Predicting Citations Using Tf*IDF Scores

Erick Draayer
Department of Computer Science
North Carolina State University
Raleigh, North Carolina 27695
Email: Ecdraayer@Gmail.com

***Abstract*—The ability to predict how many citations a paper will receive after publishing could be very useful. Publishers could use these predictions to decide which papers are most likely to be successful and thus worth publishing. This idea is explored by using the top 50 and top 100 highest Tf*IDF scores for research papers as predictors. Although all text document Tf*IDF scores follow similar patterns, the scores themselves vary. Some text documents have much higher max scores then others, and some might have a lot more low scoring words. This study aimed to find out if research papers with similar amounts of citations also had analogous Tf*IDF scores.**

***Index Terms*—Text Mining, LaTeX, Citations, Tf*IDF**

I. Introduction

The goal of this project was to explore the idea of predicting the number of times a paper will be cited. This may be useful for publishers who ideally want to publish papers which they know will be successful. The research papers used for this study originate from the electronic repository arXiv.org. The repository houses many research papers relating to various fields in science and mathematics in a format called LaTeX. The dataset collected from this repository are specifically research papers related to high energy physics published in the year 2003. A lot of preprocessing was required, beginning with the acquisition of the citations. Next, since the research papers are in LaTeX format, the removal of the LaTeX commands and comments was necessary to avoid error and noise in calculations of our predictor variables. To achieve the goal of predicting citations, the term frequency times inverse document frequency (Tf*IDF) scores were calculated. These scores represent the relative importance of a word within a paper. The highest 50 and 100 Tf*IDF scores were calculated for each paper and used as the predictor variables themselves. The idea behind this approach is that papers with low, medium, and high citations have distinguishing Tf*IDF score values that can be used to predict the number of citations a paper will receive. The learners used for this experiment are MultilayerPerceptron, REPTree, and M5P. They are prebuilt and come from Weka.

II. The Data

The data consisted of 956 research papers published in 2003 that covered various subjects in high energy physics. The papers are standard text files in LaTeX format, meaning they include LaTeX tags. The file name for each file is a random five digit number preceded by the last two digits of the year they were published. In addition, individual text files containing the abstracts only were available. The data was retrieved from the 2003 KDD Cup website, and is originally sourced from the hep-th portion of the arXiv e-print database. Papers from previous years were also available, going back to 1992.

III. The Process of Preprocessing

A. Getting Citations

The first step taken to preprocess the data was retrieving all citations for each paper. To automate this process, a script taking advantage of the scholarly 0.2.2 Python module was written to find the number of citations for each paper. The module provides API that allows interaction with Google Scholar to get the number of citations, among other information, from Google Scholar search results. From the search results, only the records for the number one match were used to get the number of citations.

Originally, the searches were designed to use the title from the paper but this brought about several challenges. First, the actual retrieval of the titles from the papers had some difficulties. Most papers had tagged titles in the form of "\title{}", for which the fetching was trivial. However, not all papers were tagged like this; instead, the titles were embedded within other LaTeX commands that manipulate text. The problem with this scenario is that these other commands are varied and could be used for other text that appears in the research paper. One would have to recognize each possible tag that could be used to make a title in the paper and find some way to make sure it was the actual title and not some other text. Ignoring these papers would reduce the data size by 30% which could greatly affect the ability for our learner to learn. Besides this, there is another problem with searching by title. Google Scholar makes the assumption that its users are trying to find relevant and popular papers, which is normally the case. For our purposes, though, we are just as interested in the papers with two citations as the ones with thousands of them. This assumption means that if a paper has a generic title, Google will treat the entered title as a subject and return papers that pertain to the subject rather than papers with that title. In these situations, often the returned papers will have larger numbers of citations because they are more popular and so the intended paper will be mapped to an incorrect number of citations, biased upwards. For example, the paper *Inflating p-branes*, which has only 16 citations, will not show up as the

first result or even on the first page, but instead a paper with many more citations will. This could potentially introduce a lot of error in our already small dataset.

The solution to this problem is actually simple. Each paper is linked to some kind of identification number through arXiv. This identification number is the seven digit number file name for each file. Combined with the specific arXiv category the paper is filed under, in this case hep-ph, a new search term is made. Querying Google Scholar with this new search term will guarantee the correct paper as the number one result, if the search succeeds. For this reason, searching by identification is greatly preferred over searching by title. Even if the files are not named after the paper's identification number, this number can always be found within the text near the bottom and can be easily retrieved. In general, the younger the paper the more likely it is that the search by identification number will succeed. Searching the data set of research papers from 2003 and 2002 had a 100% success rate with this search method, with success rates taking a steady decline for older papers down to 30% by 1992.

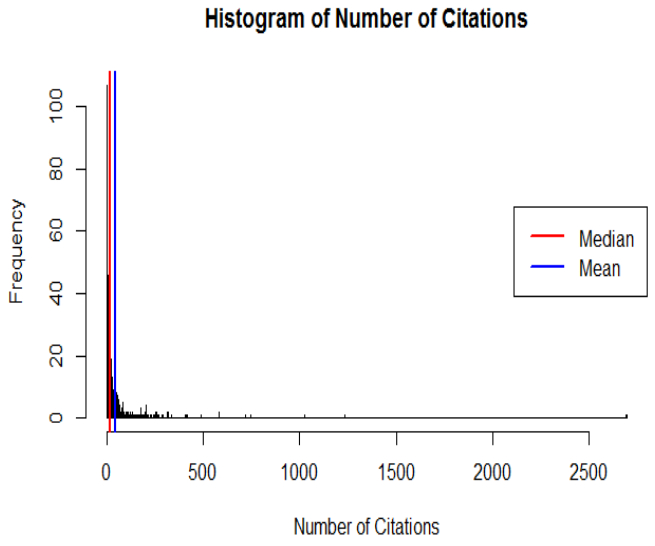

Fig. 1. Histogram of Number of Citations

A combination of both search methods was used in the final script to retrieve citations for the research papers. The script first tries to search by identification number. If this search fails, then a search by title is attempted. If this search fails then the file is recorded as not found and is ignored in later processing. Figure 1 shows the histogram of the number of citations. The histogram is heavily skewed right which indicates that there are a few papers which have a much higher number of citations than the majority of papers. Quantiles for these numbers can be seen in Table I. We see that while most papers have between 5 to 40.5 citations, some papers have as high as 2694 citations and some papers have none.

TABLE I
QUANTILES FOR NUMBER OF CITATIONS

Quantiles for Number of Citations	
	Number of Citations
Min	0
Q1	5
Median	16
Q3	40.5
Max	2694

B. DeTeXing the LaTeX

Initially, Tf*IDF score calculations were problematic because they did not account for the presence of LaTeX commands used to display equations, create figures, etc. These commands were incorrectly included as words when calculating the Tf*IDF scores, and consequently showed up as top scoring words. Additionally, the Tf*IDF score calculations were picking up personal comments made by the LaTeX file authors that were only meant to be seen through a LaTeX file editor, not part of the paper. Since this study focuses on how the words visible to regular readers will determine the citations for a paper, comments visible only to LaTeX editors should not be part of the score calculation. The removal of these LaTeX commands and comments was important for ensuring the integrity of the Tf*IDF scores.

A tool called DeTeX was used to remove the LaTeX comments and commands from the papers. DeTeX is an open source tool that is still regularly updated. DeTeX recognizes LaTeX commands and comments and purges them from the file without removing anything else. For the papers in this project, DeTeX generally worked very well but was unable to process 31 of the papers. This was an acceptable loss to the dataset.

C. Preprocessing the text

After the the LaTeX commands and author comments were removed, actual steps of analyzing the text could be taken. The overall theme of this process was to reduce the natural high dimensionality of the text documents. The first part of this process was tokenization. This involves the replacement of all punctuation with spaces, converting all letters to lowercase, and lastly removing non-printable characters. After this, a stop list was employed to remove words. The stop list contains words deemed unimportant when doing text mining analysis. Examples of these words include "a", "be", "can", "the". Figure 2 shows the full stop list used. Finally, stemming was used to merge words with the same or similar meaning. An example of this would be the words "measure", "measuring", and "measured". All these words have the same general meaning and therefore they should all be seen as the same word. With these three steps, the dimensionality of the text data was greatly reduced. All of these methods were employed using the Python module scikit-learn.

IV. TF*IDF SCORES

Tf*IDF scores were calculated using this equation:

```
ENGLISH_STOP_WORDS = frozenset([
    "a", "about", "above", "across", "after", "afterwards", "again", "against",
    "all", "almost", "alone", "along", "already", "also", "although", "always",
    "am", "among", "amongst", "amoungst", "amount", "an", "and", "another",
    "any", "anyhow", "anyone", "anything", "anyway", "anywhere", "are",
    "around", "as", "at", "back", "be", "became", "because", "become",
    "becomes", "becoming", "been", "before", "beforehand", "behind", "being",
    "below", "beside", "besides", "between", "beyond", "bill", "both",
    "bottom", "but", "by", "call", "can", "cannot", "cant", "co", "con",
    "could", "couldnt", "cry", "de", "describe", "detail", "do", "done",
    "down", "due", "during", "each", "eg", "eight", "either", "eleven", "else",
    "elsewhere", "empty", "enough", "etc", "even", "ever", "every", "everyone",
    "everything", "everywhere", "except", "few", "fifteen", "fifty", "fill",
    "find", "fire", "first", "five", "for", "former", "formerly", "forty",
    "found", "four", "from", "front", "full", "further", "get", "give", "go",
    "had", "has", "hasnt", "have", "he", "hence", "her", "here", "hereafter",
    "hereby", "herein", "hereupon", "hers", "herself", "him", "himself", "his",
    "how", "however", "hundred", "i", "ie", "if", "in", "inc", "indeed",
    "interest", "into", "is", "it", "its", "itself", "keep", "last", "latter",
    "latterly", "least", "less", "ltd", "made", "many", "may", "me",
    "meanwhile", "might", "mill", "mine", "more", "moreover", "most", "mostly",
    "move", "much", "must", "my", "myself", "name", "namely", "neither",
    "never", "nevertheless", "next", "nine", "no", "nobody", "none", "noone",
    "nor", "not", "nothing", "now", "nowhere", "of", "off", "often", "on",
    "once", "one", "only", "onto", "or", "other", "others", "otherwise", "our",
    "ours", "ourselves", "out", "over", "own", "part", "per", "perhaps",
    "please", "put", "rather", "re", "same", "see", "seem", "seemed",
    "seeming", "seems", "serious", "several", "she", "should", "show", "side",
    "since", "sincere", "six", "sixty", "so", "some", "somehow", "someone",
    "something", "sometime", "sometimes", "somewhere", "still", "such",
    "system", "take", "ten", "than", "that", "the", "their", "them",
    "themselves", "then", "thence", "there", "thereafter", "thereby",
    "therefore", "therein", "thereupon", "these", "they", "thick", "thin",
    "third", "this", "those", "though", "three", "through", "throughout",
    "thru", "thus", "to", "together", "too", "top", "toward", "towards",
    "twelve", "twenty", "two", "un", "under", "until", "up", "upon", "us",
    "very", "via", "was", "we", "well", "were", "what", "whatever", "when",
    "whence", "whenever", "where", "whereafter", "whereas", "whereby",
    "wherein", "whereupon", "wherever", "whether", "which", "while", "whither",
    "who", "whoever", "whole", "whom", "whose", "why", "will", "with",
    "within", "without", "would", "yet", "you", "your", "yours", "yourself",
    "yourselves"])
```

Fig. 2. List of stop words

$$\text{Tf*IDF}_i = W[i]/W\_total * log(D\_total/D[i])$$

This equation is simple and easy to understand. Here, i is the word for which we are calculating a Tf*IDF score. $W[i]$ represents the number of times word i appears in the set of documents, D_tot. $D[i]$ is the number of documents in which word i appears and W_tot is the total number of tokens in our set of documents. This equation was employed through the Python scikit-learn module. Results of the top 100 Tf*IDF scores can be seen in Figure 3. Looking at the graph we see that all document Tf*IDF scores follow a similar shape. However, they differ by maximum and minimum scores and rate of decline. It was the hope of this study that these differences would correspond directly with how many citations a paper will receive.

V. Results

The results come from running weka explorer prebuilt learners with the created data. The learners used are REPTree, Multilayer Perceptron, and M5P. Ten fold cross-validation was used to test the accuracies of the learners. Each learner reported the correlation coefficient, which gives us an idea of how well our predictors work. Both the top highest 50 and 100 Tf*IDF score files were used to see if one could get better correlations than the other. We chose not to use more than 100 scores due to computation limitations. The summary of the results is reported in Table II. None of the learners reported promising results that our hypothesis is true. The highest correlation was 0.0634 which was reported by the Multilayer Perceptron using the 100 Tf*IDF score dataset. An example of the REPTree constructed by weka using the highest 50 Tf*IDF scores data can be seen in Figure 4.

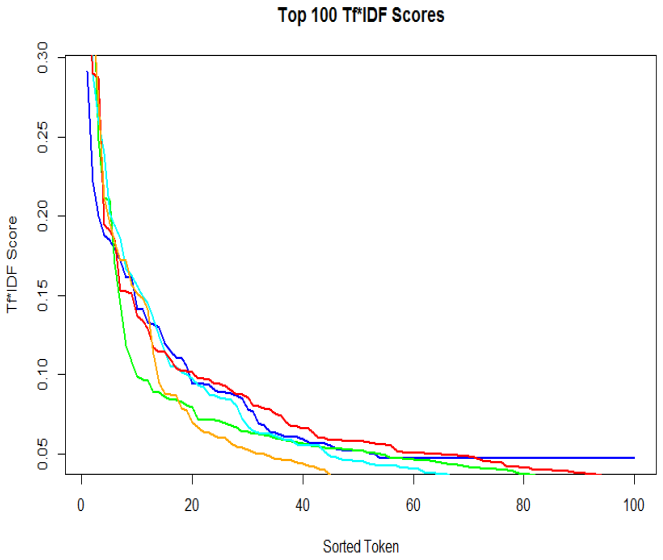

Fig. 3. Tf*IDF scores for five random doucments

TABLE II
Correlation between TFIDF scores and number of citations

Correlation from Learners			
Number of Tf*IDF Scores	REPTree	Multilayer Perceptron	M5P
Highest 50	-0.0475	0.0468	-0.0281
Highest 100	-0.0347	0.0634	0.0484

VI. Conclusion

Tf*IDF scores were shown to be a very poor predictor for number of citations. Each learner applied to the dataset reported a correlation of almost zero. Clearly a different approach is needed to accurately predict the number of citations. Another method that could be used is to instead combine all of the papers together. Then, run Tf*IDF on this combined document to find the overall highest scoring words. For each individual document, calculate the term frequencies of these highest scoring words. These term frequency numbers now become the independent variables used to predict number of citations. This new method focuses more on the idea that

```
REPTree
============

 Score32  < 0.09
|    Score1  < 0.36 : 43.67 (214/12140.84) [128/3202.05]
|    Score1  >= 0.36
|   |    Score15  < 0.13
|   |   |    Score15  < 0.09 : 48.24 (90/6115.23) [38/16770.14]
|   |   |    Score15  >= 0.09
|   |   |   |    Score30  < 0.07
|   |   |   |   |    Score34  < 0.06 : 31.3 (155/1312.41) [64/7208.45]
|   |   |   |   |    Score34  >= 0.06
|   |   |   |   |   |    Score35  < 0.07
|   |   |   |   |   |   |    Score31  < 0.07
|   |   |   |   |   |   |   |    Score32  < 0.07 : 19.64 (18/54.81) [7/6870.54]
|   |   |   |   |   |   |   |    Score32  >= 0.07
|   |   |   |   |   |   |   |   |    Score23  < 0.08 : 13.96 (14/44.41) [12/598.1]
|   |   |   |   |   |   |   |   |    Score23  >= 0.08
|   |   |   |   |   |   |   |   |   |    Score31  < 0.07 : 38.67 (4/108.75) [5/2410.65]
|   |   |   |   |   |   |   |   |   |    Score31  >= 0.07 : 15.63 (19/82.83) [11/241.42]
|   |   |   |   |   |   |   |    Score31  >= 0.07 : 3.73 (12/12.89) [3/12.78]
|   |   |   |   |   |   |    Score35  >= 0.07 : 99 (2/361) [1/25281]
|   |   |   |   |    Score30  >= 0.07
|   |   |   |   |   |    Score30  < 0.07 : 205.25 (18/7690.46) [6/1164700.78]
|   |   |   |   |   |    Score30  >= 0.07 : 33.59 (77/991.83) [40/9334.75]
|   |   |    Score15  >= 0.13 : 86.8 (4/13577.69) [1/9457.56]
 Score32  >= 0.09 : 344 (3/231884.67) [0/0]

Size of the tree : 25
```

Fig. 4. Example of REPTree constructed using weka

papers with similar word counts will share similar number of citations. Applying a filter to the data could also be used to try and improve results.

Although our results are not promising, there are still many more ways to use this data in terms of running different learners and tuning parameters that may show better evidence for a relationship between Tf*IDF scores and citation numbers.

ACKNOWLEDGMENT

Thank you to Dr. Timothy Menzies for guiding me through my first independent study. Also thanks to all of his TAs for giving insights into this project.

REFERENCES

[1] "KDD Cup 2003" KDD Cup 2003. [Online]. Available: https://www.cs.cornell.edu/projects/kddcup/. [Accessed: 05-May-2017].
[2] "arXiv.org" arXiv.org e-Print archive. [Online]. Available: https://arxiv.org/. [Accessed: 05-May-2017].
[3] "scikit-learn," scikit-learn: machine learning in Python scikit-learn 0.18.1 documentation. [Online]. Available: http://scikit-learn.org/stable/. [Accessed: 05-May-2017].
[4] "Weka 3: Data Mining Software in Java," Weka 3 - Data Mining with Open Source Machine Learning Software in Java. [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/. [Accessed: 05-May-2017].