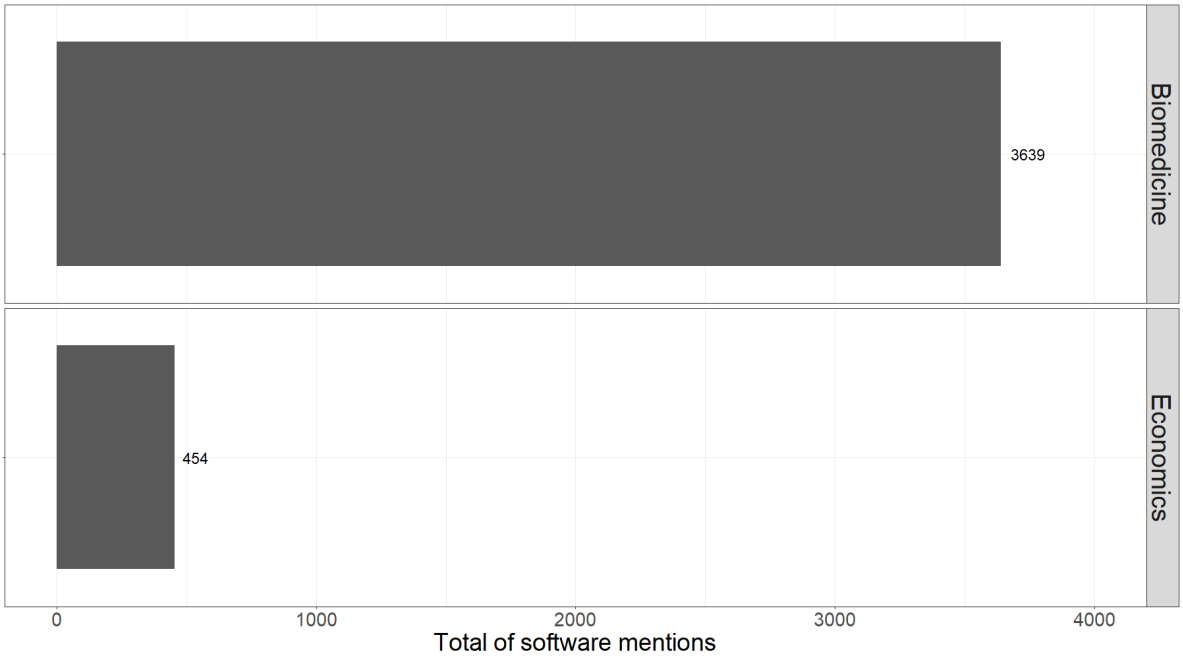


Mention details

In our dataset, less than half (44%) of the annotated software mentions include details such as version, publisher, or URLs. Only 31% of the software mentions include a software `publisher`, and only 27% have `version`. Only 17% have an `URL` associated with the mention, pointing to issues with potential access. Software mentions in economic articles contain even fewer software attributes than biomedical articles. Detailed counts of software mentions with different attributes can be found in Figure 5 (made using R package UpSetR by Conway et al. (2017)).

Figure 4

Total number of software mentions in two annotated article sets



Dataset validation

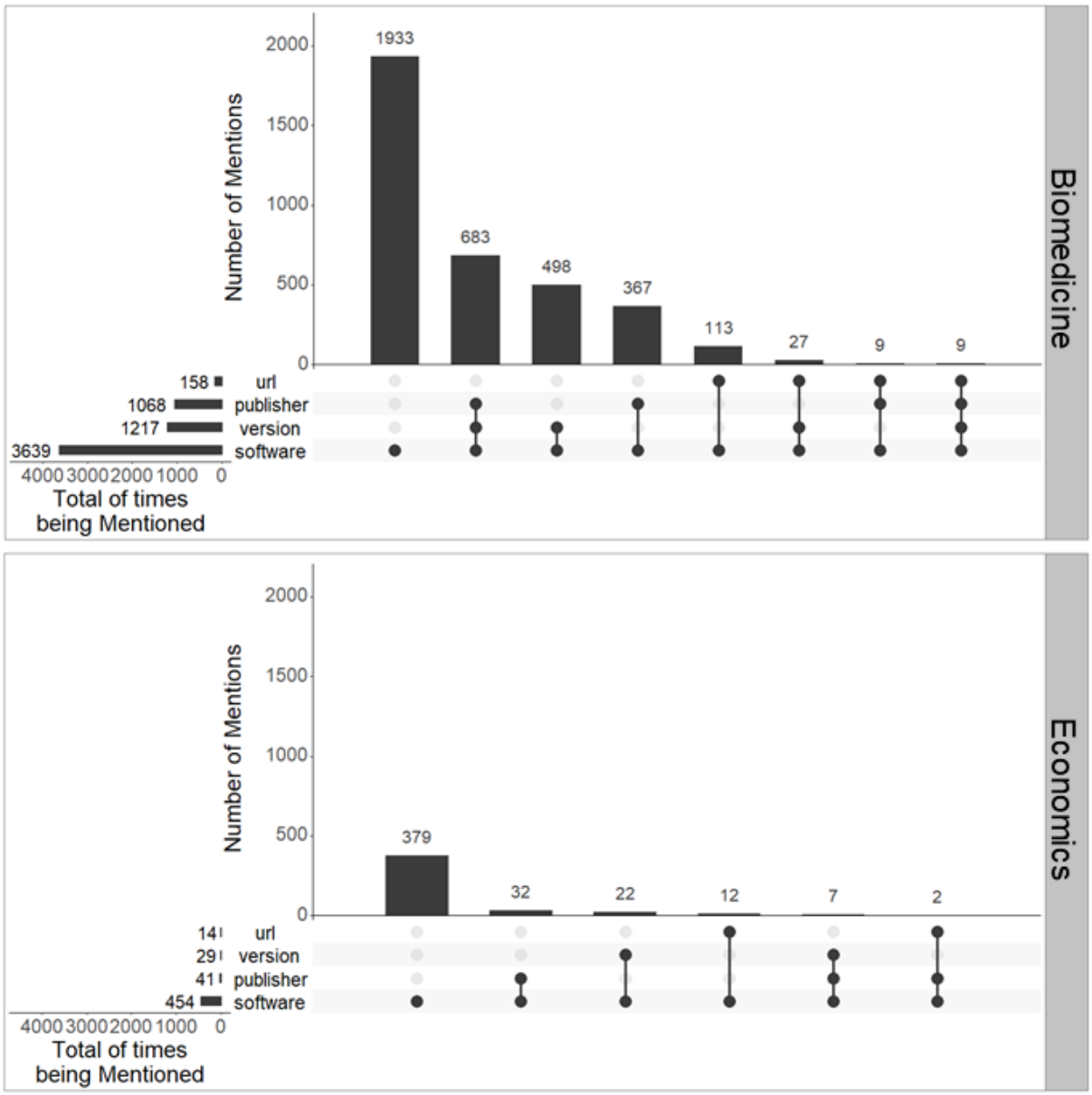
We demonstrate the usefulness of the Softcite dataset by employing it for training a baseline NER model via supervised learning. We implemented a linear CRF (Conditional Random Field) model (Lafferty et al., 2001), a supervised learning approach for sequence labelling. This baseline model trained with our dataset shows good performance considering the results reported by prior studies using bootstrapping and rule-based extraction, with similar precision and higher recall.

We report span-level precision, recall, and F-score of the CRF model using 10-fold cross-validation on the Softcite dataset, both negative and positive examples included as training data. Among all the annotation fields, we have obtained an average precision of .86, recall of .76, and F-score of .81. The metrics for each annotation field are detailed in Table 1.

Other state-of-art deep neural architectures, such as Bidirectional LSTM with a CRF activation layer (BidLSTM-CRF) and ELMo embeddings (Peters et al., 2018) can be trained on our corpus to achieve better NER accuracy. Here we do not report our draft

Figure 5

How different software attributes are mentioned along with software



results with those deep learning architectures, leaving "low hanging fruit" (Raymond, 1998) to motivate uptake and build a user community around this dataset.

Dataset creation details

An overview of our dataset creation process is illustrated in Figure 6.

Table 1

CRF-based NER results for software mentions on the Softcite dataset, averaged over 10 folds

Fields	Precision	Recall	F-score
software	85.87	74.39	79.71
publisher	84.63	74.95	79.46
version	90.6	85.58	87.98
url	72.62	62.36	66.72
micro-average	86.27	76.23	80.94

In this section we discuss the creation process of the Softcite dataset. We provide greater detail for three reasons.

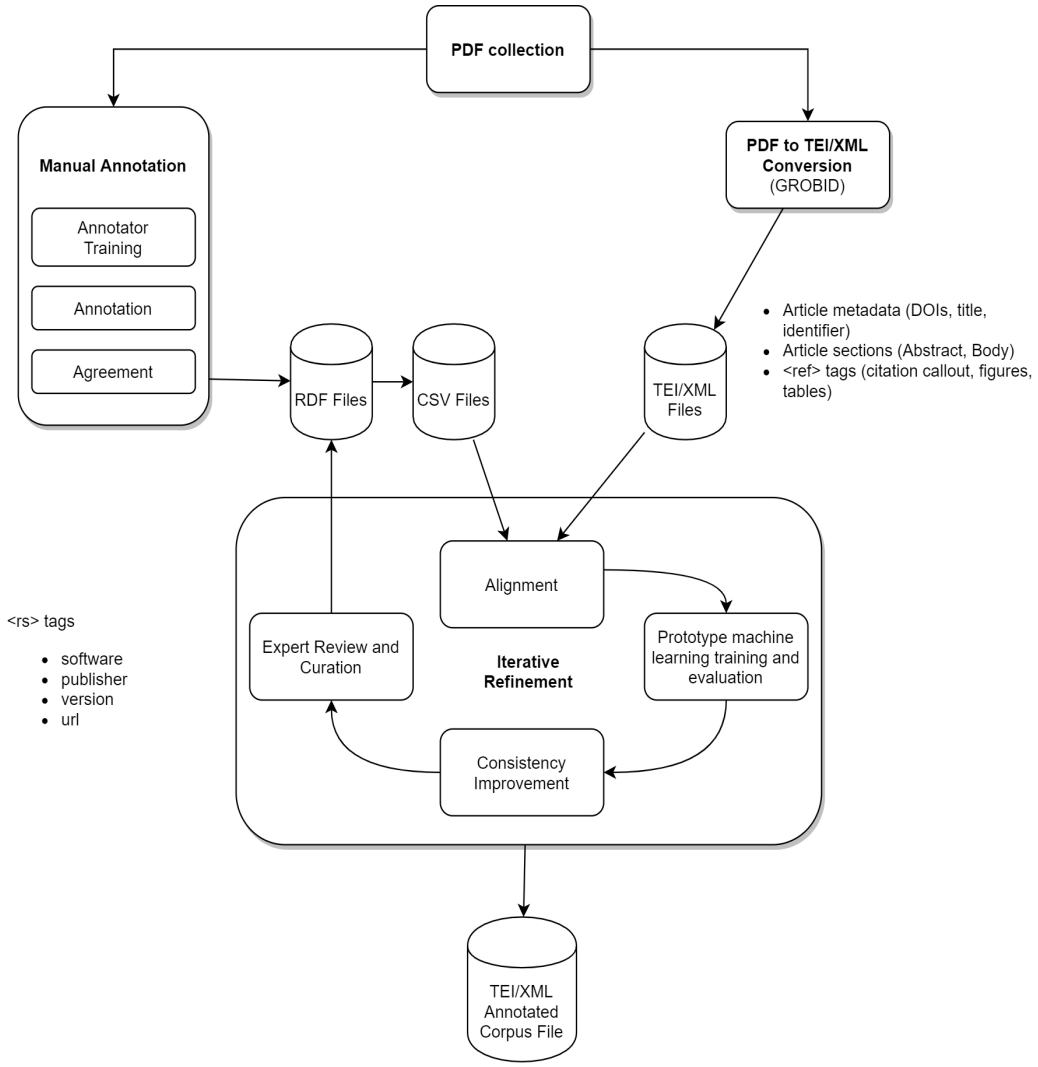
First, machine learning is increasingly understood to be sensitive to the context of its training data. Machine learning models trained on one dataset have been found to fail in a deployment context unmatched with the training/testing data, potentially reproducing, or even exacerbating unwanted social biases (Bender & Friedman, 2018; Gebru et al., 2018; Jo & Gebru, 2020; Selbst et al., 2019). Explaining how a dataset is created can help future users better harness the potential of the dataset, while facilitating consideration of unintended social impacts.

Second, the annotation and curation of large datasets are not always treated as "front-stage" research activities (Goffman, 1959), and are therefore less visible in data-intensive research reporting. However, the very process of creating and curating labels directly affects the validity and usability of the dataset, so reporting the "back-stage" data generation process increases the transparency and accountability of the dataset while providing methodological guidance for others.

Third, this account of dataset creation offers a human-readable rationale and guidance on how the provenance metadata could be used, complementing the metadata in the published dataset.

Figure 6

The creation process of the Softcite dataset



Genesis of the Softcite project

Prior empirical research showed that software is rarely cited, and inconsistently mentioned in scholarly literature (Howison & Bullard, 2016; Howison et al., 2015; Howison & Herbsleb, 2011), motivating us to work towards automatic extraction of informal software mentions. While other measures of impact and use are possible (e.g., downloads/installation data) mentions in publications link more directly to scientific impact and provide a visible public acknowledgement of intellectual contribution. (Howison et al., 2015). We used the manually extracted software mentions from Howison and Bullard

(2016) as our starting point.

We recruited and trained a team of graduate and undergraduate students on campus as annotators, paying $15 an hour. Several students from a local minority-serving institution were recruited for the project; unanticipated constraints on their work authorization documentation at our institution limited their contributions. Students mostly have backgrounds in information/computer science or the natural sciences, with a smaller number of students from the liberal arts or the social sciences. Students were restricted to work 10 hours weekly to mitigate fatigue-induced errors and balance contributions across annotators.

Training sessions were held in a large classroom. We provided tutorials on the annotation scheme and infrastructure (detailed below). We identified and resolved disagreements through group discussions as we improved the annotation scheme. The full scheme used by the annotators includes illustrative examples and is available online [5] and its history is visible in version control.

When annotators encountered a possible mention of software, we asked them to give their best guess of the mentioned entity type and self-rate a `certainty` score indicating their confidence about their guess. They were also encouraged to search the Web to validate their guess and to write down their reasoning in a free-text field. This step helps address the conceptual ambiguity of software: some entities are database systems, algorithms, or web platforms that arguably are software or made of software systems, and was useful in expert review. At first, we also instructed students to annotate software cited in the article reference list, and identify if the reference entry points to a software *publication*, *user guide*, *project page*, or *project name*, but, as detailed below, these were dropped as we were later able to automatically extract in-line citations and their linked reference items.

Finally, during initial annotation we had students annotate whether software appeared to have been used in the research, or whether it was mentioned for another reason such as comparison or as a metaphor, using a `software_was_used` code.

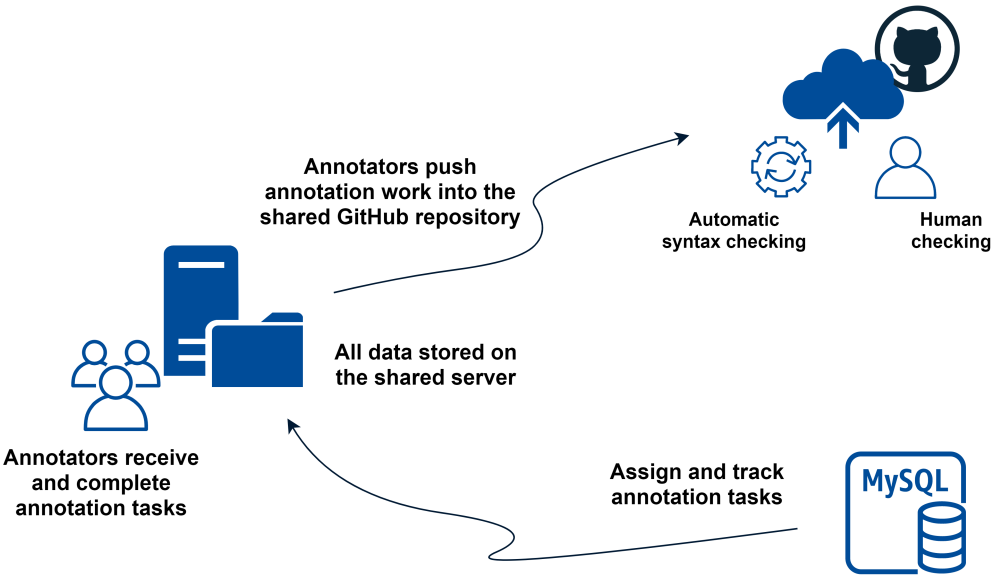
Collaboration infrastructure

Our workflow is illustrated in Figure 7. We worked as a distributed team with periodic meetings for training and agreement discussions. We collected annotations via a GitHub repository and each annotator had an individual fork in a home directory on our server. We assigned PDFs via a command-line script, tracked in an MySQL ("MySQL", n.d.) database; the script generated a template file in RDF turtle format and a link to a PDF article. The script assignment process enabled multiple annotators to be assigned to a single article.

Annotators downloaded and read the PDF article then filled out the RDF template in the Atom text editor ("Atom", n.d.). They were required to identify the software mention, then copy both the words denoting the software (e.g. the software name) as well as the surrounding context (e.g. the sentence containing the name), paste them into the RDF template, then indicate software attributes (e.g. the version), if any. Files were stored on a shared server and edited using the remote-ftp ("remote-ftp", n.d.) plugin.

Figure 7

The Softcite collaboration infrastructure



After annotation, annotators committed to their fork and pushed to GitHub, creating pull requests to our central repository, which were then reviewed by a doctoral student. We deployed Travis CI ("Travis CI", n.d.) to check the syntax of incoming

annotation work. All the collected annotations were parsed into a single RDF graph, facilitating export to CSV files, which were then used for analysis and alignment with TEI/XML text produced by GROBID.

We set up this infrastructure with the following considerations. First, the RDF graph offered more flexibility to represent semantic relations between annotations than other annotation tools. Second, initially our PDF-to-text conversion was unusable and manual conversion via copy-and-paste from PDFs was a temporary solution. Third, we intentionally adopted tools that afford more autonomy, avoiding dependencies on third-party annotation tools or time-consuming tool development. Fourth, working with version control, virtual machine, and data semantic technologies provided learning and teaching opportunities. Finally, the shared server backed-up all in-progress annotations meaning we were able to recover completed but uncommitted work left in individual forks.

Agreement assessment

We assessed inter-annotator agreement during training to identify needed clarification in the annotation scheme, a measure commonly used for demonstrating annotators' ability to label categories consistently (Artstein & Poesio, 2008). Since our entire dataset was reviewed by expert annotators, we did not rely on inter-annotator agreement as a measure of quality for our published dataset. During the assessment period we assigned 569 articles to multiple annotators.

In order to compare annotations from different annotators, we first needed to align the RDF-encoded mentions and their context quotes with article full-text extracted from PDFs to see if two annotators found a software mention at the same location. Initially, we experimented with several pdf-to-text conversion tools, including the pdftotext ("pdftotext", n.d.) routine called from the Poppler package ("Poppler", n.d.) and the rOpenSci [6] package fulltext (Chamberlain, n.d.). However, the text output was too inconsistent to allow successful alignment (e.g., headers, or lines from different columns interleaved).

We thus turned to the PMC OA XML documents to assess agreement (the availability of XML content was one reason that we had chosen to annotate articles from

PMC first). We used R packages Biostrings (Pagès et al., 2017) and tidyverse (Wickham et al., 2019) to identify overlapping quotes copied-and-pasted from PDF publications by different annotators, by aligning them with the XML version of the same publications. We were not able to align all quotes for several reasons including manual typos in annotated quotes and inconsistent copy-and-paste outcomes from PDFs rendered in varied PDF reader applications used by annotators. Counting overlap as agreement, we obtained a percentage agreement of 65.4% in the annotations of PMC articles.

Our understanding at the time was that there were three sources of disagreement: misalignment, fatigue, and annotation scheme clarity.

We anticipated that misalignment would improve with better PDF conversions. Indeed, after we added an NLP expert to our team and began working with GROBID PDF conversions (discussed below), we returned to our double-annotated articles and were able to improve alignment substantially and recalculate percentage agreement as 75.5% (The agreement statistic was computed using Java ("Java", n.d.) library DKPro Agreement (Meyer et al., 2014).) Even then, some misalignment remained. Manual inspection suggested that annotators had sometimes manually typed rather than copied from the PDF, and some issues remained with problematic fonts or layout as copy-and-paste does not always convert PDF content faithfully.

We addressed fatigue from the outset by limiting annotators to 10-hour weekly work. Nonetheless, we reasoned that the sparseness of software mentions and the reading level of the material mean that some missing annotations of software mentions are unavoidable. We did not think these false negatives would be systematic such that affect the machine learning performance as long as false negatives were not included in the training/testing data. To make this most likely, we eventually only include annotated text that has been fully reviewed and thus reached the gold-standard quality, and indicate their review status by marking them up with the `<catRef target="with_reconciliation_and_scripts">` attribute in the TEI/XML dataset.

To improve the annotation scheme, we added examples to illustrate the conceptual definitions and discussed them with the team. For example, when confusion arose among annotators about whether a programming language is software, we gave examples such as

"R" or "Perl", confirming that they are software in our annotation scheme. However, as we reasoned that we were unlikely to anticipate all conceptual issues in identifying software, we encouraged annotators to record their `certainty` and to note their reasoning in a free-text field, which we referred to during expert review.

Despite a desire for higher levels of agreement, we made the decision to shift to single annotation. We reasoned that the dataset would be reviewed by experts and that the best use of our resources would be to annotate more articles and gain the most annotations to be reviewed. We anticipated greatly improved alignment when PDF conversion improved and were simultaneously concerned that our trained annotators would graduate, incurring more cost in training rather than producing more annotations.

Iterative refinement

Expert review and curation were conducted at the end of our student annotation phase as we worked towards gold-standard annotation, simultaneously with alignment improvement via GROBID PDF conversions. As we progressed we prototyped NER model training to validate the usefulness of the dataset.

Expert annotators reviewed all the annotations in single-annotated articles. In double-annotated articles, experts reviewed annotations from the annotator who found the highest number of software mentions. This review process exposed further annotation inconsistencies, reconciled iteratively through discussion creating rules implemented via automatic scripts and manual edits.

As anticipated, some inconsistencies came from conceptual ambiguity from examples not seen during training. We thus refined the scope of software entities to be annotated. Originally annotators were encouraged to record possible but unlikely software mentions and to categorize them as probable *hardware*, *algorithm*, or *database*, or general-purpose *web platform*. These occurred infrequently and yielded results less relevant to our goal of increasing software visibility in research. Thus we changed this step to binary classification: software vs. non-software. Only the `software` category is included in the final dataset, but others can be found in the original RDF annotations.

Other inconsistencies were syntactic, increasing unwanted variance in the dataset.

To resolve such inconsistencies, we removed prefixes of software version numbers like "ver.", "v.", or "version"; we also specified that the annotation of software `publisher` only includes names of organizations or individuals, excluding information about individual affiliation or organization geo-location that often follows. For example, for "GraphPad Software, San Diego, CA, USA", we only annotated "GraphPad Software". Finally, we included acronyms as part of the software mention to be annotated (e.g., "Statistical Package for Social Sciences (SPSS)" as a whole string). Reducing syntactic variance in this way helped the machine learning algorithm learn from the text patterns.

Other than consistency improvement, we also minimized false negatives using a script to exhaustively search for any occurrences of validated software mention strings across all the full-text paragraphs included in our corpus. Expert annotators reviewed these string matches and added annotations for those confirmed to be software mentions.

For annotations edited and added during expert review, `certainty` scores were dropped since they were judgments by the original annotator.

We also dropped manually annotated formal software in-text citations because GROBID provided automatically recognized and marked-up in-text citations linked to references in the bibliography. GROBID's annotations were also reviewed during expert review.

We adjusted the annotation categories to make them more consistent with actual annotations. During annotation we used `software_name`, `creator`, and both `version_number` and `version_date`. In the published data these became `software`, `publisher` and a combined category `version`.

Finally, we found that expert review for `software_was_used` was problematic, requiring reviews of greater article-level context. We therefore dropped `software_was_used` code, retaining it in provenance. We plan to return to this code for further review.

Both original `certainty` and manual annotations of software references linked to their in-text citations and categorizations remain accessible in our RDF provenance. All these changes implemented by expert reviewers are also recorded in Git log history and our GitHub repository issues.

Discussion

Interdisciplinary tensions in annotation practice

Corpus annotation is interdisciplinary. Initially, we followed content analysis methodology to develop the annotation scheme and train annotators. Later we added a machine learning specialist for dataset curation and corpus construction. Here we discuss two interdisciplinary tensions.

First, we differed on the valuation of text annotated as negative examples. Annotators read 4,971 full-text articles and found 3,743 articles that did not contain any software mentions at all. Thus we identified substantial negative examples through content analysis. However, for ML-based NLP, training data is expected to have balanced numbers of positive and negative examples, with minor imbalances carefully used to fine-tune for precision and recall. Thus a great deal of negative text annotated is not included in the final corpus, despite the time and resources devoted to annotating them.

A second tension arose over inter-annotator agreement: while debated, 0.7-0.8 is often thought acceptable in content analysis methodology (Landis & Koch, 1977; Neuendorf, 2016). We initially accepted approximately these levels (although obscured by alignment misses) when moving from double to single annotation. Our machine learning specialist expected much higher agreement levels of 0.95 or more. This disparity in expectation led to debates over the trade-off between additional percentages of agreement and larger numbers of annotated examples. Either choice would incur high cost; and their likely impact on machine learning performance was unclear to us. While an acceptable level of agreement attests to the reliability of the scheme application during annotation, it does not equal to the usefulness of the dataset for machine learning (Artstein & Poesio, 2008). For instance, agreement can be increased by simplifying annotation procedures and decisions; the trade-off is a less sophisticated system. Disagreement can reflect the complexity of the concepts; removing it from the dataset can create a false impression of a system's abilities or result in unreasonably high training costs. Guidance or evidence for shaping the right trajectory of dataset annotation is scant.

Uncertainty in provenance management

Provenance data is intended for dataset reuse (Pasquetto et al., 2019; Ribes, 2017; Simmhan et al., 2005). However, dataset reuse does not always happen (Chao, 2015), making the return on investment unclear, especially when the opportunity cost is a smaller overall dataset. The importance of data provenance has been acknowledged (Polyzotis et al., 2017; Ribes, 2017), but discussions of its practices and reporting are rare. Research in machine learning also increasingly addresses the importance of documenting the context and process of dataset creation (Bender & Friedman, 2018; Gebru et al., 2018; Holland et al., 2018) but guidance is still incomplete, perhaps unavoidably. Throughout the project we had difficulties in determining the cost-benefit of maintaining high-resolution provenance data. For example, a concrete question was: "To which granularity should we implement and present provenance?" Some recommend documenting the demographics of annotators (Bender & Friedman, 2018) as possibly affecting the bias and accountability of the dataset; however, should we discuss the immigration status of annotators, even after being made anonymous, as we did above? How can we balance uncertain future relevance against the possibility of breaking confidentiality and placing participants at risk? Additional guidance and discussion on what to be documented and disclosed as data provenance would be helpful.

Reflections on pipelines for scholarly document processing

Publications on literature mining rarely describe their full pipeline for scholarly document processing, yet our experience showed its importance.

We experienced substantial difficulties in working with annotations of text extracted from PDFs and available XML of academic publications. While ineffective PDF conversion was a barrier, the publication XML provided by publishers does not necessarily correspond to the PDF publications. These issues extend beyond the PMC XML to formats like JATS (Journal Article Tag Suite) that provide a more generalized encoding of journal publications than the actual PDF presentations. However, reading the actual PDFs is significantly easier for annotators, especially for article level context.

We found that the GROBID PDF conversion was substantially better than other

options we had tried, and that it enables the direct addition of manually generated annotations to texts extracted from PDFs. This capability obviates falling back to publication abstracts or XML for text mining efforts; thus the opportunity is opened up for exploiting conference proceedings, grey literature, and the full expanse of OA articles instead of the limited overlap between OA PDF and XML, or unreliable contract negotiations for XML access. GROBID also provides other opportunities such as annotating PDFs via PDF coordinates. We note that GROBID is also part of the underlying infrastructure for the recent widely-used CORD-19 dataset published by the Allen Instituted of AI (Kohlmeier et al., 2020).

Similar software NER efforts, like the RRID initiative, do share their NLP processing pipeline (Hsu et al., 2019). Sharing these adds validity to NER performance and findings; detailed and critical discussions of scholarly pipelines would be beneficial.

Conclusion and Future Work

We provide a dataset of annotated software mentions from full-text academic literature in biomedicine and economics directly converted from published PDFs with reproducible infrastructure. It is the largest gold-standard dataset of software mentions to date, includes provenance, and is formatted for immediately usefulness in NLP. These properties make the Softcite dataset useful for supervised learning at scale.

Our future work lies in two directions: (1) collaborative expansion of software mention datasets, and (2) application of a software knowledge base constructed with NER models trained on the dataset.

First, we are working towards a community project to extend the size and scope of similar datasets. The intended expansion involves the incorporation of existing annotated corpora across research domains, collaborative annotation of articles in other academic fields, and using the gold-standard to validate the bootstrapped datasets. We plan to undertake further review of our dropped `software_was_used` code as part of this work. Existing resources such as software items indexed by digital registries, including projects such as the WoS Data Citation Index, ASCL, swMATH, and the RRID system, have the potential to be utilized together with the Softcite dataset (including adding identifiers from

those projects to the Softcite dataset). Improving these connections would enable further generalization of the software NER models to other research fields such as natural sciences and mathematics, through cross-domain training/validation, and seeding article selection for expanding gold-standard annotations.

Second, we are deploying machine learning at scale trained on the Softcite dataset to construct a knowledge base, processing tens of millions of open access PDFs. We intend to make the knowledge base available for scientometric researchers to examine topics such as diffusion and interdisciplinary influence. It will also be useful to develop systems for research software developers and end-users, including metric and crediting system, citation recommendation system, and information retrieval of research software. Particularly, we plan to deploy the database in an existing system ("CiteAs.org", n.d.), showing research software developers how their work is mentioned in academic publications and motivating them to make explicit requests for their preferred citations.

Named entity recognition of software in academic publications requires substantial cost and effort, but holds the potential to improve visibility of key infrastructural work and generate insights that, when carefully used, can advance scholarship.

Acknowledgement

We appreciate anonymous reviewers and journal editors. The research has been supported by the Alfred P. Sloan Foundation. The authors also thank our anonymous student annotators for dedicating their efforts to the gold-standard annotation.

Notes

[1]https://www.force11.org/group/software-citation-working-group

[2]https://www.force11.org/group/software-citation-implementation-working-group

[3]https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/

[4]https://tei-c.org/release/doc/tei-p5-doc/en/html/ref-rs.html

[5]https://howisonlab.github.io/softcite-dataset/coding-scheme.html

[6]https://ropensci.org/

References

Allen, A., & Schmidt, J. (2014). Looking before leaping: Creating a software registry. *arXiv preprint arXiv:1407.5378*.

Allen, A., Teuben, P. J., & Ryan, P. W. (2018). Schroedinger's code: A preliminary study on research source code availability and link persistence in astrophysics. *The Astrophysical Journal Supplement Series*, *236*(1), 10.

Artstein, R., & Poesio, M. (2008). Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, *34*(4), 555–596. https://doi.org/10.1162/coli.07-034-R2

Atom. (n.d.). Retrieved June 12, 2020, from https://atom.io/

Bandrowski, A. E., & Martone, M. E. (2016). Rrids: A simple step toward improving reproducibility through rigor and transparency of experimental methods. *Neuron*, *90*(3), 434–436.

Bandrowski, A., Brush, M., Grethe, J. S., Haendel, M. A., Kennedy, D. N., Hill, S., Hof, P. R., Martone, M. E., Pols, M., Tan, S. S., Et al. (2016). The resource identification initiative: A cultural shift in publishing. *Neuroinformatics*, *14*(2), 169–182.

Bender, E. M., & Friedman, B. (2018). Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, *6*, 587–604.

Chamberlain, S. (n.d.). Ropensci: The fulltext package. Retrieved June 16, 2020, from https://docs.ropensci.org/fulltext/

Chao, T. (2015). Mapping Methods Metadata for Research Data. *International Journal of Digital Curation*, *10*, 82–94. https://doi.org/10.2218/ijdc.v10i1.347

Chue Hong, N. P., Allen, A., Gonzalez-Beltran, A., de Waard, A., Smith, A. M., Robinson, C., Jones, C., Bouquin, D., Katz, D. S., Kennedy, D., Ryder, G., Hausman, J., Hwang, L., Jones, M. B., Harrison, M., Crosas, M., Wu, M., Löwe, P., Haines, R., ... Pollard, T. (2019, October 15). *Software citation checklist for authors*. Zenodo. https://doi.org/10.5281/zenodo.3479199

Chue Hong, N. P., Allen, A., Gonzalez-Beltran, de Waard, A., Smith, A. M., Robinson, C., Jones, C., Bouquin, D., Katz, D. S., Kennedy, D., Ryder, G., Hausman, J.,

Hwang, L., Jones, M. B., Harrison, M., Crosas, M., Wu, M., Löwe, P., Haines, R., ... Pollard, T. (2019, October 15). *Software citation checklist for developers*. Zenodo. https://doi.org/10.5281/zenodo.3482769

Citeas.org. (n.d.). Retrieved July 4, 2020, from http://citeas.org/

Conway, J. R., Lex, A., & Gehlenborg, N. (2017). Upsetr: An r package for the visualization of intersecting sets and their properties. *Bioinformatics*, *33*(18), 2938–2940.

Di Cosmo, R. (2020). Archiving and referencing source code with software heritage. *arXiv preprint arXiv:2004.00514*.

Di Cosmo, R., Gruenpeter, M., Marmol, B., Monteil, A., Romary, L., & Sadowska, J. (2019). Curated archiving of research software artifacts: Lessons learned from the french open archive (hal).

Di Cosmo, R., Gruenpeter, M., & Zacchiroli, S. (2019). Referencing source code artifacts: A separate concern in software citation. *Computing in Science & Engineering*, *22*(2), 33–43.

Document engineering | science-miner. (n.d.). Retrieved October 25, 2020, from https://science-miner.com/document-engineering/

Duck, G., Kovacevic, A., Robertson, D. L., Stevens, R., & Nenadic, G. (2015). Ambiguity and variability of database and software names in bioinformatics. *Journal of biomedical semantics*, *6*(1), 29.

Duck, G., Nenadic, G., Filannino, M., Brass, A., Robertson, D. L., & Stevens, R. (2016). A survey of bioinformatics database and software usage through mining the literature. *PloS one*, *11*(6).

Edwards, P. N., Jackson, S. J., Chalmers, M. K., Bowker, G. C., Borgman, C. L., Ribes, D., Burton, M., & Calvet, S. (2013). *Knowledge Infrastructures: Intellectual Frameworks and Research Challenges*. Retrieved June 7, 2020, from http://pne.people.si.umich.edu/PDF/Edwards_etal_2013_Knowledge_Infrastructures.pdf

Eric, A. (2016). Classification of scientific journals. version 1.06. Science-Metrix. Retrieved November 19, 2014, from https://science-metrix.com/?q=en/classification

Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., III, H. D., & Crawford, K. (2018). Datasheets for datasets.

Goffman, E. (1959). *The Presentation of Self in Everyday Life*. Doubleday.

Greuel, G.-M., & Sperber, W. (2014). Swmath–an information service for mathematical software, In *International congress on mathematical software*. Springer.

Grobid. (2008–2020). GitHub. https://github.com/kermitt2/grobid

Halevy, A., Norvig, P., & Pereira, F. (2009). The Unreasonable Effectiveness of Data. *IEEE Intelligent Systems*, *24*(2), 8–12. https://doi.org/10.1109/MIS.2009.36

Holland, S., Hosny, A., Newman, S., Joseph, J., & Chmielinski, K. (2018). The dataset nutrition label: A framework to drive higher data quality standards. *arXiv preprint arXiv:1805.03677*.

Howison, J., & Bullard, J. (2016). Software in the Scientific Literature: Problems with Seeing, Finding, and Using Software Mentioned in the Biology Literature. *Journal of the Association for Information Science and Technology*, *67*(9), 2137–2155. https://doi.org/10.1002/asi.23538

Howison, J., Deelman, E., McLennan, M. J., Ferreira da Silva, R., & Herbsleb, J. D. (2015). Understanding the Scientific Software Ecosystem and its Impact: Current and Future Measures. *Research Evaluation*, *24*(4), 454–470. https://doi.org/10.1093/reseval/rvv014

Howison, J., & Herbsleb, J. D. (2011). Scientific Software Production: Incentives and Collaboration, In *Proceedings of the ACM 2011 conference on Computer supported cooperative work*, Hangzhou, China, Association for Computing Machinery. https://doi.org/10.1145/1958824.1958904

Hsu, C.-N., Bandrowski, A. E., Gillespie, T. H., Udell, J., Lin, K.-W., Ozyurt, I. B., Grethe, J. S., & Martone, M. E. (2019). Comparing the use of research resource identifiers and natural language processing for citation of databases, software, and other digital artifacts. *Computing in Science & Engineering*, *22*(2), 22–32.

Hwang, L. J., Pauloo, R. A., & Carlen, J. (2019). Assessing the impact of outreach through software citation for community software in geodynamics. *Computing in Science & Engineering*, *22*(1), 16–25.

Java. (n.d.). Retrieved July 4, 2020, from https://www.java.com/en/

Jo, E. S., & Gebru, T. (2020). Lessons from archives: Strategies for collecting sociocultural data in machine learning, In *Proceedings of the 2020 conference on fairness, accountability, and transparency*.

Kohlmeier, S., Lo, K., Wang, L. L., & Yang, J. (2020, March 16). COVID-19 open research dataset (CORD-19) [type: dataset]. Zenodo. https://doi.org/10.5281/zenodo.3813567

Krüger, F., & Schindler, D. (2020). A Literature Review on Methods for the Extraction of Usage Statements of Software and Data. *Computing in Science Engineering*, *22*(1), 26–38. https://doi.org/10.1109/MCSE.2019.2943847

Lafferty, J., McCallum, A., & Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Landis, J. R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *biometrics*, 159–174.

Li, K., & Yan, E. (2018). Co-mention Network of R Packages: Scientific Impact and Clustering Structure. *Journal of Informetrics*, *12*(1), 87–100. https://doi.org/10.1016/j.joi.2017.12.001

Mayernik, M. S., Hart, D. L., Maull, K. E., & Weber, N. M. (2017). Assessing and Tracing the Outcomes and Impact of Research Infrastructures. *Journal of the Association for Information Science and Technology*, *68*(6), 1341–1359. https://doi.org/10.1002/asi.23721

McLennan, M., & Kennell, R. (2010). HUBzero: A Platform for Dissemination and Collaboration in Computational Science and Engineering. *Computing in Science Engineering*, *12*(2), 48–53. https://doi.org/10.1109/MCSE.2010.41

Mehta, H., Bradley, A., Hancock, M., & Collins, C. (2017). Metatation: Annotation as Implicit Interaction to Bridge Close and Distant Reading. *ACM Transactions on Computer-Human Interaction*, *24*(5), 1–41. https://doi.org/10.1145/3131609

Meyer, C. M., Mieskes, M., Stab, C., & Gurevych, I. (2014). DKPro Agreement: An Open-Source Java Library for Measuring Inter-Rater Agreement, In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: System Demonstrations*, Dublin, Ireland, Dublin City University; Association for

Computational Linguistics. Retrieved June 12, 2020, from https://www.aclweb.org/anthology/C14-2023

MySQL. (n.d.). Retrieved June 12, 2020, from https://www.mysql.com/

Nangia, U., & Katz, D. S. (2017). Understanding software in research: Initial results from examining nature and a call for collaboration, In *2017 ieee 13th international conference on e-science (e-science)*. IEEE.

Neuendorf, K. A. (2016). *The Content Analysis Guidebook*. SAGE.

Ozyurt, I. B., Grethe, J. S., Martone, M. E., & Bandrowski, A. E. (2016). Resource disambiguator for the web: Extracting biomedical resources and their citations from the scientific literature. *PLoS One*, *11*(1), e0146300.

Pagès, H., Aboyoun, P., Gentleman, R., & DebRoy, S. (2017). Biostrings: Efficient Manipulation of Biological Strings. Bioconductor version: Release (3.6). https://doi.org/10.18129/B9.bioc.Biostrings

Pan, X., Yan, E., & Hua, W. (2016). Disciplinary differences of software use and impact in scientific literature. *Scientometrics*, *109*(3), 1593–1610.

Pan, X., Yan, E., Wang, Q., & Hua, W. (2015). Assessing the Impact of Software on Science: A Bootstrapped Learning of Software Entities in Full-text Papers. *Journal of Informetrics*, *9*(4), 860–871. https://doi.org/10.1016/j.joi.2015.07.012

Park, H., & Wolfram, D. (2019). Research software citation in the data citation index: Current practices and implications for research software sharing and reuse. *Journal of Informetrics*, *13*(2), 574–582.

Pasquetto, I. V., Borgman, C. L., & Wofford, M. F. (2019). Uses and reuses of scientific data: The data creators' advantage. *Harvard Data Science Review*, *1*(2). https://doi.org/10.1162/99608f92.fc14bf2d

Pdftotext: Use pdftotext to get text from a pdf in sckott/extractr: Extract Text from 'PDFs'. (n.d.). Retrieved June 12, 2020, from https://rdrr.io/github/sckott/extractr/man/pdftotext.html

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations, In *Proceedings of naacl-hlt*.

Piwowar, H., Priem, J., Larivière, V., Alperin, J. P., Matthias, L., Norlander, B., Farley, A., West, J., & Haustein, S. (2018). The state of oa: A large-scale analysis of the prevalence and impact of open access articles. *PeerJ*, *6*, e4375.

Polyzotis, N., Roy, S., Whang, S. E., & Zinkevich, M. (2017). Data management challenges in production machine learning, In *Proceedings of the 2017 acm international conference on management of data*.

Poppler. (n.d.). Retrieved June 30, 2020, from https://poppler.freedesktop.org/

Python. (n.d.). Retrieved June 12, 2020, from https://www.python.org/

Raymond, E. S. (1998). The Cathedral and the Bazaar. Retrieved June 11, 2020, from https://firstmonday.org/ojs/index.php/fm/article/download/578/499?inline=1

Remote-ftp. (n.d.). Retrieved June 12, 2020, from https://atom.io/packages/remote-ftp

Ribes, D. (2017). Notes on the Concept of Data Interoperability: Cases from an Ecology of AIDS Research Infrastructures, In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, Portland, Oregon, USA, Association for Computing Machinery. https://doi.org/10.1145/2998181.2998344

Russell, P. H., Johnson, R. L., Ananthan, S., Harnke, B., & Carlson, N. E. (2018). A large-scale analysis of bioinformatics code on github. *PLoS One*, *13*(10), e0205898.

Schindler, D., Zapilko, B., & Krüger, F. (2020). Investigating software usage in the social sciences: A knowledge graph approach, In *European semantic web conference*. Springer.

Selbst, A. D., Boyd, D., Friedler, S. A., Venkatasubramanian, S., & Vertesi, J. (2019). Fairness and abstraction in sociotechnical systems, In *Proceedings of the conference on fairness, accountability, and transparency*.

Simmhan, Y. L., Plale, B., & Gannon, D. (2005). A Survey of Data Provenance in e-Science. *ACM SIGMOD Record*, *34*(3), 31–36. https://doi.org/10.1145/1084805.1084812

Smith, A. M., Katz, D. S., & Niemeyer, K. E. (2016). Software citation principles. *PeerJ Computer Science*, *2*, e86.

Star, S. L., & Ruhleder, K. (1996). Steps Toward an Ecology of Infrastructure: Design and Access for Large Information Spaces. *Information Systems Research*. https://doi.org/10.1287/isre.7.1.111

TEI Consortium. (2020). TEI P5: Guidelines for Electronic Text Encoding and Interchange. https://doi.org/10.5281/ZENODO.3413524

The R Project for Statistical Computing. (n.d.). Retrieved June 12, 2020, from https://www.r-project.org/

Thomer, A. K., Wickett, K. M., Baker, K. S., Fouke, B. W., & Palmer, C. L. (2018). Documenting provenance in noncomputational workflows: Research process models based on geobiology fieldwork in yellowstone national park. *Journal of the Association for Information Science and Technology*, *69*(10), 1234–1245.

Travis ci. (n.d.). Retrieved June 12, 2020, from https://travis-ci.org/

Unpaywall. (n.d.). Retrieved July 3, 2020, from https://unpaywall.org/

Wei, Q., Zhang, Y., Amith, M., Lin, R., Lapeyrolerie, J., Tao, C., & Xu, H. (2020). Recognizing software names in biomedical literature using machine learning. *Health Informatics Journal*, *26*(1), 21–33.

Wickham, H. (2016). *Ggplot2: Elegant graphics for data analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., ... Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. https://doi.org/10.21105/joss.01686