SoftCite - Interim Year 1 Report for Sloan Foundation

James Howison, Jason Priem, and Heather Piwowar

October 1st, 2017

SoftCite goals and activities

Our grant proposal lays out the goals of the SoftCite project:

> Our goal is to improve software in scholarship (science, engineering, and the humanities) by raising the visibility of software work as a contribution in the literature, thus improving incentives for software work in scholarship.

Our major activites are:

1. Develop a "Gold Standard" manually coded dataset of software mentions in the literature
2. Use that standard toward machine learning discovery of software mentions
3. Develop a database linking software tools to associated citations
4. Using this database, prototype and study three tools: CiteAs, CiteSuggest, and Software ImpactStory

In this report we describe progress in each of these activities.

Progress on "Gold Standard" dataset development

Since coding began in Spring 2017 the team at UT Austin has coded 151 articles from the PubMed Open Access Collection, finding 523 mentions of software (and many, many more examples of sentences that do not mention software, just as important to the machine learning). The coders have found 74 articles with mentions of software and 82 articles without software mentions and thus far have identified 187 separate software packages. En route we trained a doctoral student (Hannah Cohoon) in content analysis, revised our coding scheme to best match the future machine learning task, established a collaborative workflow hosted on github (publicly available repository, documentation). We have on-boarded and trained ~35 undergraduate and masters students.

We are growing the student group: from ~4 in Spring 2017, ~8 over Summer 2017, to ~25 in Fall 2017 (the fall group has only just finished training and has not yet contibuted coding of new articles). In addition we have reached out to the minority-serving institution Houston Tillotson University and will on-board a group of ~5 coders from there in the next two weeks. Currently each article is being coded by two students, but as agreement between new coders improves, only a small percentage will be double coded.

Our speed of coding is accelerating; we expect to have ~400 PubMed articles coded by the end of this semester. We will then move onto collections from astronomy and economics. Nonetheless, we are also exploring two techniques to move faster. Both techniques will require validation from our current completely manually coded dataset and so must follow the work we're currently doing.

The first is bootstraping using a "whitelist" of software packages, identifying sentences with those words in them. We would then manually code just those sentences. Assuming that packages known beforehand (and therefore on the white list) are not mentioned in systematically different ways bootstraping can train a system to learn generally how packages are mentioned and apply that learning to find mentions of packages not on the whitelist. Colleagues have taken this approach (Pan et al. 2015) but they have not had the fully manually coded set to validate their approach; our manually coded full dataset will provide validation.

The second approach takes advantage of one finding so far: software mentions cluster together in articles. However they do not cluster predictably (e.g., one paper might have multiple clusters and clusters are not neccessarily only in methods sections). We will experiment using a whitelist to locate mentions of known packages and then expand up and down 2 or 3 paragraphs to increase our chance of finding mentions of

packages not on the whitelist; this ought to amerliorate bias resulting from the choice of the whitelist and allow us to more reliably characterize mentions per article en rouate to a full machine learning system.

Progress towards machine learning discovery of software mentions

The machine learning to automate recognition of software mentions builds on the dataset discussed above; only when it is large enough will we be able to begin training. However, we have worked to ensure that the results of manual coding are most useful for machine learning training. Specifically we have experimented with approaches to converting the pdf articles to text in a way that we can label the sentences identified by the coders. As expected this is a difficult process (issues include headers and column conversion problems). We have two approaches to this. First we chose to work with the PubMed Open Access dataset which has clean XML encoded versions of each article, that decision will enable us to get started training without solving pdf conversion issues. Second Heather and Jason at ImpactStory have been exploring new generation pdf conversion tools, such as CERMINE (CERMINE Project 2017; Tkaczyk et al. 2015), which identify elements such as headers, columns, tables etc. prior to text conversion.

Progress towards CiteAs

The CiteAs tool, available at citeas.org, allows users to map from software to ways to cite that software. The backend system provides an API (in preparation for future integration into other tools, both ours and others) integrated with the database described above. The web frontend uses the API to provide the current application. The open repository is available at https://github.com/Impactstory/citeas-api and https://github.com/Impactstory/citeas-webapp. Users can provide the text name of a software package "yt", a URL such as a project landing page "yt-project.org" or "`https://cran.r-project.org/web/packages/stringr`" or a DOI (publication or software), such as "10.5281/zenodo.160400". The system then checks the database and if no entry is found kicks off a procedure to locate an appropriate citation. Potential sources are checked in sequence and those results returned to users with a provenance chain which helps users known where the suggestions come from and will help software projects adjust how they request citations.

As an example Figure 1 shows a screenshot showing the results of querying "yt-project". In the background the system has downloaded the page and parsed it for a citation request. In the case of `yt` the citation request on the yt-project homepage links to a bibtex file which was then downloaded and parsed to provide the citation.

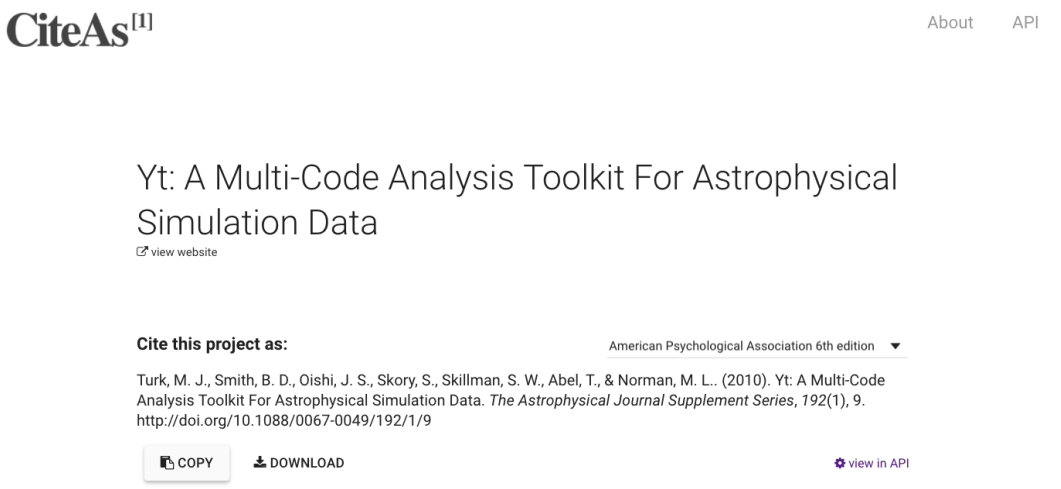

Figure 1: CiteAs output for http://yt-project.org

Figure 2 shows an example of a user seeking a citation for an R package. The system downloads the URL `https://cran.r-project.org/web/packages/stringr` and identifies a link to a repository (here the repository is github). We then seek a file in the repository that makes a citation request, checking README and the DESCRIPTION file on CRAN (in the future we will also check for a CITATION or CITATION.TXT in the repository). The citation below is built from the DESCRIPTION file.

Figure 2: CiteAs output for stringr

The provenance chain is illustrated in the API output in Figure 3, which shows the locations and techniques that CiteAs attempted in finding a citation. In this case the system identified DOIs located in the README file; the DOIs relate to both an article and to a zenodo archived version of the software. These DOIs are each are mapped to a citation and a prefered citation displayed to the user.


```
"provenance": [
  {
    "context": "GitHub URL",
    "location": "http://api.citeas.org/product/https://github.com/CeON/CERMINE",
    "source_type": "request parameters"
  },
  {
    "context": "GitHub README.md file",
    "location": "https://github.com/CeON/CERMINE/raw/master/README.md",
    "source_type": "DOI found"
  },
  {
    "context": "DOI API",
    "location": "http://doi.org/10.5281/zenodo.569829",
    "source_type": "DOI metadata"
  }
],
"url": "https://github.com/CeON/CERMINE"
```


Figure 3: CiteAs API data showing origns of CERMINE citation

We are exposing these citations alternatives in the interface, shown below in Figure 4, in reference the the stringr package.

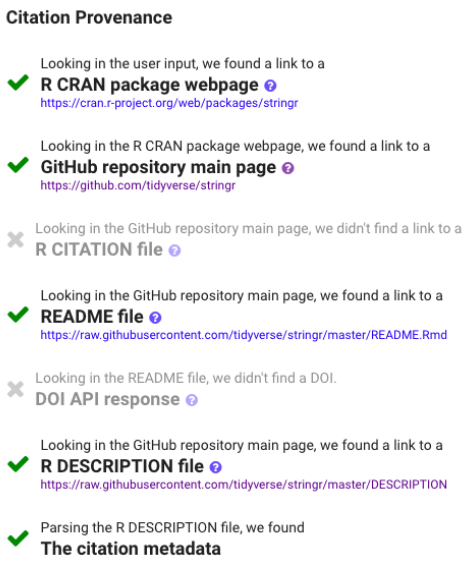

Figure 4: CiteAs interface showing provenance chain

Progress towards database linking software tools to associated citations

The backend for CiteAs consults what will become the database linking tools to associated citations. Thus far the database is built from the techniques that CiteAs uses to locate citation requests.

Over time we will expand this database in two ways. First we will run our citation lcoation techniques pre-emptively using catalogues of scientific software (e.g., libraries.io, NSF and NIH listings, and field specific listings such as the ASCL and SBGrid). Second, we will add tool-to-citation links from the literature including initially those identified through our manual coding, and eventually those identified as our machine learning improves. As links are added to the database they will become available to CiteAs and our two other tools (CiteSuggest and Software ImpactStory).

Our collaboration is working well

We are working well as a group. We have scheduled calls each second week to report progress and provide input to each other's activities. Between these calls we have regular email contact and watch each others software repositories. Jason and Heather remotely attended one of James and Hannah's student coding sessions to motivate the students by explaining the end-use of the dataset and demonstrating the CiteAs tool. We distributed CiteAs stickers to the students.

We may seek to arrange face-to-face working sessions, co-located with conferences we would naturally attend (such as the 2018 RDA or FORCE11 Scholarly Communications conference) but we do not see the need to spend funds on separate face to face meetings.

We hope to expand our collaboration beyond our funded group. As mentioned above we are collecting software mention coding from others and working to make it compatible with our schema to publish them together. We are also in communication with libraries.io which will form a natural extension, allowing the joining of their dependency information with our citation information, per software package.

Plans for next year

Our priorities for work in the immediate future are:

1. Expand and accelerate our gold-standard dataset production at UT Austin.
2. Soft-launch CiteAs.org and monitor feedback and iterate the tool
3. Prototype machine learning using the gold-standard dataset
4. Drawing on the database and API created for CiteAs and machine learning results as they become available we will develop CiteSuggest, then develop Software ImpactStory.
5. James at UT Austin will recruit an additional doctoral student to undertake socio-technical studies of the three tools.

While the grant was for 2 years, we have adjusted our budgeting to anticipate completing the work over three years, so that we can gather sufficient coded articles and implement the machine learning. ImpactStory plans to implement initial versions of the prototypes and work on the machine learning but then slow their pace of work until a large enough "gold standard" collection has been completed.

References

CERMINE Project. 2017. "CERMINE: Content ExtRactor and MINEr." Centre for Open Science. https://github.com/CeON/CERMINE.

Pan, Xuelian, Erjia Yan, Qianqian Wang, and Weina Hua. 2015. "Assessing the Impact of Software on Science: A Bootstrapped Learning of Software Entities in Full-Text Papers." *Journal of Informetrics* 9 (4): 860–71. doi:10.1016/j.joi.2015.07.012.

Tkaczyk, Dominika, Paweł Szostek, Mateusz Fedoryszak, Piotr Jan Dendek, and Łukasz Bolikowski. 2015. "CERMINE: Automatic Extraction of Structured Metadata from Scientific Literature." *International Journal on Document Analysis and Recognition (IJDAR)* 18 (4): 317–35. doi:10.1007/s10032-015-0249-8.