Maggie Connell, Kathryn Mechura, Mason Stilwell
Professor Richard Brown
CS390 - Capstone
16 December, 2016

Tracing Retractions in Scientific Literature

Abstract

In this paper we attempt to address the growing need for a mechanism to track the impact of retracted papers in the body of scientific literature. The number of retracted articles has increased significantly over the past several years making the need for such a mechanism or tool increasingly necessary. In order to meet this need we build a web of scientific literature, linking articles together through their citations. Through their citations we trace retracted articles through the web to determine their impact on individual articles. In order to do so we wrote scripts to mass download articles from the PubMed Open Access Subset database as xml files. Next we parse these files to extract identifying information and citations and use that parsed information to enter them into a database that keeps track of the articles and their citations. Finally, we apply an algorithm to assess the impact of retractions on individual articles. This algorithm traverses the article's citations three levels down to see if and retracted articles were cited. While this data does not provide a reliable measure of the article's reliability, it is incredibly useful in determining the impact of retracted articles and empowering researches with the knowledge of how retracted articles may affect other articles. Ultimately due to time constraints we we were unable to build a large enough dataset to draw any significant conclusions on the extent of the impact of retracted articles on the PubMed database. However we produced a functioning tool and methodology to build a database of scientific literature, which can then be used to analyze the impact of retracted articles on other articles in the database.

Introduction

The main goal of this capstone project is to address the research question: “What is the impact of retracted scientific papers on other scientific papers and the scientific community?” To investigate this, we created a database of scientific or academic articles and implemented a directed graph data structure within a SQL database that links them through their citations. The results of an analysis of that database indicates the impact of retracted papers on the body of scientific and academic literature. This agenda required several steps: downloading the documents in XML format from PubMed’s online open source database; parsing and researching the documents to extract meta information including the title, id numbers, and the document’s

retraction status; storing that information in an SQL database; then designing and implementing an algorithm to determine the impact of retracted articles on a given article or set of articles.

Motivations

The number of papers that are retracted has climbed dramatically within the last 10 years (Steen 2013). What's more concerning, these papers are still being cited as if they weren't retracted (Steen 2010). This may be in a large part due to inconsistencies in flagging articles as retracted, where some articles may not even be properly flagged (Wright 2011). One study found that 32% of withdrawn articles are not labeled as retracted (Steen 2010). This can become a serious issue when articles that have been retracted for error or fraud advise some unattainable result. Many retracted papers can have serious consequences, especially in medicine. The case of Dr. Reuben is a clear example of this, where his 21 papers that were retracted due to fabrication may have lead to under medication of patients for postsurgical pain (Neal 2009). In an analysis of 788 retracted papers, one study found that of this subset 9,189 patients received treatment that was based in these studies (Steen 2010).

A recent study found that 65% of journals within their sample had a retraction policy. This represents an over threefold increase from a similar study done in 2004, where in a similar sampling of journals 21% had a retraction policy (Atlas 2004). These results indicate that editors have at least begun to realize the importance in noting and publishing retractions. However, as previously noted many articles that have been labeled as retracted are still cited. The potential impact of these citations is massive, as the information from these articles are shared and permeate through the scientific community. There have been no concrete studies that look at how much retracted information may be inherent in the average scientific article, hence the desire for this tool. We also hope to create a tool that makes it easy to determine if an article is actually retracted, or if it directly uses results from a retracted article. This comes from the inconsistencies in flagging retracted articles across different databases, and the seeming confusion we see in scientists who continue to cite retracted literature.

Our tool looks to streamline this process by allowing the user to pick a particular article to analyse, or even just have them import their list of citations. For each article chosen, the tool will indicate first if the article itself has been retracted. It will also tell if any articles that it has cited are retracted. From there, it will see if any of those articles cite a retracted article, and list those as secondary level retractions for the paper being analyzed. It will do this once more, looking at the third level retractions. While none of these results should necessarily invalidate the results of said article, we hope it will allow scientists to more critically analyse the information they use. Additionally, we plan to run some analysis on a large subset of articles contained within the PMC database. We will gather how many cite retracted articles, and use this as a starting point in saying how prevalent information that comes from retracted sources is within the scientific literature. This analysis will also extend to looking at retracted articles, and seeing how many times they are cited by other articles and how that information might flow.

Literature Review

Assessing the impact of a retracted paper on other scientific papers and on the scientific and academic communities produces several challenges. A researcher must first understand various aspects of retraction, such as why papers are retracted and why papers and articles continue to cite known retracted papers. That researcher must also design a system to judge the influence of retracted papers while accommodating the various aspects of retractions.

According to Steen (2010), the top reasons for retractions of scientific or academic papers and articles fall into two categories: error and fraud. Included in the error category are:

- Plagiarism: the theft of text from an uncredited author
- Self-plagiarism: using substantially the same data or words in different publications without citation
- Scientific mistake: a mistake that invalidates research, typically explained carefully in a retraction notice
- Ethical issues: violations of accepted publication practice
- Journal error: accidental duplicative publication by a journal through no fault of the authors
- Unstated: retraction notice is non-informative, simply stating that retraction has happened

Under fraud:

- Data fabrication: theft of data from an uncredited author or generation of completely artificial data
- Data falsification: editing or manipulation of authentic data to "prove" a hypothesis

This information can be used to create a deeper understanding of retractions and can be factored into a scoring metric during the analysis of this project's data.

Often, retracted papers are still positively cited, or cited as a means of bolstering one's argument, due to the way retraction information is disseminated (Gabehart, 2005). Before electronic publication, and even now that electronic publication is widespread, retraction information was given in the following edition of the journal issuing the retraction. Because a researcher or scientist may not check each journal for retraction information, they would be unaware that one of their sources was retracted (Gabehart, 2005).

In 2010, Steen published a paper on the analysis of 742 retracted papers that were available on PubMed. His analysis focused on the reason for retraction that was given by the author or the publication journal to investigate whether or not instances of fraud in scientific papers was rising. The analysis revealed two possible causes for increased reports of fraud: incidents of fraud in scientific papers was truly increasing or journals were strengthening efforts to discover and report fraud in their published papers. Steen's analysis can help inform our analysis and provide some background on cases of fraud and methods of analyzing retracted papers.

Several different paper ranking systems and algorithms already exist. One source lists and compares different paper ranking algorithms, including PageRank, CiteRank, and NewRank, which use citation analysis and citation graphs to compute scores for the influence or similarity of academic papers (Dunaiski, M. & Visser, W. 2012). This research provides a ranking system, and can also serve as a base for creating a more customized system. Also included in this paper is a method of downloading and parsing the citations and other meta information from documents and putting it into a database. Their process is very similar to the one that was implemented for this project, although theirs uses a queue structure which was not feasible for the number of documents analyzed in this project.

Recommendations

The past work done on analysis of retracted papers shows that not only is the fact that a paper was retracted is significant but the given reason for retraction can influence perceptions and the impact of that retracted paper. Classification of the retracted papers into "Error" and "Fraud" can provide significant insight into scientific retractions and would be important to be included in any analysis, should the information be available. Some of the past work can be built off of to create a tool that fits our vision for this project, but modifications need to be made such as how the meta information is stored and methods of displaying analysis results.

Methodology

PubMed Downloading

We chose to create a python script to download xml documents from the PubMed database. The script works by first grabbing all PubMed DOIs for a particular range, which here we set to be all open access marked in the PMC subset of PubMed. It then sorts these DOIs by year, and creates an empty xml for every year. This is done to keep the xml files smaller and a little more manageable, but these parameters can be changed to be smaller or larger. The program will then go in chronological order through the DOIs, by generating the associated PubMed url based on that DOI and then downloading the meta information.

Processing these files involves populating fields of newly created objects with the metadata of the article such as authors, journal type, citations, etc… These objects then are appended to the new xml file of the associated year, which is used to keep formatting consistent. The finder itself will input a DOI, which it can then append to the http address to grab specific articles. This tool can also be used to download individual articles by giving it a DOI input, although we did not use this feature in our implementation.

XML Parsing

In order to extract relevant information from the academic articles we analyzed, we had to parse the xml documents that they are contained in. In order do so we use tinyXML, an

external C++ xml parsing library. TinyXML converts an xml file into a document object model (DOM) tree composed of TiXmlNodes and TiXmlElements, which represent the xml tags from the document. TinyXML provides functions for traversing this xml tree. The lightweight nature of the library meant having to define our own functions to find tags that match a particular query string. The simplicity of tinyXML enabled us to learn and use it very quickly, which was essential given the time frame of our project. We wrote two separate functions to extract information from our documents' DOM trees, based on simple in-order tree traversal. The first function finds the first instance of a tag matching a search string parameter, while the second produces a vector of all occurrences of a search string. The purpose of having these two separate functions is to created shorter search time when we know that we are searching for a tag that only occurs once or if we only need the first occurrence of the tag. However, the second function enables us to compile information such as obtaining every citation or counting the number of times an article gets cited. This method of xml parsing enables us to extract and compile all of the necessary information to create database entries for the articles.

As we parse the xml documents we generate SQL commands to input all relevant information from the articles into our database. In order to do so we output commands into a file named citations.sql. We generate these commands using a standard template for commands to insert the articles and fill in the column details with information pulled from the xml document. This automatically generated .sql file is then run in Postgres to insert the documents into the SQL database.

Due to the large volume of articles in each xml document that we parsed this code takes a relatively long time to run. For a 700MB xml file it runs for about an hour and a half. This is because the parsing program runs in linear time proportional to the number of nodes in the xml document. So for very large documents this can be very time consuming. However, this could be vastly improved on by running the program in parallel. This would be fairly simple to accomplish, however due to time constraints on our project we were unable to add this feature. Running the .sql file to enter the articles is even more time consuming. To enter just over 250 thousand articles and document over 10 million citations (before deleting duplicates) took over six days to complete. The time necessary to execute a Select command in SQL increases as the table grows. Strategies exist for increasing the efficiency and speed of insert statements, however again due to time constraints we were unable to implement these strategies. Additionally, extra care could be taken to avoid inserting duplicate rows in the first place. Finally, we run into a memory issue when trying to run this code with some of our much larger files, especially those larger than 10GB. There is not enough space in memory to allocate for the computations on such a large file as this program asks for. This issue could be solved by using a machine with more RAM available. However, we solved this issue by breaking the xml files into smaller documents to be parsed individually instead of all together.

Database Design

We chose to store all relevant data about the xml documents in an SQL database to enable great speed up in lookup time and lower storage requirements in comparison to storing full xml documents. In particular we stored all relevant information on a PostgreSQL server. When creating database entries it was essential that we have a unique entry for each article so that we could accurately link articles together through their citations. This meant finding some unique identifier with which we could identify each document. Unfortunately, we found that the identifying numbers PubMed provided to each article were inconsistent. The articles we downloaded and the articles that they cited contained PMC and PMID identifying numbers. However, not every article contained either or both of these identifiers. Therefore, we could not use either of PubMed's built in identifiers as a primary key in our database. Instead, we generated a unique identifying key for each document entered into the database. We created a table called "Documents" containing entries for every parsed or cited document. Each entry contained a unique key, the article's PMC identifier, it's PMID identifier, it's title, a Boolean representing if the article is retracted, and a text array for any relevant notes. However, without using an element from the article itself as the primary key in our SQL table we risked having duplicate entries for each article. In order to avoid duplicates we created a view called "Unique_Documents" that selected only the minimum unique key and merged the remaining columns for any entries that have the same PMC identifier, PMID identifier, and title, if those fields are present. Unique_Documents does not have a retracted field because if the first entry of the article was not marked as retracted that tag gets lost when the entries are combined. So, to determine if an article is retracted, it must be searched for in the Documents table. To account for formatting differences, we removed spaces and punctuation from and lowercased all characters in the title strings.

We also created a table called "Cites" to represent the relationship between articles and the articles that they cite. This table contains two columns. The first column holds the unique key of the article that cites other articles. The second column holds the unique key of the article cited by the article whose key is in the first column. To fill this table we selected the article's unique identifying key from the Unique_Documents view by querying the view using the articles PMC identifier, PMID identifier, and/or title. Ultimately, this table holds the relationship between an article and every article that it cites. These entries can be thought of as the edges of a graph structure where each node represents an article and each edge a citation. Cites has the possibility of holding duplicate entries. Therefore, after a set in insert commands gets run in the database we search the table for identical rows and delete all except one.

Documents					
ID	PMID	PMC	Title	Retracted	Notes

cites			
DID	CitedDID	Notes	Count

Unique_Documents				
DID	PMID	PMC	Title	Notes

Figure 1: A visual representation of the database design.

Analysis

Methods

Our original approach called for a flat score to be given to an article, which would be based on how often it cites retracted papers, how many retracted papers it cites, and at what level those retractions occur. However, we determined that this method is nearly impossible to implement with any degree of consistency as retractions are a complicated and multifaceted problem (see ethical analysis for more). Instead, we have decided on giving the user a more raw output of what we find. For each analysis, an output a table will be created, which gives the number of first, second, and third level retractions for each paper that was analyzed (table 1).

Paper ID	First level retractions	Second level retractions	Third level retractions
123456	1	7	15

Table 1: Table representative of the output given for a particular paper that is being analysed. The number shown here are entirely fabricated for demonstrative purposes

To help answer the question of how pervasive retractions are within the medical literature, we looked to analysed 25 article subsets of our total database. Each 25 article subset would be taken from the publications of a particular journal. Then the numbers of cited papers within each level, and generated weighted averages for each category. Within these, we would also compare the standard deviations of these numbers between each individual paper. These numbers would then be compared across journals to see if retractions and their prevalence were more common to a particular journal or field within medicine.

However, constraints on the current size of the database limited our ability to analyze the prevalence of retractions in this way. The database only includes articles from 1990-2000, where we find far fewer retracted articles. While it is still expanding, at this time there is not enough information to go forward with this type of analysis.

Future work

The future work for this project includes downloading, parsing, and analyzing other papers from sources beyond PubMed's database, as well as looking further into the impact of retracted articles and papers. Opening up the analysis to other databases of scientific papers and articles would provide a stronger sample set and allow for potentially deeper and more intricate interactions between the papers. From past work on analyzing the impact of retracted papers and our own attempts at determining their influence, we found that the exact effects are difficult to determine. Further research, possibly using the tool we've created, could provide missing information on the impact for us and other researchers.

To include sources, an important step would be to optimize this code. This could come from parallelizing the import process into the SQL database. It could also come from the xml parsing program, which could be made to run faster through parallel computing or simple optimization algorithms and steps. This program could also be modified to handle larger xml documents, since currently the program can only handle smaller files and as such we need to manually break them up.

In creating this database, we have consolidated a lot of information regarding how articles are linked together. This could be used to create an interesting and informative visualization that shows how articles cite one another, and thus how information flows within the scientific community. Once the tool is more robust and provides high coverage of more articles, we could include methods to import a bibliography section from an article. With this, an author could simply import their works cited to the tool and check if they were cited any potentially retracted information.

Conclusion

Due to the limited size of our current database we cannot draw concrete conclusions on the breadth of the impact of retracted articles. However, we have laid the groundwork and created a tool that will allow for further analysis of the body of scientific literature as our sample size grows. We can safely determine that there is some impact of retracted papers on the PubMed Open Access database because there are at least 619 retracted papers in it. Included in the PubMed database is article with PMID 10190894 (unique_did 59164), which we have found does cite a known retracted article. With further research and analysis, the direct impact of these 619 papers can be discovered. Since there no tool known to us which looks at the impact of retracted papers or is able to find if a paper cites a previously retracted paper, our tool has the possibility to aid researchers while doing background research or even spark the conversation about how to deal with, classify, and keep track of retracted articles within the scientific community.

References

Atlas MC. Retraction policies of high-impact biomedical journals. J Med Lib Assoc. 2004 Apr;92(2):242–50.

Dunaiski, M. and Visser, W. Comparing paper ranking algorithms. In *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference*(Centurion, Tshwane, South Africa, 2012). ACM, New York, NY, USA, 21-30.

Gabehart, M. E. An analysis of citations to retracted articles in the scientific literature (Master's thesis). Retrieved from UNC School of Information and Library Science database.

Neal, Joseph M. "Closure on retraction of articles by Dr. Reuben." *Regional anesthesia and pain medicine* 34.4 (2009): 385.

Resnik, D. B., Wager, E., & Kissling, G. E. (2015). Retraction policies of top scientific journals ranked by impact factor. Journal of the Medical Library Association : JMLA, 103(3), 136–139.

Steen, R.G., Casadevall A, Fang FC (2013) Why Has the Number of Scientific Retractions Increased? PLoS ONE 8(7): e68397. doi:10.1371/journal.pone.0068397

Steen, R. G. Retractions in the scientific literature: is the incidence of research fraud increasing? *Journal of Medical Ethics* (2010): jme-2010.

Wright, K., & McDaid, C. (2011). Reporting of article retractions in bibliographic databases and online journals. Journal of the Medical Library Association : JMLA, 99(2), 164–167.