On the Use of Web Search to Improve Scientific Collections

Krutarth Patel, Cornelia Caragea, Sujatha Das Gollapalli
Kansas State University, University of Illinois at Chicago, National University of Singapore
kipatel@ksu.edu, cornelia@uic.edu, idssdg@nus.edu.sg

Abstract

Despite the advancements in search engine features, ranking methods, technologies, and the availability of programmable APIs, current-day open-access digital libraries still rely on crawl-based approaches for acquiring their underlying document collections. In this paper, we propose a novel search-driven framework for acquiring documents for such scientific portals. Within our framework, publicly-available research paper titles and author names are used as queries to a Web search engine. We were able to obtain $\approx 267,000$ unique research papers through our fully-automated framework using $\approx 76,000$ queries, resulting in almost $200,000$ more papers than the number of queries. Moreover, through a combination of title and author name search, we were able to recover 78% of the original searched titles.

1 Introduction

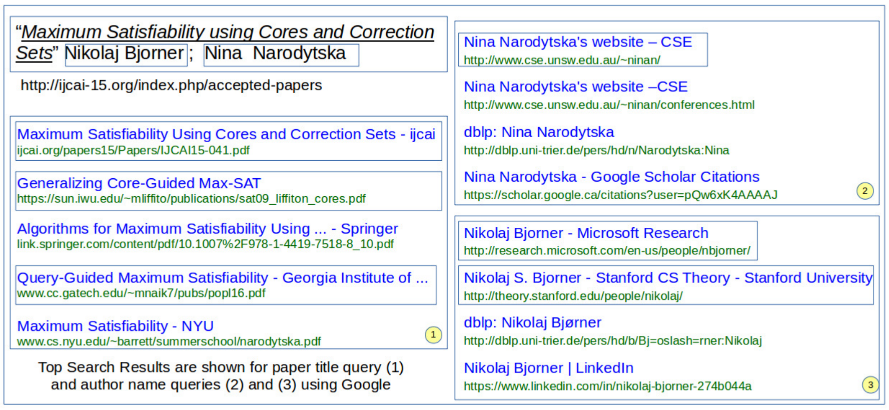
Scientific portals such as Google Scholar, Semantic Scholar, ACL Anthology, CiteSeer^x, and ArnetMiner, provide access to scholarly publications and comprise indispensable resources for researchers who search for literature on specific subject topics. Moreover, many applications such as document and citation recommendation (Bhagavatula et al., 2018; Zhou et al., 2008), expert search (Balog et al., 2007; Gollapalli et al., 2012), topic classification (Caragea et al., 2015; Getoor, 2005), and keyphrase extraction and generation (Meng et al., 2017; Chen et al., 2020) involve Web-scale analysis of up-to-date research collections.

Open-access, autonomous systems such as CiteSeer^x and ArnetMiner acquire and index freely-available research articles from the Web (Li et al., 2006; Tang et al., 2008). Researchers' homepages and paper repository URLs are crawled for maintaining the research collections in these portals, using focused crawling. Needless to say, the crawl seed lists cannot be comprehensive in the face of the ever changing Scholarly Web. Not only do new authors and publication venues emerge, but also existing researchers may stop publishing or they may change affiliations, resulting in outdated seed URLs. *Given this challenge, how can we automatically augment the document collections in open-access scientific portals?*

To address this question, in this paper, we propose a novel framework (based on Web search) for both automatically acquiring and processing research documents. To motivate our framework, we recall how a Web user typically searches for research papers or authors. As with regular document search, a user typically issues Web search queries comprising of representative keywords or paper titles for finding publications on a topic. Similarly, if the author is known, a "navigational query" (Broder, 2002) may be employed to locate the homepage where the paper is likely to be hosted. To illustrate this process, Figure 1 shows an anecdotal example of a search using Google for the title and authors of a research article. As can be seen from the figure, the intended research paper and the researchers' homepages (highlighted in sets 2 and 3) are accurately retrieved. Moreover, among the top-5 results shown for the title query (set 1), four of the five results are research papers on the same topic (i.e., the first four results). The document at the Springer link is not available for free, whereas the last document corresponds to course slides. The additional three papers are potentially retrieved because scientific paper titles comprise a large fraction of keywords (Chen et al., 2019), and hence, the words in these titles serve as excellent keywords that can retrieve not only the intended paper, but also other relevant documents.

Our framework mimics precisely the above search and scrutinize the approach adopted by

Proceedings of the First Workshop on Scholarly Document Processing, pages 174–183
Online, November 19, 2020. ©2020 Association for Computational Linguistics
https://doi.org/10.18653/v1/P17


"*Maximum Satisfiability using Cores and Correction Sets*" Nikolaj Bjorner ; Nina Narodytska

http://ijcai-15.org/index.php/accepted-papers

Maximum Satisfiability Using Cores and Correction Sets - ijcai
ijcai.org/papers15/Papers/IJCAI15-041.pdf

Generalizing Core-Guided Max-SAT
https://sun.iwu.edu/~mliffito/publications/sat09_liffiton_cores.pdf

Algorithms for Maximum Satisfiability Using ... - Springer
link.springer.com/content/pdf/10.1007%2F978-1-4419-7518-8_10.pdf

Query-Guided Maximum Satisfiability - Georgia Institute of ...
www.cc.gatech.edu/~mnaik7/pubs/popl16.pdf

Maximum Satisfiability - NYU
www.cs.nyu.edu/~barrett/summerschool/narodytska.pdf
1

Top Search Results are shown for paper title query (1) and author name queries (2) and (3) using Google

Nina Narodytska's website – CSE
http://www.cse.unsw.edu.au/~ninan/

Nina Narodytska's website –CSE
http://www.cse.unsw.edu.au/~ninan/conferences.html

dblp: Nina Narodytska
http://dblp.uni-trier.de/pers/hd/n/Narodytska:Nina

Nina Narodytska - Google Scholar Citations
https://scholar.google.ca/citations?user=pQw6xK4AAAAJ
2

Nikolaj Bjorner - Microsoft Research
http://research.microsoft.com/en-us/people/nbjorner/

Nikolaj S. Bjorner - Stanford CS Theory - Stanford University
http://theory.stanford.edu/people/nikolaj/

dblp: Nikolaj Bjørner
http://dblp.uni-trier.de/pers/hd/b/Bj=oslash=rner:Nikolaj

Nikolaj Bjorner | LinkedIn
https://www.linkedin.com/in/nikolaj-bjorner-274b044a
3

Figure 1: An anecdotal search example for illustration.

Scholarly Web users. Freely-available information from the Web for specific subject disciplines[1] is used to frame title and author name queries in our framework. Our contributions are as follows:

- We propose a novel integrated framework based on search-driven methods to automatically acquire research documents for scientific collections. To our knowledge, we are the first to use "Web Search" based on author names to obtain seed URLs for initiating crawls in an open-access digital library.
- We design a novel homepage identification module and adapt existing research on academic document classification, which are crucial components of our framework. We show experimentally that our homepage identification module and the research paper classifier substantially outperform strong baselines.
- We perform a large-scale, first-of-its-kind experiment using $43,496$ research paper titles and $32,816$ author names from Computer and Information Sciences. We compare our framework with two baselines, a breadth-first search crawler and, to the extent possible, the Microsoft Academic. We discuss that our framework does not substitute these systems, but rather they very well complement each other. As part of our contributions, we will make all the constructed datasets available.

2 Our Framework

Figure 2 shows the control flow paths of our proposed framework to obtain research papers and thus augment existing collections. In **Path 1**, paper titles are used as queries and the PDF documents

[1] For example, from bibliographic listings such as DBLP or paper metadata available in ACM DL.

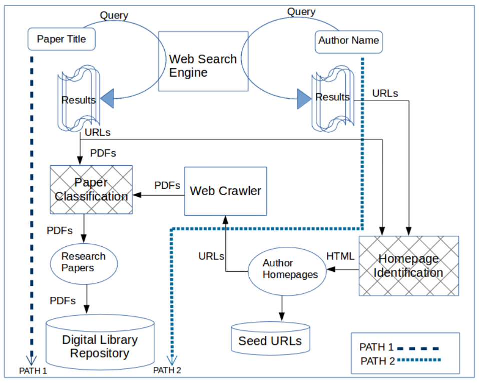


Figure 2: Schematic Diagram of our Framework.

resulting from each title search are classified with a paper classifier based on Random Forest. Author names comprise the queries for Web search in **Path 2**, the results of which are filtered by a homepage identification module trained using RankSVM. The predicted author homepages from **Path 2** serve as seed URLs for the crawler module that obtains all documents up to a depth 2 starting from each seed URL. The paper classification module is once again employed to retain only research papers. Note that we crawl only publicly available and downloadable documents those appear in the search responses of the Web search or from the researcher homepages.

The accuracy and efficiency of our Search/Crawl framework is contingent on the accuracy of two components: (1) the homepage identifier, and (2) the paper/non-paper classifier.

2.1 Homepage Identification

Among the works focusing on researcher homepages, both Tang et al. (2007) and Gollapalli et al. (2015) treated homepage finding as a binary classification and used URL string features and content features (extracted from the entire .html page) for classification. However, given our Web search setting, the non-homepages retrieved in response to an author name query can be expected to be diverse with webpages ranging from commercial websites such as LinkedIn, social media websites such as Twitter and Facebook, and several more. To handle this aspect, we frame homepage identification as a supervised ranking problem. Thus, given a set of webpages in response to a query, our objective is to rank homepages higher relative to other types of webpages, capturing our preference among the retrieved webpages. Preference information needed for the ranking can be easily modeled through appropriate objective functions in learning to rank approaches (Liu, 2011). For example, RankSVM (Joachims, 2002) minimizes the Kendalls τ measure based on the preferential ordering information in the training examples.

Note that, unlike classification approaches that independently model both positive (homepage) and negative (non-homepage) classes, we are modeling instances in relation with each other with preferential ordering (Wan et al., 2015). In Section 4, we show that our ranking approach outperforms classification approaches for homepage identification. We design the following feature types for our ranking model, which capture aspects (e.g., snippets) useful for a Web user to find homepages:

1. **URL Features**: Intuitively, the URL strings of academic homepages can be expected to contain (or not) certain tokens. For example, a homepage URL is less likely to be hosted on domains such as "linkedin" and "facebook." On the other hand, terms such as "people" or "home" can be expected to occur in the URL strings of homepages (see examples of homepage URLs in Figure 1). We tokenize the URL strings based on the "slash (/)" separator and the domain-name part of the URL based on the "dot (.)" separator to extract our URL and DOMAIN feature dictionaries.

2. **Term Features**: The current-day search engines display the Web search results as a ranked list, where each webpage is indicated by its HTML title, the URL string as well as a brief summary of the content of the webpage (also known as the "snippet"). We posit that Scholarly Web users are able to identify homepages among the search results based on the term hints in titles and snippets (for example, "professor", "scientist", "student"), and use words from titles and snippets to extract our TITLE and SNIPPET dictionaries.

3. **Name-match Features**: These features capture the common observation that researchers tend to use parts of their names in the URL strings of their homepages (Tang et al., 2007; Gollapalli et al., 2015). We specify two types of match features: (1) a boolean feature that indicates whether any part of the author name matches a token in the URL string, and (2) a numeric feature that indicates the extent to which name tokens overlap with the (non-domain part of) URL string given by the fraction: $\frac{\#\texttt{matches}}{\#\texttt{nametokens}}$. For the example author name "Soumen Chakrabarti" and the URL string: **`www.cse.iitb.ac.in/~soumen`**, the two features have values "true" and 0.5, respectively.

The dictionary sizes for the above feature types based on our training datasets (see Section 3) are listed below:

Feature Type	Size
URL+DOMAIN term features	2025
TITLE term features	19190
SNIPPET term features	25280
NAME match features	2

2.2 Paper/Non-Paper Classification

In order to obtain accurate paper collections, it is important to employ a high-accuracy paper/non-paper classifier. Caragea et al. (2016) studied the classification of academic documents into six classes: Books, Slides, Theses, Papers, CVs, and Others. The authors showed that a small set of 43 structural, text density, and layout features (Str) that are designed to incorporate aspects specific to research documents, are highly indicative of the class of an academic document. Because we are mainly interested in research papers to augment research collections and because binary tasks are considered easier to learn than multi-class tasks (Bishop, 2006), we adapted this prior work on multi-class document type classification (Caragea et al., 2016) and re-trained the classifiers for the two-class setting: paper/non-paper.

3 Datasets

The datasets used in the evaluation of our framework and its components are summarized in Table 1 and are described below:

DBLP Homepages. For evaluating homepage finding using author names, we use the researcher

Dataset		
DBLP Homepages		42,548(T) 4,255(+)
Research Papers	(Train)	960(T) 472(+)
	(Test)	959(T) 461(+)
CiteSeer^x		43,496 (Titles), 32,816 (Authors)

Table 1: Summary of datasets. Total and positive instances are shown using (T) and (+), respectively.

homepages from DBLP. In contrast to previous works that use this dataset to train homepage classifiers on academic websites (Gollapalli et al., 2015), in our Web search scenario, the non-homepages from the search results of an author name query need not be restricted to academic websites. Except the true homepage, all other webpages therefore correspond to negatives. We constructed the DBLP homepages dataset as follows: DBLP provided a set of author homepages along with the authors' names. Using these authors' names as queries, we perform Web search using Bing API and scan the top-10 results (Spink and Jansen, 2004) in response to each query. If the true homepage provided by DBLP is listed among the top-10 search results, this URL and the others in the set of Web results are used as training instances. We were able to locate homepages for $4,255$ authors in the top-10 results for the author homepages listed in DBLP.

Research Papers. To evaluate the paper/non-paper classifier, we used two independent sets of ≈ 1000 documents each, randomly sampled from the crawl data of CiteSeer^x, obtained from Caragea et al. (2016). These sets, called Train and Test, respectively, were manually labeled with six classes: Paper, Book, Thesis, Slides, Resume/CV, and Others. We transform the documents' labels as the binary labels, Paper/Non-paper.

CiteSeer^x. Our third dataset is compiled from the CiteSeer^x digital library. Specifically, we extracted research papers that were published in venues related to machine learning, data mining, information retrieval and computational linguistics. These venues along with the number of papers in each venue are listed in Table 2. Overall, we obtained a set of $43,496$ paper titles and $32,816$ authors (unique names) for the evaluation of our framework at a large scale.

Total # of papers: 43,496, #authors (unique): 32,816
NIPS (5211), IJCAI (4721), ICRA (3883), ICML (2979), ACL (2970), VLDB (2594), CVPR (2373), AAAI (2201), CHI (2030), COLING (1933), KDD (1595), SIGIR (1454), WWW (1451), CIKM (1408), SAC (1191), LREC (1128), SDM (1111), EMNLP (920), ICDM (891), EACL (760), HLT-NAACL (692)

Table 2: Conference venue (#papers) in the CiteSeer^x dataset.

4 Experiments and Results

In this section, we describe our experiments on homepage identification and paper classification along with their performance within the search then crawl then process paper acquisition framework.

Performance measures. We use the standard measures Precision, Recall, and F1 for summarizing the results of author homepage identification and paper classification (Manning et al., 2008). Unlike classification where we consider the true and predicted labels for each instance (webpage), in RankSVM the prediction is per query (Joachims, 2002). That is, the results with respect to a query are assigned ranks based on scores from the RankSVM and the result at rank-1 is chosen as the predicted homepage.

4.1 Author Homepage Identification

We aim to determine how accurate is RankSVM in identifying a homepage for each author name query. Table 3 shows the five-fold cross-validation performance of the homepage identification on the positive class trained using RankSVM compared with various classification algorithms, Naïve Bayes, Maximum Entropy and Support Vector Machines. The results in the table are averaged across all five test sets of cross-validation. Hyperparameter tuning (e.g., C for SVM) was performed on a development set extracted from training.

Method	Precision	Recall	F1
RankSVM	**0.8933**	0.8933	**0.8933**
Naïve Bayes	0.4830	**0.9239**	0.63432
MaxEnt	0.8207	0.8002	0.8102
Binary SVM	0.8353	0.8149	0.8249

Table 3: RankSVM vs. supervised classifiers on DBLP.

As can be seen from the table, RankSVM performs much better compared with the classification approaches, in terms of Precision and F1, although Recall is higher for Naïve Bayes. Hence, RankSVM is able to capture the relative preferential ordering among the search results and performs the best in identifying the correct author homepage in response to a query. A possible reason for the lower performance of the classification approaches such as Binary SVMs, Naïve Bayes, and Maximum Entropy is that they model the positive and negative instances independently and not in relation to one another for a given query. Moreover, the diversity in webpages among the negative class is ignored and they are modeled uniformly as a single class in the classification approaches.

4.2 Research Paper Classification

We compare the performance of classifiers trained using the 43 structural features (Str) with that of classifiers trained using the "bag of words" (BoW), URL-based features (URL), and a Convolutional Neural Network (CNN) model. For BoW and URL, we used the same text processing operations as in Caragea et al. (2016). We experimented with several classifiers: Random Forest (RF), Decision Trees (DT), Naïve Bayes Multinomial (NBM), and Support Vector Machines with a linear kernel (SVM). For CNN, we use the words as a sequence as an input; we first get the word embeddings as a part of the network followed by the CNN filter, max-pooling, concatenation, and the fully connected layer for the classification task, similar to Kim (2014). All models are trained on the "Train" dataset and are evaluated on the "Test" dataset. We tuned model hyper-parameters in 10-fold cross-validation experiments on "Train" (e.g., C for SVM and the number of trees for RF).

Feature/Cls. (Setting)	Precision	Recall	F1
BoW / DT (P-B)	0.860	0.920	0.889
URL / SVM (P-B)	0.729	0.729	0.729
Str / RF (P-B)	**0.933**	**0.967**	**0.950**
CNN (P-B)	0.816	0.890	0.851
Str / RF (A-B)	**0.952**	**0.951**	**0.951**
Str / RF (P-M)	0.918	0.965	0.941
Str / RF (A-M)	0.893	0.902	0.892

Table 4: Performance of paper classifier on "Test". "P" stands for the paper class, while "A" for the average of classes. "B" and "M" stand for binary and multi-class, respectively.

Table 4 shows the performance (Precision, Recall, and F1) for the binary setting on "Test" for each feature type, BoW, URL, and Str, and the CNN, with the classifiers that give the best results for the corresponding feature type or model (first four lines). The results are shown for the "paper" class (P). In the table, we also show the performance on the "paper" class with the multi-class (M) setting and the weighted averages (A) of all measures over all classes for both the settings. As can be seen from the table, the best classification performance is obtained using Random Forest trained on the 43 structural features with the overall performance above 95% being substantially higher in the binary setting compared with the multi-class setting. The reason behind lower performance of the CNN classifier can be the wide variety of documents present in the dataset and the small number of the training examples.

Title Queries
Knowledge-based Knowledge Elicitation. filetype:pdf
Solving Time-Dependent Planning Problems. filetype:pdf
Author Name Queries
Eric T. Baumgartner filetype:html
Nelson Alves filetype:html

Table 5: Example of title and author name queries.

4.3 Large-Scale Experiments

Finally, we evaluate our "search then crawl then process" framework and its components in practice in large scale experiments, using our CiteSeer^x subset. To this end, we evaluate the capability of our framework to obtain large document collections, quantified by the number of research papers it acquires (through both paths). For **Path 1**, we use the $43,496$ paper titles directly as search queries. Structural features extracted from the resulting PDF documents of each search are used to identify research papers with our paper classifier. For **Path 2**, the $32,816$ unique author names are used as queries. The RankSVM-predicted homepages from the results of each author name query are crawled for PDF documents up to a depth of 2, using the wget utility.[2] Again, the paper classifier is employed to identify the papers from the crawled documents. In all experiments, we used the Bing API to perform Web searches. Examples of title and author name queries are provided in Table 5.

4.3.1 Overall Yield

The total numbers of PDFs and research papers found through the two paths in our Search/Crawl/Process framework are shown in Table 6 (the columns labeled as #CrawledPDFs and #PredictedPapers, respectively). Intuitively, the overall yield can be expected to be higher through **Path 2**. This is because once an author homepage is reached, other research papers that are linked from this homepage can be directly obtained. Indeed, as shown in the table, the numbers of PDFs as well as predicted papers are significantly higher along **Path 2**. Crawling the RankSVM-predicted homepages of the $32,816$ authors, we obtain on average ≈ 14 research papers per query ($\frac{452273}{32816} = 13.78$). In contrast, examining only the top-10 search results along **Path 1**, we obtain ≈ 5 papers per query on average ($\frac{213683}{43496} = 4.91$). The high percentage of papers found along **Path 2** is consistent with previous findings that researchers tend to link to their papers via their homepages (Lawrence, 2001; Gol-

[2] https://www.gnu.org/software/wget/

#Queries	#CrawledPDFs	#PredictedPapers	#UniquePapers	#MatchesWithOriginalTitles
43,496 titles (Path 1)	322,029	213,683	91,237	32,565
32,816 names (Path 2)	665,661	452,273	204,014	17,627
Overlap: **Path 1** & **Path 2**	-	-	**28,374**	**16,188**
Total # of papers: **Path 1** + **Path 2**	-	-	**266,877**	**34,004**

Table 6: Number of papers obtained through **Path 1** and **Path 2** in our Search/Crawl/Process framework.

lapalli et al., 2015). Note that in all experiments, since the original $43,496$ titles are extracted from CiteSeer^x, for a fair evaluation, we removed all title search results that point to the CiteSeer^x domain, i.e., http://citeseerx.ist.psu.edu/.

Furthermore, the numbers of unique papers found along each of the two paths are shown in Table 6 (the column labeled as #UniquePapers). We used ParsCit[3] to extract the titles of the research papers obtained from both the paths and then calculated the duplicates from these titles.[4] As can be seen from the table, we are able to obtain $91,237$ and $204,014$ unique papers from **Path 1** and **Path 2**, respectively, which account for ≈ 2 papers per title query on average ($\frac{91237}{43496} = 2.09$) and ≈ 6 papers per author query on average ($\frac{204014}{32816} = 6.21$). However, since our objective is not to use one path or the other, but use a combination of both Path 1 and Path 2, we further expanded our analysis to show the overlap between Path 1 and Path 2 in terms of unique titles.

4.3.2 Overlap between Path 1 and Path 2

Table 6 shows also the overlap in the two sets of unique papers (between **Path 1** and **Path 2**), which is $28,374$. Compared to the overall yields along **Path 1** and **Path 2** ($213,683$ and $452,273$, respectively) and even with the number of unique papers along each path, this small overlap indicates that the two paths are capable of reaching different sections of the Web and play complementary roles in our framework. For example, the top-20 domains of the URLs from which we obtained research papers along **Path 1** are shown in Figure 3. As can be seen from the figure, via Web search, we are able to reach a wide range of domains. This is unlikely in crawl-driven methods without an exhaustive list of seeds since only links up to a specified depth from a given seed are explored (Manning et al., 2008). Interestingly, using a combination of both Path 1 and Path 2, we were able to obtain $266,877$ (=$91,237$+$204,014$-$28,374$) unique papers.

[3] http://aye.comp.nus.edu.sg/parsCit/

[4] To find duplicates, we convert the text to lowercase, and remove punctuation and whitespace.

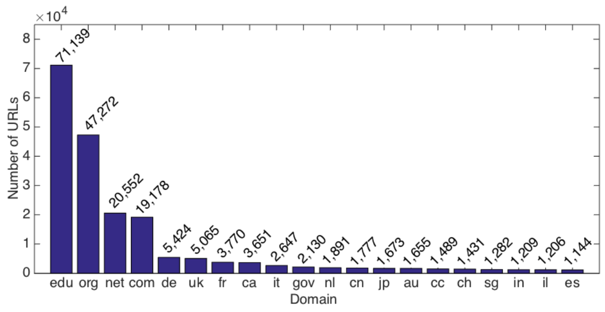


Figure 3: The top-20 domains from which papers were obtained along Path 1 of our framework.

Next, we investigate the recovery power of our framework. Precisely, how many of the original $43,496$ titles were found through each path as well as their combination?

4.3.3 Overlap with the Original Titles

The numbers of papers that we were able to obtain from the original $43,496$ titles through both paths are shown in the last column of Table 6, labeled as #MatchesWithOriginalTitles. To compute these matches, we used the title and author names available in our CiteSeer^x subset to look up the first page of each PDF document. As can be seen from the table, we were able to recover 75% ($\frac{32565}{43496}$) of the original titles through **Path 1** compared to the 40% ($\frac{17627}{43496}$) through **Path 2**. The total number of matches with the original titles between **Path 1** and **Path 2** was $16,188$. Overall, through a combination of both **Path 1** and **Path 2**, we were able to recover 78% ($\frac{34,004}{43496}$) of the original titles ($34,004$=$32,565$+$17,627$-$16,188$ papers obtained through both paths out of the original titles).

To summarize, using about $76,312$ queries ($43,496 + 32,816$) through **Path 1** and **Path 2**, we are able to build a collection of $665,956$ papers ($213,683 + 452,273$) and $266,877$ unique titles ($91,237 + 204,014 - 28,374$). About 32-33% of the obtained documents are "non-papers" along both paths. Scholarly Web is known to contain a variety of documents including resumes, and presentation slides (Ortega et al., 2006). Some of these documents may include the exact paper titles and may appear in paper search results as well as be linked from author homepages.

4.3.4 Anecdotal Evidence

Given the size of our CiteSeer^x dataset and the large number of documents obtained via our framework (as shown in Table 6), it is extremely labor-intensive to manually examine all documents resulting from the large scale experiment. However, since our classifiers and rankers achieve performance above 95% and 89% based on our test datasets compiled specifically for these tasks, we expect them to continue to perform well "in the wild." We show anecdotal evidence to support this claim, i.e., an estimate of how many true papers we are able to obtain via our Search/Crawl/Process framework starting from a small set of titles.

To obtain such an estimate, we randomly selected 10 titles from the CiteSeer^x dataset. From the corresponding papers of these 10 titles, we extracted 33 unique authors. We manually inspected all PDFs that can be obtained via title search (**Path 1**) as well as the homepages obtained via author name search (**in Path 2**) That is, through **Path 1**, we searched the Web for the 10 selected titles and manually examined and annotated the top-10 resulting PDFs for each title query. The title search resulted in 59 PDFs, of which 33 are true papers and 26 are non-papers. Our paper classifier predicted 32 out of 33 papers correctly and 38 papers overall and achieved a precision and recall of 84% and 97%, respectively.

Similarly, through **Path 2**, we searched for the 33 author names from the Web and manually examined and annotated the top-10 resulting webpages for each author name query. From the author search, manually, we were able to locate 19 correct homepages of the 33 authors. A manual inspection of the predicted homepages revealed that our framework was not able to locate 6 out of the 19 correct homepages. Table 7 shows a few examples where our framework was not able to locate the correct homepages. For example, occasionally, RankSVM ranks university faculty profile or faculty research group at the first rank, which is then predicted as a homepage by RankSVM (e.g., URLs 4 and 6 in Table 7). URL 5 in Table 7 is wrongly predicted by RankSVM as the homepage for the researcher name "David Bell." This is precisely because there is a well known novel writer and also baseball player with the same name, which get ranked higher in the results of the search engine. Note that, interestingly, the actual homepage of David Bell, corresponding to URL 2 was not retrieved in the top 10 search responses of Bing.

Actual homepage
1. http://destrin.smalldata.io/
2. http://www.cs.qub.ac.uk/~D.Bell/dbell.html
3. http://www.ai.sri.com/~yang

Predicted homepage
4. http://research.cens.ucla.edu/people/estrin/
5. http://davidbellnovels.com/
6. http://www.ai.sri.com/people/yang/

Table 7: A few examples where our framework was not able to locate the correct homepage

4.4 Baseline Comparisons

Breadth-first search crawler. We compare our Search/Crawl/Process framework, through **Path 1**, with a breadth-first search crawler as implemented in CiteSeer^x. The CiteSeer^x crawler starts with a list of seed URLs, performs a breadth-first search crawl and saves open-access PDF documents.

For this experiment, we randomly selected $1,000$ titles from DBLP. We then searched the Web for these titles and retrieved the top-10 resulting PDFs for each query. Through this search, we obtained a total of $5,793$ PDFs, from which we removed 110 documents that were downloaded from CiteSeer^x, since they were obtained as a result of the CiteSeer^x breadth-first search crawler. Note that there is an overlap of 6 documents, i.e., only located on CiteSeer^x (and nowhere else on the Web), between the 110 removed documents and the 1000 DBLP initial titles. From the remaining documents, our paper classifier predicted $3,427$ documents as papers, out of which $2,797$ are unique papers/titles. We searched CiteSeer^x for these $2,797$ titles to determine how many of them are found by the CiteSeer^x crawler. We found $1,037$ titles in CiteSeer^x by checking if one title string contains the other. Thus, with our framework, we were able to obtain $2,797 - 1,037 = 1,760$ additional papers. Out of the 994 ($1000 - 6$) DBLP titles, only 121 papers were found by both our framework and the CiteSeer^x crawler. In addition, our framework found 165 more papers (with a total of 286 out of 994 DBLP titles), whereas the CiteSeer^x crawler found only 92 more papers (with a total of 213 out of 994 DBLP titles). Moreover, out of the additional yield of our framework, i.e., 2511 ($= 2797 - 286$) papers, only 552 are found by the CiteSeer^x crawler (identified by searching for the 2511 titles in the CiteSeer^x digital library - by exact match). These results are summarized in Figure 4. We note that the two approaches are not substitut-

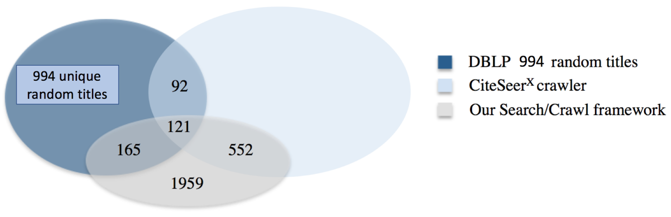


Figure 4: Comparison of the Search/Crawl framework with the CiteSeer^x breadth-first search crawler.

ing, but rather complementing each other.

Microsoft Academic. Searching for feeds from publishers (e.g., ACM and IEEE) and using webpages indexed by Bing is also considered by Microsoft Academic (MA) to collect entities such as paper, author, and venue, to be added to the MA graph (Sinha et al., 2015). An edge in the graph is added between two entities if there is a relationship between them, e.g., *publishedIn*. In contrast, in our framework, we collect not only the intended paper for a title search, but also all papers that are found for that search. In addition, we identify author homepages through author name search and, unlike MA, we use them to collect research papers from these homepages. To our knowledge, we are the first to use "Web Search" based on author names to obtain seed URLs for initiating crawls to acquire documents in scientific portals. Both our framework and MA use Bing for searches. Thus, using MA strategy to collect paper entities, $32,565$ papers are recovered out of the $43,496$ original titles. Adding the author search in our framework, we are able to collect an additional $1,439$ (=$34,004 - 32,565$) papers from the original titles and $234,312$ (=$266,877 - 32,565$) overall additional unique papers (see Table 6).

5 Related Work

Web crawling is a well-studied problem in information retrieval, focusing on issues of scalability, effectiveness, efficiency, and freshness (Manning et al., 2008). Despite its simplicity, research has shown that breadth-first search crawling produces high-quality collections in early stages of the crawl (Najork and Wiener, 2001). Focused crawling was introduced by Chakrabarti et al. (1999) to deal with the information overload on the Web in order to build specialized collections focused on specific topics. Since the introduction of focused crawling, many variations have been proposed (Menczer et al., 2004). In contrast to focused crawling, our framework is able to acquire research documents that are not limited to a specific taxonomy.

Prior research has also focused on enhancing digital libraries content to better satisfy the needs of digital library users (Pant et al., 2004; Zhuang et al., 2005; Carmel et al., 2008). Several works studied the coverage in scientific portals such as Microsoft Academic, Google Scholar, Scopus and the Web of Science (Hug and Brändle, 2017; Harzing and Alakangas, 2017). Multiple works focus on better ranking of retrieved documents for a given query (Yang et al., 2017; MacAvaney et al., 2019; Boudin et al., 2020).

Homepage finding and document classification are well-studied in information retrieval. The homepage finding track in TREC 2001 resulted in various machine learning systems for finding homepages (Xi et al., 2002; Upstill et al., 2003; Wang and Oyama, 2006). Tang et al. (2007) and Gollapalli et al. (2015) treated homepage finding as a binary classification task and used various URL and webpage content features for classification. In the context of scientific digital libraries, document classification into classes related to subject-topics (for example, "machine learning," "databases") was studied previously (Getoor, 2005; Caragea et al., 2015). In contrast with existing work, we investigate features from Web search engine results and formulate researcher homepage identification as a learning to rank task. In addition, we are the first to interleave various components of Web search, crawl, and document processing to build an efficient paper acquisition framework.

6 Conclusion and Future Directions

We proposed a framework for automatically acquiring research papers from the Web. We showed the experiments illustrating the state-of-the-art performance for two major modules of our framework: a homepage identifier and a paper classifier. Through an experiment using a large collection of $\approx 76,000$ queries (titles + authors names), our framework was able to automatically acquire an overall collection of $\approx 267,000$ unique research papers and was able to recover 78% of the original searched titles, i.e., $\approx 34,000$ papers from the $43,496$ original searched titles. We also showed that our approach is not meant to replace existing crawling strategies, but can be used in conjunction, to enhance the content of digital libraries.

In the future, it would be interesting to apply our framework to other domains, and study the integration of topic classification.

References

Krisztian Balog, Maarten De Rijke, et al. 2007. Determining expert profiles (with an application to expert finding). In *IJCAI*, volume 7, pages 2657–2662.

Chandra Bhagavatula, Sergey Feldman, Russell Power, and Waleed Ammar. 2018. Content-based citation recommendation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 238–251, New Orleans, Louisiana. Association for Computational Linguistics.

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc.

Florian Boudin, Ygor Gallina, and Akiko Aizawa. 2020. Keyphrase generation for scientific document retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Andrei Broder. 2002. A taxonomy of web search. In *ACM Sigir forum*, volume 36, pages 3–10. ACM New York, NY, USA.

Cornelia Caragea, Florin Adrian Bulgarov, and Rada Mihalcea. 2015. Co-training for topic classification of scholarly data. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 2357–2366.

Cornelia Caragea, Jian Wu, Sujatha Das Gollapalli, and C Lee Giles. 2016. Document type classification in online digital libraries. In *AAAI*, pages 3997–4002.

David Carmel, Elad Yom-Tov, and Haggai Roitman. 2008. Enhancing digital libraries using missing content analysis. In *Proceedings of the 8th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '08, pages 1–10.

Soumen Chakrabarti, Martin van den Berg, and Byron Dom. 1999. Focused crawling: A new approach to topic-specific web resource discovery. In *Proceedings of the Eighth International Conference on World Wide Web*, WWW '99, pages 1623–1640. Elsevier North-Holland, Inc.

Wang Chen, Hou Pong Chan, Piji Li, and Irwin King. 2020. Exclusive hierarchical decoding for deep keyphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1095–1105, Online. Association for Computational Linguistics.

Wang Chen, Yifan Gao, Jiani Zhang, Irwin King, and Michael R. Lyu. 2019. Title-guided encoding for keyphrase generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6268–6275.

Lise Getoor. 2005. Link-based classification. In *Advanced methods for knowledge discovery from complex data*, pages 189–207. Springer.

Sujatha Das Gollapalli, Cornelia Caragea, Prasenjit Mitra, and C Lee Giles. 2015. Improving researcher homepage classification with unlabeled data. *ACM Transactions on the Web (TWEB)*, 9(4):1–32.

Sujatha Das Gollapalli, Prasenjit Mitra, and C. Lee Giles. 2012. Similar researcher search in academic environments. In *Proceedings of the 12th ACM/IEEE-CS Joint Conference on Digital Libraries, JCDL '12, Washington, DC, USA, June 10-14, 2012*, pages 167–170.

Anne-Wil Harzing and Satu Alakangas. 2017. Microsoft academic: is the phoenix getting wings? *Scientometrics*, 110(1):371–383.

Sven E Hug and Martin P Brändle. 2017. The coverage of microsoft academic: Analyzing the publication output of a university. *Scientometrics*, 113(3):1551–1571.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Steve Lawrence. 2001. Free online availability substantially increases a paper's impact. In *Nature*, 411 (6837), pages 521–521.

Huajing Li, Isaac G Councill, Levent Bolelli, Ding Zhou, Yang Song, Wang-Chien Lee, Anand Sivasubramaniam, and C Lee Giles. 2006. Citeseerχ: a scalable autonomous scientific digital library. In *Proceedings of the 1st international conference on Scalable information systems*, pages 18–es.

Tie-Yan Liu. 2011. *Learning to rank for information retrieval*. Springer Science & Business Media.

Sean MacAvaney, Andrew Yates, Arman Cohan, and Nazli Goharian. 2019. Cedr: Contextualized embeddings for document ranking. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1101–1104.

Christopher D Manning, Hinrich Schütze, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*. Cambridge university press.

Filippo Menczer, Gautam Pant, and Padmini Srinivasan. 2004. Topical web crawlers: Evaluating adaptive algorithms. *ACM Trans. Internet Technol.*, 4(4):378–419.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592. Association for Computational Linguistics.

Marc Najork and Janet L. Wiener. 2001. Breadth-first crawling yields high-quality pages. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 114–118.

José Luis Ortega, Isidro Aguillo, and José Antonio Prieto. 2006. Longitudinal study of content and elements in the scientific web environment. *Journal of Information Science*, 32(4):344–351.

Gautam Pant, Kostas Tsioutsiouliklis, Judy Johnson, and C Lee Giles. 2004. Panorama: extending digital libraries with topical crawlers. In *Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*, pages 142–150. ACM.

Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-june Paul Hsu, and Kuansan Wang. 2015. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th international conference on world wide web*, pages 243–246. ACM.

Amanda Spink and Bernard J Jansen. 2004. *Web search: Public searching of the Web*, volume 6. Springer Science & Business Media.

Jie Tang, Duo Zhang, and Limin Yao. 2007. Social network extraction of academic researchers. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 292–301. IEEE.

Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 990–998.

Trystan Upstill, Nick Craswell, and David Hawking. 2003. Query-independent evidence in home page finding. *ACM Transactions on Information Systems (TOIS)*, 21(3):286–313.

Ji Wan, Pengcheng Wu, Steven C. H. Hoi, Peilin Zhao, Xingyu Gao, Dayong Wang, Yongdong Zhang, and Jintao Li. 2015. Online learning to rank for content-based image retrieval. In *IJCAI*.

Yuxin Wang and Keizo Oyama. 2006. Web page classification exploiting contents of surrounding pages for building a high-quality homepage collection. In *International Conference on Asian Digital Libraries*, pages 515–518. Springer.

Wensi Xi, Edward Fox, Roy Tan, and Jiang Shu. 2002. Machine learning approach for homepage finding task. In *String Processing and Information Retrieval*, pages 169–174. Springer.

Peilin Yang, Hui Fang, and Jimmy Lin. 2017. Anserini: Enabling the use of lucene for information retrieval research. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1253–1256.

Ding Zhou, Shenghuo Zhu, Kai Yu, Xiaodan Song, Belle L Tseng, Hongyuan Zha, and C Lee Giles. 2008. Learning multiple graphs for document recommendations. In *Proceedings of the 17th international conference on World Wide Web*, pages 141–150.

Ziming Zhuang, Rohit Wagle, and C Lee Giles. 2005. What's there and what's not?: focused crawling for missing documents in digital libraries. In *Digital Libraries, 2005. JCDL'05. Proceedings of the 5th ACM/IEEE-CS Joint Conference on*, pages 301–310. IEEE.