

Searching Online Book Documents and Analyzing Book Citations

Zhaohui Wu[†], Sujatha Das[†], Zhenhui Li[‡], Prasenjit Mitra^{†‡}, C. Lee Giles^{†‡}

[†]Computer Science and Engineering, [‡]Information Sciences and Technology
Pennsylvania State University, University Park, PA 16802, USA

{zzw109, gsdas}@psu.edu, {jessieli, pmitra, giles}@ist.psu.edu

ABSTRACT

Academic search engines and digital libraries provide convenient online search and access facilities for scientific publications. However, most existing systems do not include books in their collections although several books are freely available online. Academic books are different from papers in terms of their length, contents and structure. We argue that accounting for academic books is important in understanding and assessing scientific impact. We introduce an open-book search engine that extracts and indexes metadata, contents, and bibliography from online PDF book documents. To the best of our knowledge, no previous work gives a systematical study on building a search engine for books.

We propose a hybrid approach for extracting title and authors from a book that combines results from CiteSeer, a rule based extractor, and a SVM based extractor, leveraging web knowledge. For “table of contents” recognition, we propose rules based on multiple regularities based on numbering and ordering. In addition, we study bibliography extraction and citation parsing for a large dataset of books. Finally, we use the multiple fields available in books to rank books in response to search queries. Our system can effectively extract metadata and contents from large collections of online books and provides efficient book search and retrieval facilities.

Categories and Subject Descriptors

H.3.7 [Information Storage and Retrieval]: Digital Libraries; I.7.5 [Document and Text Processing]: Document Capture—*Document analysis*

General Terms

Measurement, Experimentation, Algorithms

Keywords

Book Search, Book Structure Extraction, Book Citation Analysis

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

DocEng '13, September 10–13, 2013, Florence, Italy.

Copyright 2013 ACM 978-1-4503-1789-4/13/09 ...\$15.00.

<http://dx.doi.org/10.1145/2494266.2494282>.

1. INTRODUCTION

Many challenges arise from electronic publishing. Not only is more publishable content being released digitally and made available online, but several printed books are being digitized. *Thus, what is the best way to organize and access information from online books.* Google Books¹ has become the leading project for this. In 2010, Google estimated that since the invention of printing, approximately 130 million unique titles had been published². In addition, we’ve seen recent initiatives such as the Gutenberg³, OCA (Open Content Alliance)⁴, and Million Book Project⁵. Although these projects have made progress in indexing the increasing number of online books, there are several challenges yet to be addressed.

Techniques for accurately extracting metadata from books enable better search. Machine learning approaches have been shown to be effective for extracting metadata from scientific papers and office documents [11, 13]. However, the metadata in books is in diverse formats and is different and typically richer than that in scientific papers. Therefore, novel features and techniques are required to address metadata extraction from books.

Another interesting difference between scientific papers and books is the table of contents or ToC. The ToC of a book concisely captures the logical structure of a book. Accurate extraction of the table of content is a challenging task for book retrieval systems. ToC recognition was previously studied to enable inside book search and navigation [7, 21]. However, they assume entries at the same level in a ToC share consistent features and each entry can be matched to the related title in the body part. We found those assumptions do not always hold in large diverse book datasets, while more common properties are regularities of the numbering, ordering and indentation.

In addition to the above differences, books and scientific papers differ with respect to their bibliographic references. The bibliographic formats and layouts are more varied in books than those in scientific papers. In addition, unlike papers where references typically are found at the end, a bibliography for a book could appear in various locations such as the end of each chapter, the end of the book, or

¹<http://books.google.com/>

²“Books of the world, stand up and be counted! All 129,864,880 of you.” Google Books Search. August 5, 2010.

³<http://www.gutenberg.org/>

⁴<http://www.opencontentalliance.org/>

⁵<http://archive.org/details/millionbooks>

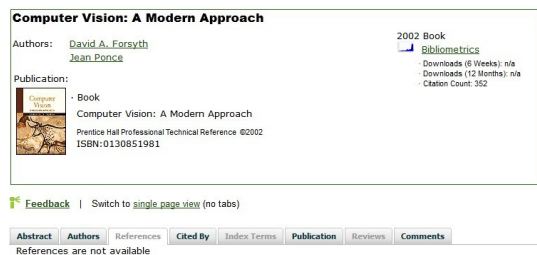


Figure 1: The absence of book references

before a back-of-the-book index. Furthermore, there is no standard citation format.

Book citations [18, 19] have been studied but not on a large scale dataset. Google Scholar recently integrated Google Books data to provide a more complete citation graph that now includes books. Thus, it is now possible to find the references of a book during a book search. In addition, Thomson Reuters has released their Book Citation Index, giving researchers access to the citation network between books and the wider world of scholarly and scientific research and full bibliographies from books and book chapters⁶. However, to the best of our knowledge, there are no academic search engines or digital libraries that provide a citation list of a book that enables navigation to the sources cited in a book, even though this facility is typically available for papers (for example, in CiteSeer [9]). As an example of what happens with prominent publishers, the ACM Digital Library does not include references for books, as shown in Figure 1.

This paper introduces a search engine for online books. These books are taken from PDF files crawled from open resources on the web that our crawler believes to be books. We create rules based on ISBN, table of contents and number of pages to identify books among other crawled PDF documents. Next, we design extraction techniques to harvest metadata such as title, authors, ISBN, etc., as well as table of contents and bibliography. To effectively extract title and authors, we devise a novel hybrid approach based on an ensemble method which entails voting from multiple sources, including CiteSeer metadata, a rule based extractor derived from sampled books, and a SVM based extractor. The ground truth of the SVM based extractor is queried from a Web knowledge base (Google Books). We also propose techniques for extracting the table of contents and bibliography in order to gain a better understanding of a book’s structure and bibliographical roles. These table of contents and chapter titles are indexed for use in inside book search. Using these indexed fields, we design an efficient ranking model for better ranking results for book search.

We demonstrate experimentally that our system can effectively extract metadata and table of contents at a large scale. Using our techniques, we were able to construct a book citation dataset containing the bibliography of 28,714 books with more than 2 million citations mentions⁷. Our citation analysis using this bibliographic data indicates that book citations are valuable and should be considered in research and scholarly assessment.

⁶http://wokinfo.com/products_tools/multidisciplinary/bookcitationindex/

⁷The book citation dataset is online and will be provided upon request

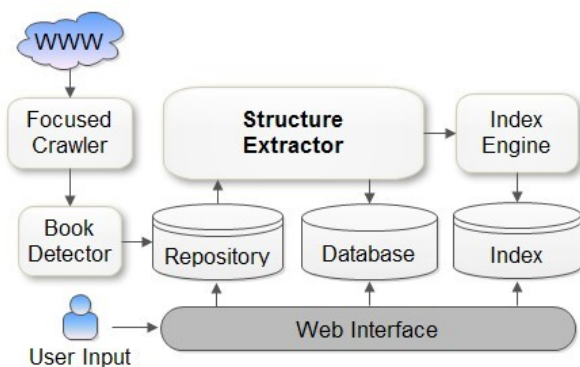


Figure 2: Architecture of the search engine demo

The rest of the paper is organized as follows. Section 2 introduces the architecture of the system, consisting of four major components: crawler and book detector, structure extractor, index engine, and web interface. Section 3 discusses technical details of each component. Section 4 describes experimental evaluations and analysis. Section 5 studies the related work. Finally, we present conclusions and future work in section 6.

2. ARCHITECTURE

We first briefly introduce the system architecture in this section, shown in Figure 2. The focused crawler, namely the CiteSeerX crawler⁸, has crawled nearly 3 million PDF documents for CiteSeerX⁹. It crawls online documents based on black and white URL lists [28]. The book detector filters potential books from crawled PDF documents and then copies those book PDF documents to our system’s repository. The structure extractor extracts the metadata, ToC and bibliography information, stores important metadata information such as title, authors and ISBNs in database. and sends the books’ indexed fields (including title, authors, chapter titles, table of contents, etc.) to an index engine. The index engine manages the index by providing add, update, delete functions. The web interface is provided for user input and shows ranked retrieved results (SERP).

The popular open source enterprise search platform Apache Solr¹⁰ is used as the index engine and apache tomcat¹¹ as the web server. A working system can be visited online¹², running on a Red hat Enterprise Linux Server.

3. IMPLEMENTATION

3.1 Crawler and Book Filter

The crawler focuses on online PDF, PS, and compressed files found from a white list containing over 700,000 URL seeds, covering many academic and research institutes all over the world [28]. Quality of seeds is evaluated by multiple factors such as number of online documents, rate of generating new documents, and the academical reputation or rank

⁸<http://csxstatic.ist.psu.edu/about/crawler>

⁹<http://citeseerx.ist.psu.edu/>

¹⁰<http://lucene.apache.org/solr/>

¹¹<http://tomcat.apache.org/>

¹²<http://sundance.ist.psu.edu:8080/solr1/index.html>

of a URL's institute. The crawler periodically updates its white list based on historical evaluations. It also contains a black list of less than 1000 URLs. Since the crawler maintains a huge white list, it limits its crawling depth to 2 to avoid unnecessary crawling of other sites. Currently, it has crawled nearly 3 million unique online PDF documents.

The book detector filters books from the crawled documents. There is no exact definition of a book. An ISBN number certainly can be regarded as a valid indicator. We thus informally define a book as any document satisfying the following three rules: 1) there exists table of contents in its first 20 pages; 2) there is a valid ISBN number in some page before the table of contents; 3) the number of its total pages is larger than 100. Unfortunately, the strong ISBN based rule found only 4905 books. However, several online books are unofficial copies provided by their authors without ISBNs. To cover those books, we weaken our constraints to only 1) and 3). Using rule 3) can retrieve 196,425 documents while adding rule 1) makes the number decrease to 73,982. These 73,982 documents are then stored in the repository as book candidates for further processing.

3.2 Extractor

We note that a common structure for a digital book includes three parts: front material, body and back material. The front material usually consists of frontispiece, title page, copyright page, table of contents, list of figures, list of tables, dedication, acknowledgments, foreword, preface and introduction. The body refers to the text or contents and is often divided into chapters. The back material contains appendix, glossary, index, notes, bibliography and colophon¹³. Our structure extractor is responsible for extracting metadata information and hierarchical structure. The metadata includes title, authors, ISBN, publish date and copyright. Since ISBN, date, copyright can be detected using strong rules, our main focus is on title and authors extraction. We extract table of contents to represent the hierarchical logical structure of a book. We also extract references either at the end of a book or the end of each chapter for book citation analysis.

3.2.1 Metadata Extraction

ISBNs can be readily detected by just searching for the string "ISBN". A sequence of digits following "ISBN" can be interpreted as the ISBN number. However, this might be incorrect if "ISBN" is not in the right page. For example, we found ISBNs of other books appear in the body of a book. To make sure the matched ISBN is valid for the book, we only search for ISBN patterns in the first 8 pages. The ISBN patterns include two types, one for a 10-digit ISBN and another for 13. As regular expressions, it appears as: `'i\s?s\s?b\s?n(10|[\s-]10)?[: -]?[\s]{0,5}([\dx-]{13})'` `'i\s?s\s?b\s?n(13|[\s-]13)?[: -]?[\s]{0,5}([\dx-]{17})'` When compiling or searching, we use the ignorecase mode.

Title and authors of books can come from three sources. First, we import the existing title and authors information from CiteSeerX database. However, that accuracy cannot be guaranteed since those are extracted by a metadata parser trained for scholarly papers, whose titles and authors can be quite different from those of books. However, since that metadata is available, it can be used as an initial metadata candidate.

¹³http://en.wikipedia.org/wiki/Book#Digital_format

Second, we develop new title and authors extractors based on heuristic rules derived from a small sample of books. We assume that title and authors are always on the same page, i.e. the title page, and the title page is before ToC, foreword or preface. We limit the title page candidates to a range from page 1 to the first page having table of contents, foreword or preface. If no such page is found, we use a page 1 to 10 as the default range. For each title page candidate, we then extract title or authors candidates based on multiple heuristics using font size, layout, length, and number of occurrences. A title or authors candidate is a visual text block containing several continuous lines without a newline break and significant font size change. In each page, the block with the maximum font size will be selected as a title candidate; the previous block and the next 2 blocks of the title candidate will be selected as authors candidates. In addition, the first block and the last block in each page are considered as authors candidate. Finally, we select the title candidate with most occurrences as the title. If there is a tie, then choose the one with larger font size. For all author candidates that start with 'By' or 'Edited by' is considered as an authors block. Otherwise, we choose the candidate which contains most naming words by looking it up in an external name dictionary containing 159,291 names.

Third, we harvest title and authors from Google Books through its API using ISBN search¹⁴, which we believe is the best source of ISBN data. Querying using the 4905 ISBNs other well known sources, including Abebooks¹⁵, Amazon book¹⁶, ISBNSearch¹⁷, and BookFinder4U¹⁸, we retrieved 814, 1275, 1170, 1301 books respectively while Google Books returned 4329 books with valid title and authors and which covered all others. The first four pages of these 4329 books are then extracted line by line and represented by features shown in Table 1. The lines with text content matched to the title and authors are labeled class '1' and class '2' respectively; other lines are labeled class '0'. By ruling out those not being successfully extracted or perfectly matched, we finally have 2496 books we consider as ground truth. We use Libsvm [1] to train a 3-class model on all the lines extracted from the 2496 books and apply it to all other books without title and authors.

Finally, the title and authors information of a book not in ground truth is identified based on "vote" from the above three sources. If more than one of them agree on the title T or author A, then T will be set as the title, or A will be one of the author. However, if all of them disagree, we simply choose the results from the third source.

3.2.2 Table of Contents Extraction

In general, to effectively extract the ToC from a document, three sub-tasks need to be addressed: ToC detection, parsing and linking [14, 23]. ToC detection attempts to locate the boundary of the ToC, usually based on explicit heuristics. ToC parsing extracts the semantics and the hierarchy of the ToC, after which the ToC will be interpreted as a tree

¹⁴<https://www.googleapis.com/books/v1/volumes?q=isbn:ISBN&key=API Key>

¹⁵www.abebooks.com/servlet/SearchResults?isbn=ISBN

¹⁶www.amazon.com/gp/search/ref=sr_adv_b/?field-isbn=ISBN

¹⁷<http://www.isbnsearch.org/isbn/ISBN>

¹⁸www.bookfinder4u.com/IsbnSearch.aspx?mode=direct&isbn=ISBN

Table 1: Features used in book metadata extraction (all feature values are rescaled to [0, 1] for training)

Feature	Description	Value type
font size	<i>Initial Font</i> : the font size of the starting character	float
	<i>Average Font</i> : the average font size of all the characters	float
	<i>Font Changes</i> : number of changes in font size	int
location	<i>Start X, End X, Start Y, End Y</i> : the coordinates of the line block in the page	float
	<i>Line Number</i> : the (order) number of the line within the page, e.g. 2 indicates the second line	int
	<i>Page Number</i> : the (order) number of the page	int
text	<i>Bag-of-words</i> : Top 200 words selected by DF rank in the whole dataset; 1 indicates a word is in the line	boolean
others	<i>Number of Words</i> : the total number of words in the line	int
	<i>Number of Digits</i> : the total number of digital words in the line	int

where each node represents an entry in the ToC. ToC linking determines the corresponding content in the body text w.r.t each entry in the ToC. We summarize the challenges of ToC recognition as follows by examining our book dataset. First, there are various types of ToC, making it difficult to find universal rules or templates governing all possible ToCs. For example, a ToC could be a full page, multiple pages, or part of a page; some documents may have multiple ToC (one for the whole document and one per chapter). Second, a ToC might contain noisy or inconsistent content. For example, there may exist some decorative content within a ToC page, or some entries of a ToC may be in multiple lines whose styles are inconsistent. Third, text in a ToC does not necessarily contain the exact title of the body sections. However, previous work assumes entries of the same level share consistent features and every entry can always be linked to the related title in the body part [7, 21].

Our ToC recognition is based on the following rules: 1) a ToC is generally in the first few pages of the document; 2) a ToC usually contains some regularities of numbering and indentation; 3) a ToC generally contains ordered references correlated (but not exactly matched) to titles or sections in body pages. The last property can also be broken into 5 sub-properties: 1) contiguity: a ToC consists of a series of contiguous references to some other parts; 2) ordering: the references and the referred parts appear in the same order in the document; 4) no self-reference: all references refer outside the contiguous list of references; 5) distinctness: the link from the references of ToC to the outside parts is injective, or every reference refers to a distinctive part. Our method does not rely on visual features such as font size or layout so that we can do the detection purely based text, which is much more efficient for large scale extraction.

3.2.3 Bibliography Extraction

Bibliography usually has obvious indicators such as “References”, “Bibliography” or “Sources”. However, unlike papers, books may have a bibliography at the end of each chapter. Thus, we need to search bibliography in the whole body of book rather than in only the last few pages. If we find a line contains only one of the three keywords and the lines followed are ordered reference items, we identify it as a bibliography block. We search the ordered number at the beginning of each reference until there are no continuously increasing number found in the following 30 lines. 30 seems like a large distance for references. But we do find some references contained near 10 lines. Also we believe that the distance between two bibliography blocks in two chapters will be much larger than 30. All the bibliographic files are

Table 2: Rules for generating venue alias

Rule	Examples of Venue Name
None	IEEE Transactions on Pattern Analysis and Machine Intelligence
Transactions->Trans. Journal->J Proceedings->Proc.	IEEE Trans. on Pattern Analysis and Machine Intelligence
Remove "of", "on", "in", "the"	IEEE Trans. Pattern Analysis and Machine Intelligence
Acronymization	IEEE Trans. PAMI
Pure acronymization	PAMI
Manual edit	IEEE Trans. Pattern Anal. Machine Intell.

extracted using the above heuristics from the pre-extracted text files.

We assign each book a unique document ID for our system. Our bibliography extractor successfully extracted bibliographic files from 28,714 PDFs in the whole book document collection containing 73,982 PDFs. The total number of reference mentions is 2,501,497. The other documents include unextracted PDF documents whose bibliography consisted of references without order number. We checked a small sample of these and found all of them not in computer science. The dataset contains all bibliographic files, where the file name responds to a book ID. In a bibliographic file, every two continuous references are separated by a new line.

3.2.4 Citation Analyzer

The citation analyzer has four main functions, including reference parsing, citation normalization, citation counting, and literal errors handling.

Reference parsing

It parses all references using ParsCit [2] and then improves the parsed results using an external name dictionary of authors and a thesaurus of venue names. Author names are collected from CiteSeerX database while thesaurus of venue is constructed based on rules and manual editing. Table 2 shows the rules using “IEEE Transactions on Pattern Analysis and Machine Intelligence” as an example. The acronymization keeps the prefix part such as “IEEE Transactions on”, “Journal of”, and “Proceedings of”, while pure acronymization does not, as shown in the fourth row and fifth row. Manual editing refers to an unusual venue alias found from the data which are added to the thesaurus manually.

Citation Normalization

Citations to the same source may have widely varied formats including various placement and presentation of author names, venues, and dates. Sometimes there are even errors. As such it is necessary to normalize various citations to the same work, especially to count the number of citations to a given work. The normalization is based on matching authors, title and venue, depending primarily on

Table 3: Common literal due to PDF extraction

Literal type	Example
Absence of the initial	heoretical, bject-Oriented, ommunication
Split of one word	Springer- Verlag, L ATEX, MEM- OIRS
Corrupted string	Int\\\\\\\\\\ Conf., O\\\\\\Reilly

the correctness of title. That is to say, two citations will be marked as the same source if they have exact the same title. Otherwise, if the edit distance of the two titles is less than a threshold, say, 5% of the max length of the two titles, we check the similarity of authors and venues. Citations without titles are not included for now.

Citation counting

The primary statistic we explore is the number of “cited by” books of works, venues, and institutes, which we further employ as a metric to measure their impact. Simply, if there are 100 books citing works from an institute, the number will be 100. The other one is the total number of citations to an institute, a venue and a work. It will always be larger than the former since there can be more than one citations in a book to an institute, a venue and a work. The two statistics are strongly dependent on the quality of reference parsing and citation normalization.

Literal errors handling

A problem cannot be neglected is the unpredictable literal errors generated during the PDF extraction. We list the three most common literal types found in our dataset in Table 3. For a word in “absence of the initial” or “split of one word”, we find the closest “clean” reference to the reference it appears in based on edit distance and then correct it to the counterpart word in the “clean” reference. We do the same processing in the case when there is “corrupted string” in only a single word of a reference and choose to ignore it if there are more than two words corrupted.

3.3 Indexing and Ranking

We show the index fields and types in Table 4, which are configured in the Solr schema.xml¹⁹. The field “id” is defined as the *uniqueKey*; “contents” refers to the table of contents; “body” is the full text except the bibliography; “links” indicates the URL where the document comes from; “googleID” refers to the unique ID of the book in Google Books database, based on which we can get the page of the book in Google Books search. The “text” field is set to be the *defaultSearchField*. The default search field “text”, contains “id”, “title”, “authors”, “chapter_title”, “contents” and “body”. This setting enable us to search insides a book. We use the simple TFIDF based relevance model to rank the returned book list of a given query based on the weighted averaged score on all the *defaultSearchFields*.

$$s(q, f_i) = w(q, f_i) \cdot N(q) \cdot \sum_{t \in q} (tf(t, f_i) \cdot idf(t)^2 \cdot norm(t, f_i))$$

where $tf(t, f_i)$ correlates to the term’s frequency, defined as the number of times term t appears in the field f_i ; $idf(t)$ stands for Inverse Document Frequency; $w(q, f_i)$ is a score factor based on how many of the query terms are found in the specified field; $N(q)$ is a normalizing factor used to make scores between queries comparable; $norm(t, d)$ encapsulates a few (indexing time) boost and length factors.

¹⁹<http://wiki.apache.org/solr/SchemaXml>

Table 4: Index fields

Name	Type	Stored	MultiValued
id	string	true	false
title	text_general	true	true
chapter_title	text_general	true	true
contents	text_general	true	false
body	text_general	true	false
isbn	text_general	true	false
authors	text_general	true	false
publish_date	string	true	false
links	string	true	true
googleID	string	true	false
text	text_general	false	true

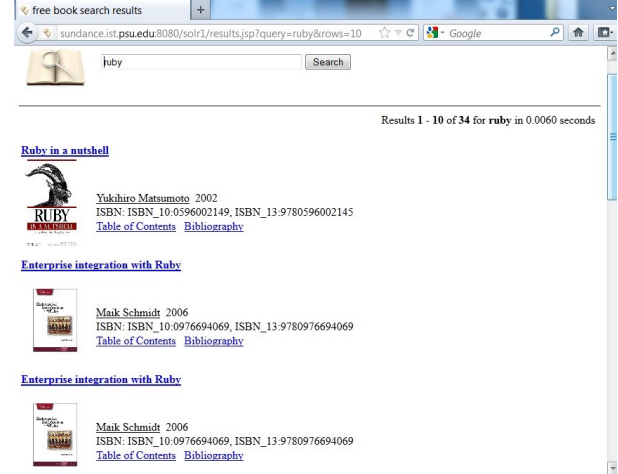


Figure 3: Results of searching “ruby”

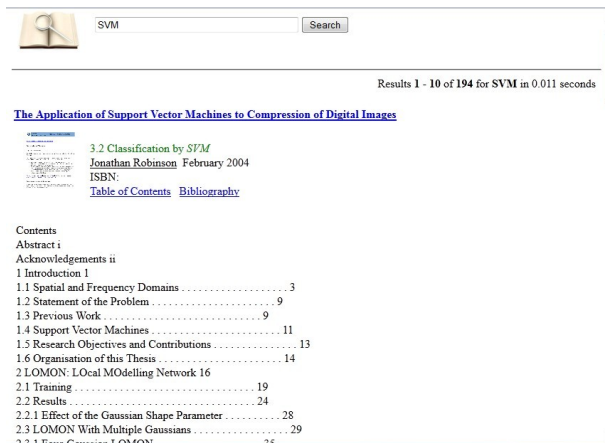
3.4 Interface

We provide a simple style search interface. The searching result page is shown in Figure 3, where each block consists of title of the book, the book cover, the most relevant chapter title, authors, ISBN, table of contents, and bibliography. Each page contains 10 blocks, showing the top 10 returned results. The title links to a download page of the book while the book cover links to a html version of the book which can provide navigation using page numbers. The html version of a book is converted using PDFMiner²⁰. There are 27,637 books with HTML copies. In Figure 4, there are no chapter titles shown in the top 3 results of searching “ruby”, but we can see “3.2 classification by SVM” as the most relevant chapter title. It also shows the table of contents in a dynamic html table, providing a quick glance for the overall content and structure of a book. In the future we will build a link between table of contents and the html pages. Then by clicking an item in the table of contents, one can go directly to the page of a book.

4. EVALUATION

We first evaluate the extractors and the search engine. The extractors were evaluated on small labeled books randomly sampled from the whole book collection. For the later, human evaluation on different queries based on mul-

²⁰<http://www.unixuser.org/~euske/python/pdfminer/index.html>



Results 1 - 10 of 194 for SVM in 0.011 seconds

[The Application of Support Vector Machines to Compression of Digital Images](#)

3.2 Classification by SVM
Jonathan Robinson February 2004
ISBN:
[Table of Contents](#) [Bibliography](#)

Contents
Abstract i
Acknowledgements ii
1 Introduction 1
1.1 Spatial and Frequency Domains 3
1.2 Statement of the Problem 9
1.3 Previous Work 9
1.4 Support Vector Machines 11
1.5 Research Objectives and Contributions 13
1.6 Organisation of this Thesis 14
2 LOMON: Local Modelling Network 16
2.1 Training 19
2.2 Results 24
2.2.1 Effect of the Gaussian Shape Parameter 28
2.3 LOMON With Multiple Gaussians 29
2.3.1 Four Gaussians LOMON 35

Figure 4: Example of table of contents

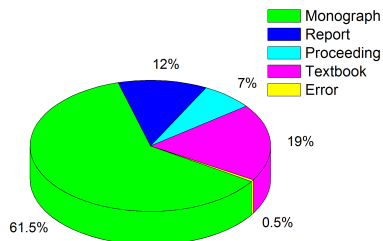


Figure 5: Components of the books

title metrics is presented. In addition, citation analysis is studied based on the constructed book citation dataset.

4.1 Components of Books

Ideally, we hope there is a valid ISBN for every book in the dataset. However, ISBN is detected in only less than 5000 books. Note sometimes ISBN appears as a picture bar-code which cannot be detected. We then manually check 200 books randomly sampled from the whole set. We classify them into 5 categories, including monograph, report, proceeding and textbook. A monograph is a detailed scholarly work of a single specialized subject. Usually it is derived from a Phd thesis or multiple research papers. Since an official Phd thesis also has an ISBN, we count a Phd thesis as a monograph. Reports refer to unpublished book-like documents from a university, government, or company. Proceedings are paper collections from a conference, workshop or journal. Since proceedings usually have an ISBN, we do not rule them out. Textbooks here represent the more traditional books including course books, reference books, and manuals. The type of books is shown in Figure 5. Error indicates one corrupted PDF filled with error codes. If we count textbook and monograph as appropriate books, we get at least 80% of those in our book citation dataset.

4.2 Extraction Evaluation

We list our evaluation results for each extractor in Table 5. The 5-fold cross validation in the 2496 books for the SVM based extractor achieves an classification accuracy of 89%.

Table 5: Experimental evaluation for the extractors

Extractor Name	dataset size	precision	recall
Title	100	87%	89%
Authors	100	90%	92%
Table of Contents	147	85%	90%
ISBN	300	99%	97%
publish date	300	95%	90%
Copy right	300	92%	94%
Bibliography	200	96%	93%

The rule-based extractor achieves precision of 78% and recall of 81% on the 2496 books. To gain better performance on the whole data collection, we train the SVM model on the whole 2496 books. A small manually labeled set (out of the 2496 books with ground truth) of size 100 is used for testing. For table of contents extraction, it uses another dataset containing 147 books with table of contents manually labeled. The reason we choose different datasets for different extractors is some books do not have all of this information. For example, some books with a valid ISBN do not have a ToC. ISBN, copy right and publish date are easier tasks based on the same dataset. As we can see from the results, rule-based extractors can achieve an acceptable accuracy for most metadata. However, title and authors of more varieties are much more difficult than those with notable patterns. Even our hybrid approach based on the existing rule-based extractor and SVM-based extractor does not achieve comparable performance to other metadata extractors. We check the correctness of bibliographic files extracted from the 28,714 books. Again, we randomly sample 200 bibliographic files and check them manually. We find all the bibliographic files are correctly extracted. However, literal errors generated during PDF extraction are unavoidable. For example, one word might be split into two words and multiple words may be merged into a single one. We find 916 occurrences of “he Art of Computer Programming” and 513 occurrences of “radiate Texts in Mathematics”, where the first letter of “The” and “Graduate” is missing. It’s difficult to get the exact statistics of all these literal errors and then automatically correct them. These errors are first ignored when parsing references and calculating the statistics for each element such as work titles, venues, and institute. We then investigate the original ranking list of all these elements to find the principal heuristics that can help to rectify the missing numbers. Here, we group all titles using edit distance and then assign the longest one as the representative.

4.3 Search Evaluation

Basic statistics about the book search engine is listed in Table 6, including the total number of indexed books, the number of books with bibliographic files, the number of books with HTML copies, the number of books with more than 200 pages and the number of books with GoogleID. Metadata including title, authors, publish date of books with googleID is imported from Google Books by its API. We evaluate the relevance using the precision of the top 10 responses by manually checking a set of queries. Results of ten selected queries are shown in Table 7, based on precision, total number of returned results and response time. Since the exact number of relevant books for each query is unknown, the recall is not given. From the table, we can see that the search engine gives relevant results in the first page

Table 6: Book search engine statistics

Indicator	Number
Total indexed books	59,207
Bibliography files	28,714
HTML files	27,637
Books with 200+ pages	25,680
Books with googleID	5,945

Table 7: Searching evaluation

Query	Precision	#Results	Time
ruby	100%	34	0.005 s
python	100%	110	0.003 s
java	90%	1,467	0.004 s
matlab	100%	447	0.009 s
SQL	100%	414	0.004 s
database	100%	3,139	0.004 s
machine learning	100%	6,497	0.006 s
decision tree	50%	5,376	0.011 s
topic model	20%	18,347	0.001 s
latent dirichlet allocation	100%	2095	0.014 s

(top 10 results) for most domain-specific keywords queries. However, when a query is composed of general words, e.g. decision tree, topic model, the precision gets lower because the default operator for query parser is set to be “OR”. For query “decision tree”, the other 5 irrelevant returned results are about trees in data structure and decision in management. For “topic model”, the two words are more general than “decision” and “tree”, thus the result is even worse. However, the query operator “AND” can also be specified by set q.op=AND. A comparison to Google Books Search based on 8 different features is summarized in Table 8, including size, ranking, online preview, advance search, speed, open access, table of contents, and bibliography. For the ranking feature, our search only shows the matched results in title and chapter titles, while Google Books can give the exact matches in the pages of a book with priority to title and chapter titles. Although our search engine cannot compare to Google Books on many features, it has certain special features such as complete open access and a more complete table of contents and bibliographic information.

4.4 Overview of Book Citations

Previous work show that the most highly cited works are from books and book chapters [10, 18]. However, the value of book citations has been mostly ignored. We first investigate the number of citations from books. As shown in Figure 6, the distribution is nearly uniform around the average (87)

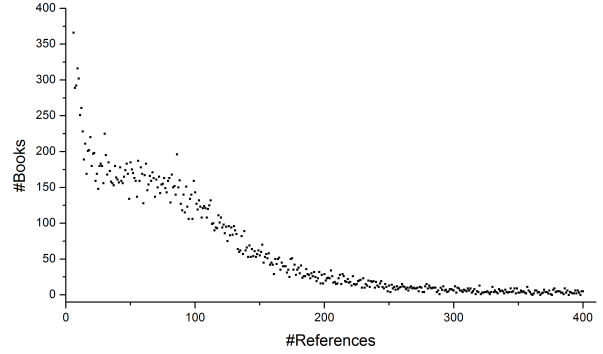


Figure 6: Distribution of number of references in books

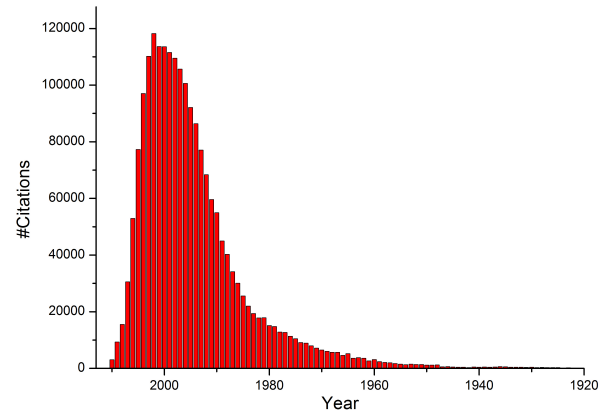


Figure 7: Number of citations each year

but steeper decreasing before 20 and after 100. Number of citations of most books (76%) range in [20, 200] while long tail appears after 200. Furthermore, we found that manual or reference books tend to have few references than academic books. The more abundant citations from books offer a wider research assessment than citation datasets of the same scale from papers. We speculate that while papers tend to cite more topic-similar peer works, books would cover a much broader literature. We believe this would make book citations a valuable source for academic assessment.

The evolving trend of number of citations each year since 1920 is shown in Figure 7. We believe the peak will keep growing in time while the left side will keep a steeper cliff shape at around 10 years. The growing trend from left to right suggests a number of publications produced each year is increasing. The steeper cliff shape implies that books tend to cite older work of several years which at the time would have been the state of art. Figure 7 may also imply the citation preference over time of a single book, considering that the number of references of most books is around the average. We conjecture that when a book is written, most works are cited in some year range. Before this year range, less will be cited as the years decrease. After that, less will be cited as the years increase. To prove or disprove this conjecture would be interesting.

4.5 Most Highly Book Cited Documents

Table 8: Comparison with Google Books

Features	The Demo	Google Books
Size	59,207	>100,000,000
Ranking	title,ch_title	title,ch_title,full text
Online preview	yes	yes
Advance search	no	yes
Speed	0.01s	0.5s
Open access	All	< 1%
Table of Contents	Yes	Yes
Bibliography	Yes	No

Table 9: Document rank based on book citations

Title	Cited by	#Citations	CiteSeer Rank	Google	ISI Search	ISI Cited
Introduction to Algorithms (B)	982	1080	2	28558	-	72
Computers and Intractability (B)	753	809	1	40578	-	4
Numerical Recipes in C: The Art of Scientific Computing (B)	676	749	127	442	-	58
The Art of Computer Programming (B)	604	1832	74	27270	-	28
Digital Image Processing (B)	566	844	-	7987	-	91
Elements of Information Theory (B)	526	563	5	26485	-	25
Partial Differential Equations (B)	518	1308	-	12404	Y	38
Graduate Texts in Mathematics (B)	513	931	-	3358	-	29
Communicating Sequential Processes (J)	474	609	15	14579	Y	1394
Genetic Algorithms in Search, Optimization and Machine Learning(B)	361	1164	4	50264	-	2
A mathematical theory of communication (J)	356	390	10	49992	Y	9800
Compilers: Principles, Techniques, and Tools (B)	356	368	-	10964	-	2
Communication and Concurrency (B)	344	379	17	8934	-	2
Design Patterns: Elements of Reusable Object-Oriented Software (B)	337	377	8	27325	-	9
Statistical Learning Theory (B)	334	427	3	37675	-	27
The C++ Programming Language (B)	332	358	119	7998	-	1
The Java Language Specification (B)	324	348	88	6430	-	2
Computer Architecture: A Quantitative Approach (B)	304	323	12	9560	-	4
Introduction to Automata Theory, Languages, and Computation (B)	296	969	14	12695	-	1
Ordinary Differential Equations (B)	274	587	-	7182	-	15

The most highly cited documents in our book dataset are listed in Table 9. In the title, “(B)” indicates a book while “(J)” indicates a journal paper. The “CiteSeer rank”²¹ represents the rank of most cited computer science citations generated from documents in CiteSeerX database in 2011. “Google” represents the “cited by” number of a document provided by Google Scholar. Since there might be duplicate results in Google Scholar, we choose the one with the highest “cited by” number. The “ISI Search” column shows whether a work is indexed by ISI, “Y” indicating yes otherwise no. If yes, the number of “ISI Cited” is the “Times Cited” number of the document in ISI search results. Otherwise, we use “Cited Reference Search” provided by ISI Web of Science to find the number of cited references²².

The first evident finding is most highly cited documents of books, as expected, are books. There are only 2 journal papers in the top 20 documents. Since a previous study showed books are favored by papers citations, we may conclude that the most highly cited documents of all research publications are books. However, from the CiteSeer rank and Google “cite by” numbers, we can see the most highly cited documents of books are different from those of papers. There are 5 books (marked by “-”) are out of top 200 in CiteSeer rank, suggesting that some books favored by books may not receive high citations from papers. A key finding here is that those highly cited books are absent in ISI, or not indexed by the Book Citation Index of ISI. By checking the “Cited Reference Search”, we find there are very few citations to those books in the ISI database. The only exception is “Partial Differential Equations”, which is searchable in Web of Science of ISI but being cited only 38 times.

4.6 Venue Rank

An key role of citation analysis is to measure importance of venues using citation based metrics [25, 26]. The impact factor (IF) [8] provided by ISI might be, if not the most satisfactory one, the most popular one for measuring journals. In a given year, the impact factor of a journal is the average number of citations received per paper published in that

journal during the two preceding years²³. Google Scholar uses *h-index*, or the largest number *h* such that at least *h* articles in that venue were cited at least *h* times each²⁴. We rank the venues based on the number of “cited by” books and list the top 20 of them in Table 10, with the IF (2009) and *h-index* shown in the fourth and fifth column for each venue. The IF value is queried using MedSci²⁵ relying on ISI and the *h-index* is from SCIMAGO²⁶ based on scopus²⁷.

The results shown in Table 10 offer several interesting observations. First, books prefer to cite more documents from journals rather than from conferences. All the top 30 venues are journals except ACM SIGCOMM. With the exception of LNCS, a book like conference proceedings as the second most highly cited venue by books, suggests that conference proceedings are not good citation sources for books. Second, the most highly cited venues are premier journals, indicating the “cited by” is a potential valuable metric for measure venues. We can see all venues are SCI indexed journals except LNCS and ACM SIGCOMM. Only 5 journals have an IF lower than 1 while half of them have an IF higher than 2. Third, there is no strong correlation between the “cited by” number and IF or *h-index*. A possible reason is our dataset is independent on ISI data or scopus data. Thus the “cited by” numbers from our book citations are missing in their statistics. For example, if they consider the 13,297 citations from 6,431 books, surely, the IF value and *h-index* of Communications of ACM will increase.

5. RELATED WORK

5.1 Book Structure Extraction

The book structure extraction attempts to harvest the logical structure of a book (pictured by table of contents) and the metadata including title, authors, ISBNs, publisher, etc. Book table of contents (ToC) recognition has been extensively studied in the document analysis and recognition

²³http://thomsonreuters.com/products_services/science/academic/impact_factor/

²⁴<http://scholar.google.com/intl/en/scholar/metrics.html>

²⁵<http://www.medsciediting.com/sciif.asp>

²⁶<http://www.scimagojr.com/journalsearch.php>

²⁷<http://www.scopus.com/home.url>

²¹<http://citeseerx.ist.psu.edu/stats/citations>

²²Our queries were conducted on May 9 2012.

Table 10: Venue rank based on book citations

Venue name	Cited by	#Citations	IF (2009)	h-index
Communications of the ACM	6431	13297	2.35	101
LNCS	3618	34740	-	75
Theoretical Computer Science	2893	10167	0.838	59
Physical Review Letters	2425	32877	7.621	349
Science	2565	6886	31.364	678
IEEE Transactions on Pattern Analysis and Machine Intelligence	2246	10274	5.027	169
IEEE Transactions on Computers	2152	4708	1.604	66
Journal of the ACM	2085	4000	3.375	72
IEEE Transactions on Software Engineering	2070	10020	2.216	88
ACM Computing Surveys	2013	6794	7.806	62
Operations Research	1691	6417	1.995	65
IEEE Transactions on Information Theory	2042	10034	2.725	155
Nature	1500	5365	36.101	698
ACM Transactions on Programming Languages and Systems	1490	2826	1.167	45
IEEE Journal on Selected Areas in Communications	1487	7498	4.232	140
Parallel Computing	1437	3489	1.086	37
Automatica	1426	2815	2.171	114
Annals of Math.	1406	2380	3.179	53
Information Sciences	1376	2220	2.833	61
ACM SIGCOMM	1322	5214	-	-

community [3, 4, 7, 21]. However, those methods based on ad hoc rules only work for a small size and domain-specific book set. It is still a challenging problem to design a ToC recognition algorithm that can be effectively applied to large scale heterogeneous books [14]. Gao et al. studied both ToC and metadata extraction from PDF book documents by modeling them as a matching problem on the bipartite graph [6]. Feng et al. studied how to restructure the OCR output of books using a Hidden Markov Model (HMM) based hierarchical alignment algorithm [5]. Metadata extraction has also been studied as an information extraction problem using classification models such as SVM for scholarly papers [11], or using sequential labeling models for general office documents [13]. While the techniques for training our SVM based extractor is similar to [11], we use a novel hybrid voting approach and our ground truth is harvested from Web.

5.2 Book Search and Retrieval

Book search and retrieval has focused on designing a better indexing to support inside book search. H. Wu et al. reported an experimental book search system that supports both database and IR style index structures. Their findings suggest that fielded retrieval is a suitable strategy to apply to collections of books [27]. W. Magdy et al. examined the effect of indexing different parts of digitized books on retrieval in response to specific information needs. These results indicate that certain portions of books, specifically titles and headers, are more valuable than other parts of books [22]. However, the evaluation of search and retrieval over large book repositories is still a difficult task. G. Kazai et al. used crowdsourcing for book search evaluation and found that well designed crowdsourcing can be an effective tool for the evaluation of book IR systems [15, 16]. A similar problem has also been tested in the social book search task [17].

5.3 Book Citation Analysis

Books and monographs play significant roles in research communication. The absence of citations from most books

and monographs from the Thomson Reuters/Institute for Scientific Information databases (ISI) has been criticized, but attempts to include citations from or to books in research evaluation in our opinion has not been that successful. Kousha and Thelwall studied the book citations analysis in science, social science, and humanities disciplines using Google Books and showed it to be a valuable new source of citation data for the social sciences and humanities[18]. They argued that in book-oriented disciplines such as the social sciences, arts, and humanities, online book citations may be sufficiently numerous to support peer review research evaluation [19]. However, to the best of our knowledge, none studied the large scale citations from books from Web. Our effort is similar to automatic citation analysis such as CiteSeer [10, 20], thought their focus is on conference and journal papers. In addition, using citation data to rank scholars or institutes has long been an important area of research [12, 24].

6. CONCLUSIONS AND FUTURE WORK

We present a search engine for online books in PDF format. The search engine explores multiple methods for navigating books and supports searches on metadata, table of contents and bibliography. We discussed techniques for extracting metadata from PDF books. In particular, we devised for extracting the title and authors information a novel hybrid approach based on a voting from multiple sources: CiteSeer metadata, a rule based extractor derived from sampled books, and a SVM based extractor learned from web knowledge. In addition, we also proposed methods for extracting the table of contents and bibliography. Using the extracted bibliography from books, we enhanced citation analysis in recent academic search engines by constructing a book citation dataset. Our initial, statistical analysis using this dataset shows that in computer science and related disciplines, books citations are valuable and should not be neglected in peer review research evaluation.

Future goals would be to further improve the accuracy of metadata and ToC information by designing better extrac-

tors and aggregating available metadata from other systems and to provide more navigation functions inside book search. For example, it would be useful to build links between ToC and the body pages and to develop metrics that capture the impact of book citations to institutes, venues and authors. Then the full scholarly document citation graph can be updated using citations to and from books.

7. ACKNOWLEDGEMENTS

We gratefully acknowledge support from the CiteseerX team, the National Science Foundation, and useful comments from referees.

8. REFERENCES

- [1] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.
- [2] I. G. Councill, C. L. Giles, and M. Yen Kan. Parscit: An open-source crf reference string parsing package. In *Proceedings of the Language Resources and Evaluation Conference*, pages 661–667, 2008.
- [3] H. Dejean and J.-L. Meunier. On tables of contents and how to recognize them. *International Journal on Document Analysis and Recognition*, 12(1):1–20, 2009.
- [4] H. D’Álmeida and J. L. Meunier. Structuring documents according to their table of contents. In *Proceedings of DocEng*, pages 2–9, 2005.
- [5] S. Feng and R. Manmatha. A hierarchical hmm-based automatic evaluation of ocr accuracy for a digital library of books. In *Proceedings of JCDL*, pages 109–118, 2006.
- [6] L. Gao, Z. Tang, X. Lin, Y. Liu, R. Qiu, and Y. Wang. Structure extraction from pdf-based book documents. In *Proceedings of JCDL*, pages 11–20, 2011.
- [7] L. Gao, Z. Tang, X. Lin, X. Tao, and Y. Chu. Analysis of book documents’ table of content based on clustering. In *Proceedings of ICDAR*, pages 911–915, 2009.
- [8] E. Garfield. Impact factors, and why they won’t go away. *Nature*, 411(6837):522–522, 2001.
- [9] C. L. Giles, K. D. Bollacker, and S. Lawrence. Citeseer: an automatic citation indexing system. In *Proceedings of the third ACM conference on Digital libraries*, DL ’98, 1998.
- [10] A. A. Goodrum, K. W. McCain, S. Lawrence, and C. L. Giles. Scholarly publishing in the internet age: a citation analysis of computer science literature. *Information Processing Management*, 37(5):661 – 675, 2001.
- [11] H. Han, C. L. Giles, E. Manavoglu, H. Zha, Z. Zhang, and E. A. Fox. Automatic document metadata extraction using support vector machines. In *JCDL*, pages 37–48, 2003.
- [12] J. E. Hirsch. Does the h index have predictive power? *Proceedings of the National Academy of Sciences*, 104(49):19193–19198.
- [13] Y. Hu, H. Li, Y. Cao, D. Meyerzon, and Q. Zheng. Automatic extraction of titles from general documents using machine learning. In *Proceedings of JCDL*, pages 145–154, 2005.
- [14] Y. Jayabal, C. Ramanathan, and M. J. Sheth. Challenges in generating bookmarks from toc entries in e-books. In *Proceedings of DocEng*, pages 37–40, 2012.
- [15] G. Kazai, J. Kamps, M. Koolen, and N. Milic-Frayling. Crowdsourcing for book search evaluation: impact of hit design on comparative system ranking. In *Proceedings SIGIR*, pages 205–214, 2011.
- [16] G. Kazai, M. Koolen, J. Kamps, A. Doucet, and M. Landoni. Overview of the inx 2010 book track: Scaling up the evaluation using crowdsourcing. In *Comparative Evaluation of Focused Retrieval*, pages 98–117. 2011.
- [17] M. Koolen, J. Kamps, and G. Kazai. Social book search: comparing topical relevance judgements and book suggestions for evaluation. In *Proceedings of CIKM*, pages 185–194, 2012.
- [18] K. Kousha and M. Thewall. Google book search: Citation analysis for social science and the humanities. *Journal of the American Society for Information Science and Technology*, 60(8):1537–1549, 2009.
- [19] K. Kousha, M. Thewall, and S. Rezaie. Assessing the citation impact of books: The role of google books, google scholar, and scopus. *Journal of the American Society for Information Science and Technology*, 62(11):2147–2164, 2011.
- [20] S. Lawrence, F. Coetzee, G. Flake, D. Pennock, B. Krovetz, F. Nielsen, A. Kruger, and C. L. Giles. Persistence of information on the web: Analyzing citations contained in research articles. In *Proceedings of CIKM*, pages 235–242, 2000.
- [21] X. Lin and Y. Xiong. Detection and analysis of table of contents based on content association. *International Journal on Document Analysis and Recognition*, 8(2):132–143, 2006.
- [22] W. Magdy and K. Darwish. Book search: Indexing the valuable parts. In *BooksOnline*, 2008.
- [23] S. Marinai, E. Marino, and G. Soda. Table of contents recognition for converting pdf documents in e-book formats. In *Proceedings of DocEng*, pages 73–76, 2010.
- [24] J. Ren and R. N. Taylor. Automatic and versatile publications ranking for research institutions and scholars. *Commun. ACM*, 50(6):81–85, June 2007.
- [25] X. Shi, J. Leskovec, and D. A. McFarland. Citing for high impact. In *Proceedings of JCDL*, pages 49–58, 2010.
- [26] Y. Sun and C. L. Giles. Popularity weighted ranking for academic digital libraries. In *Proceedings of ECIR*, pages 605–612, 2007.
- [27] H. Wu, G. Kazai, and M. Taylor. Book search experiments: Investigating ir methods for the indexing and retrieval of books. In *ECIR*, pages 234–245, 2008.
- [28] J. Wu, P. Teregowda, J. P. F. Ramírez, P. Mitra, S. Zheng, and C. L. Giles. The evolution of a crawling strategy for an academic document search engine: Whitelists and blacklists. In *Proceedings of ACM WebSci*, 2012.