The Topology of Shared Concepts in Big Content

A Thesis

Presented in Partial Fulfillment of the Requirements for the
Degree Master of Science in the Graduate School of
The Ohio State University

By

Andrew Yates, B.S.

Graduate Program in the
Department of Computer Science and Engineering

The Ohio State University

2014

Master’s Examination Committee:

Prof. Rajiv Ramnath, Advisor
Prof. Jay Ramanathan
Copyright by
Andrew Yates
2014
Abstract

Organizing large corpuses of content, or “big content,” is becoming increasingly challenging. In academia, open access publishing on the Internet continues to challenge disciplinary divisions and institutional authorities. Supervised computational techniques are ill suited to categorize such content because training sets are infeasible to produce and categorization schemes of the past fail to describe the evolving landscape of scholarly work. Inspired by this challenge, we present a method based on stochastic block modeling to produce an optimal categorization scheme for any large collection of rich content by modeling the network of shared ideas between items of content, or the “concept topology.” Our approach is unsupervised, hierarchical, and statistically robust, and it produces an optimal number of clusters and levels as an output. More practically, our method scales to large networks, which we demonstrate by compiling a corpus of every open access academic publication indexed on the Internet and clustering 1.5 million of those abstracts. In this thesis, we discuss the compilation of this corpus, describe our concept topology clustering method, and present the experimental results of applying our method to map our corpus. Since our corpus contains all published open access academic works, the resulting categorization is the best categorization scheme for publications in all of the sciences as they are freely available on the Internet.
Dedicated to Yixue Li Yates.
Acknowledgments

On its way to completion, this research project benefited from the input and feedback of many experts. In chronological order, I thank Professors Raghu Machiraju and Kun Huang for the part they played in my graduate education, and I thank Professors Jeff Parvin and Philip Payne for supporting my previous research in computational biology. Although my thesis research took a new and different direction, I learned much from them.

Later, I benefited from the unwavering support of Professors Rajiv Ramnath and Chris Bartlett. I profusely thank them and will forever be grateful to them. I also wish to extend my deep appreciation to Professor Helen Chamberlain for lending her assistance, brilliance, and solid common sense.

Friends were also there all along, when I needed them: I thank my lab mate, Michael Sharpnack, for being a supportive friend. Thank you to Shashank Agarwal for generally putting up with me.

Thank you to those who supported this research. Thank you Dave Schappell of Amazon for providing me with the computing resources needed to execute this research. Thank you to OCLC, Tip House, and Mike Teets for financial support and an introduction to the problems of organizing scholarly work in the digital age. Thanks
to Parsa Ghaffari of Aylien and Brent Walsh of AlchemyAPI for supporting this research with computational service support. Thank you to Professor Jay Ramanathan for serving on my thesis committee.

Finally, I thank my parents, Don and Jane Yates, and my wife, Yixue Li Yates. Your unwavering love and support has made all of this possible.
Vita

2011 .................................................. B.S. Computer Science and Engineering, The Ohio State University

Publications

Research Publications


Field of Study

Major Field: Department of Computer Science and Engineering
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>ii</td>
</tr>
<tr>
<td>Dedication</td>
<td>iii</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>iv</td>
</tr>
<tr>
<td>Vita</td>
<td>vi</td>
</tr>
<tr>
<td>List of Tables</td>
<td>x</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xi</td>
</tr>
<tr>
<td>1. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Main Contributions and Organization</td>
<td>2</td>
</tr>
<tr>
<td>1.1.1 Thesis Outline</td>
<td>3</td>
</tr>
<tr>
<td>2. The Open Access Academia Corpus (OAAC)</td>
<td>5</td>
</tr>
<tr>
<td>2.1 Introduction to Open Access Academic Publication</td>
<td>5</td>
</tr>
<tr>
<td>2.2 Sources</td>
<td>7</td>
</tr>
<tr>
<td>2.2.1 Primary Sources</td>
<td>7</td>
</tr>
<tr>
<td>2.2.2 Supplementary Sources</td>
<td>8</td>
</tr>
<tr>
<td>2.3 Compilation</td>
<td>9</td>
</tr>
<tr>
<td>2.3.1 Parse Source Record Format</td>
<td>9</td>
</tr>
<tr>
<td>2.3.2 Merge Duplicates</td>
<td>10</td>
</tr>
<tr>
<td>2.3.3 Select the Best Form of Each Attribute</td>
<td>11</td>
</tr>
<tr>
<td>2.3.4 Filter Records</td>
<td>11</td>
</tr>
<tr>
<td>2.4 Corpus</td>
<td>12</td>
</tr>
<tr>
<td>2.4.1 Corpus Statistics</td>
<td>12</td>
</tr>
<tr>
<td>2.4.2 English Language Filtering</td>
<td>14</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Concept Token Counts per Document</td>
<td>35</td>
</tr>
<tr>
<td>4.2 Document concept token set scores</td>
<td>35</td>
</tr>
<tr>
<td>4.3 Concept Token Scores</td>
<td>36</td>
</tr>
<tr>
<td>4.4 Concept Token Error Fractions</td>
<td>36</td>
</tr>
<tr>
<td>4.5 Number of “Best List” Designations</td>
<td>38</td>
</tr>
<tr>
<td>4.6 Number of Similarity Errors</td>
<td>38</td>
</tr>
<tr>
<td>4.7 Multiedge Frequency</td>
<td>39</td>
</tr>
<tr>
<td>C.1 Node and edge counts per level of the optimal OAAC topic category tree</td>
<td>52</td>
</tr>
</tbody>
</table>
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Publication records by source</td>
<td>7</td>
</tr>
<tr>
<td>2.2 Corpus generation workflow</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Records A, B, and C form a connected component and are aggregated to form a unique merged record</td>
<td>10</td>
</tr>
<tr>
<td>2.4 Example Python code for on-line record handling from file</td>
<td>12</td>
</tr>
<tr>
<td>2.5 Top 20 journals by document count; 529,435 documents either are not published in a journal or have no journal included in their records</td>
<td>13</td>
</tr>
<tr>
<td>2.6 Top 10 publishers by document count; 530,449 documents have no publisher</td>
<td>14</td>
</tr>
<tr>
<td>3.1 Workflow diagram of concept topology modeling</td>
<td>18</td>
</tr>
<tr>
<td>3.2 Step 1: Content items (lettered boxes) decomposed into corresponding concept token sets (colored circles)</td>
<td>19</td>
</tr>
<tr>
<td>3.3 Step 2: More common concept tokens (circles) have lower specificity scores (numbered stars)</td>
<td>21</td>
</tr>
<tr>
<td>3.4 An adjacency list with each content item (lettered box) and its corresponding list of top $K = 2$ content items with the highest sum specificity score (numbered star)</td>
<td>23</td>
</tr>
<tr>
<td>3.5 Undirected concept similarity network corresponding to the adjacency list in Figure 3.4</td>
<td>24</td>
</tr>
</tbody>
</table>
3.6 Content items organized into an associative hierarchy based on the topology of the network in Figure 3.5 .......................... 25

3.7 An example adjacency matrix, where black squares are non-zero entries and vertices in the same block are adjacent in the rows and columns. Here, a non-zero entry represents an undirected edge between the row and column. .................................................. 26

4.1 An example of a hierarchical categorization tree produced from a small sample of 20,000 documents. ................................. 40
Chapter 1: Introduction

How does one organize “big content,” the rich media extension of big data? In the past, data analysis was performed on specific data samples. This is like viewing the world from a window; you can only see out of one window at a time, and you have to know what you are looking at in advance. More than just an implementation style for “data that do not fit on one computer,” “big data” is the idea that one can analyze an entire domain at once; it is a new type of analytical exploration that reveals insights about the topology of the data domain.

Big content analysis, then, is an analytical exploration of the meanings in an entire domain of content, from images to websites to videos, to, as we address in this thesis, the published scholarly texts of in all of the sciences. If one could capture how all academic works relate to each other, or their “concept topology,” what would it look like, and how would one go about it? These are the motivating questions that guided this research.

There are two main approaches to organizing content: 1) categorize it, and 2) rank it. These approaches are complementary; frequently, one is interested in relative ranking within a category, rather than an absolute global ranking. Here, we focus on categorizing: we propose a method of unsupervised hierarchical clustering of content by the concepts shared among them, and we apply it to a corpus of all open access
academic publications that we have assembled. Although we apply shared concept topology modeling to academic texts, our method can be applied generally to any large content corpus, given a method to decompose that content into “concept tokens,” or keyword-like, discrete elements of meaning.

1.1 Main Contributions and Organization

This research is significant for many reasons. First, we assembled a computationally curated corpus of all open access academic publications and published it on the web. Given that much of data science involves the labor of assembling and “cleaning” data [28], we hope that making this free resource available, we can facilitate and accelerate research in library and computer sciences.

Second, we proposed a novel topic categorization method that has the following attractive features:

- It is unsupervised and requires no training set.

- It produces a quantifiably optimal number of clusters. Other methods require the user to specify the number of clusters.

- It is human-interpretable at multiple steps in the workflow. This facet is critically important for the validation of intermediate results when tuning and evaluating the categorization produced by the method.

- It generates hierarchical clusters. This facet corresponds to how topical categories are organized by humans.

- It supports arbitrarily and heterogeneously sized clusters; the method makes no assumptions about the size and structure of clusters a priori.
• It is computationally feasible for large networks. We demonstrate this by applying our categorization method to a network of millions of vertices and tens of millions of edges, using only consumer-grade cloud computing.

• It does not require additional information like a social or citation network. It is applied directly to a network representing shared concepts in content itself.

Finally, we contribute the hierarchical categorization itself. Since our corpus contains all published open access academic works, the our method produces the best categorization scheme for publications in all of the sciences as they are freely available on the Internet.

1.1.1 Thesis Outline

The rest of this thesis is organized as follows:

Chapter 2 presents our motivating big content problem of categorizing all open access academic publications. We describe how we have assembled and computationally curated a corpus called the “Open Access Academia Corpus” (OAAC) of all such publications, which we publish online as a resource for future research in this domain.

Chapter 3 describes the details of our method and how it can be applied on “big content” generally.

Chapter 4 describes our experiment of applying our method described in Chapter 3 on the OAAC corpus we compiled as described in Chapter 2.

Chapter 5 summarizes the results of the thesis, and gives pointers for future research.

Appendix A documents the file format used in the OAAC corpus.
Appendix B includes a sampling of validation results related to decomposing OAAC records into concepts, choosing parameters, and constructing the network.

Appendix C presents a table of the hierarchical tree structure of our OAAC categorization result.
Chapter 2: The Open Access Academia Corpus (OAAC)

The rapid growth of open access (OA) academic publication worldwide challenges academics about how to evaluate this new work and include it in scholarly catalogs. However, a systematic approach to organizing cross-disciplinary open access publication has been hindered by fragmentation of sources and inconsistent presentation of content. To facilitate research in this domain, we have compiled the holdings of the three largest directories of open access academic publications: Directory of Open Access Journals (DOAJ), PubMed Central Open Access (PMC OA), and arXiv. We standardized the record format to support cross-source comparison, and we applied a variety of heuristics and external sources to improve record quality. We then published this corpus of 2,737,377 documents as a freely available resource at http://goo.gl/KOlgVX.

2.1 Introduction to Open Access Academic Publication

“Open access academic publication” refers to scholarly work hosted on the web with unrestricted access. Interest in it as a trend in academia ranges from the political [12] to the philosophical [43], from improving scientific access internationally [14] to increasing one’s own scientific impact as measured by views [9], citations [12, 17, 23],
and impact factors [15][10]. However, this academic publishing phenomenon is perhaps best understood as an aspect of the general trend associated with the Internet: accelerating media production due to nearly free, immediate, and omnipresent publication distribution worldwide. While the prevalence and importance of open access publication has been well-researched [5][6], here we address its challenges and opportunities as they relate to computer science research on methods to categorize large corpuses of heterogeneous content, or “big content.”

To mitigate these challenges, much work has been done to categorize open access publications into online directories, including the Directory of Open Access Journals (DOAJ), PubMed Central Open Access (PMC OA), and arXiv, and including online search tools like Google Scholar and CiteSeerX [16]. These resources have been boons to both academics and computer scientists researching the categorization of large corpuses. However, proprietary tools like Google Scholar forbid batch processing of researchers’ indexes, and tools like CiteSeerX and DBLP [26] are limited to single domains like computer science. To explore the domain of all open access science, a complete compilation of open access directories across disciplines is needed to standardize record format, combine cross-referenced entries, and ameliorate the myriad of data quality issues known to affect electronic directories of academic work [2][27][28].

We have created such a compilation, the Open Access Academia Corpus (OAAC), and published it at http://goo.gl/KoliVX. To our knowledge, this is the largest, most complete corpus of open access academic research freely available on the web. The main advantage of using OAAC in computer science research to categorize large corpuses is that it makes available a wide variety of disciplines, topics, and writing
styles in a single resource, while maintaining high-quality standards per corpus item, which are intrinsic to academic publishing. This richness and comprehensiveness makes OAAC particularly useful in research related to topical content analysis on a large scale.

2.2 Sources

![Publication Index Source](image)

Figure 2.1: Publication records by source

The OAAC is a standardized and computationally curated compilation of other authoritative open access publication directory resources. To create OAAC, we compiled the complete contents of the three largest public directories of open access academic publications as of March 2014, plus a small collection provided by OCLC.org (see Fig. 2.1). We downloaded record metadata in bulk from the services provided by these repositories.

2.2.1 Primary Sources

Directory of Open Access Journals (DOAJ) aims to index all open access peer-reviewed journals in all disciplines worldwide. To date, it has indexed 9,713
Journals and 1,502,700 articles from 133 countries [32].

URL: doaj.org

**arXiv** (pronounced “archive”) is a repository of 929,442 open access academic preprints (author-published) of physics, mathematics, computer science, and other quantitative fields. Many of these preprints are later published in other journals.

URL: arxiv.org

**PubMed Central Open Access (PMC OA)** makes available a subset of 774,993 biomedical science publications in PubMed Central that were released under permissive licenses.

URL: ncbi.nlm.nih.gov/pmc/tools/openftlist

### 2.2.2 Supplementary Sources

We also supplemented and extended primary source record entries using the following resources:

**Digital Object Identifier (DOI)** is persistent unique identifier of objects of any type, in particular, of academic publications. As part of the service, dx.doi.org stores some metadata about objects in the DOI system, including author names, journal titles, publisher names, and related International Standard Serial Numbers (ISSN). When DOIs are included in a record, we query dx.doi.org to confirm and supplement that record.

URL: dx.doi.org

**JournalTOC (Journal Tables of Contents)** is the largest free collection of scholarly journals, which lists 24,229 journals by ISSN. When ISSNs are included in
a record, we query JournalTOC to confirm the journal title and publisher.

URL: [journaltoacs.ac.uk](journaltocs.ac.uk)

## 2.3 Compilation

### 2.3.1 Parse Source Record Format

Over the course of March 2014, we downloaded the entire directory of DOAJ (1,502,700 records), arXiv (929,442 records), and the PubMed Central OA subset (774,993 records) using each service’s bulk download tools for a total of 3,207,135 records. Each of these tools produced a variety of files in different formats. We wrote custom parsers for each different file type to standardize each source record as a Python dictionary object of Unicode strings. Cleaning records included decoding XML, HTML, and \LaTeX^{\textregistered} encoded character sequences to Unicode, collapsing whitespace strings into single spaces, stripping spurious presentation mark-up and whitespace padding, and representing author names in a standard format of given names first, surnames last, and initials followed by a period. In the case of arXiv, we also converted arXiv, American Computing Machinery (ACM classification
system, acm.org/about/class/ccs98-html), and Mathematics Subject Classification (MSC) subject tags to their plain text mappings as additional author-provided keywords. For DOAJ records, we did not include DOAJ-suggested categories in our compilation.

![Diagram](image.png)

Figure 2.3: Records A, B, and C form a connected component and are aggregated to form a unique merged record.

### 2.3.2 Merge Duplicates

To merge duplicate records, we created a shared-identifier graph, where vertices are records and edges are when the two incident records share a title, URL, or DOI (see Fig. 2.3). Each connected component of this graph represents a unique canonical record, which we created by merging all of the records in the component. Connected components can be computed in $O(V + E)$ time [11], where $V$ is the number of records and $E$ is the number of shared URL, title, or DOI links between records. For each merged record, we then included any supplementary source information as linked by the merged record’s DOI or ISSN. The resulting merged record is a set of attributes where the value of each attribute is a list of values from all included sources.
2.3.3 Select the Best Form of Each Attribute

We selected the best form of each attribute by a variety of heuristics that differ slightly for each attribute type. In general, we converted all attributes to strings and selected first by the most frequent value and then by the longest string with the largest upper-lower-case entropy. For URL, we preferentially selected the URL that is not at a known directory domain and that has either a .pdf or .html extension. For author list, we preferred the author list as provided by a source in PubMed, arXiv, dx.doi.org, and DOAJ, in decreasing order of preference. We also examined all other author lists and the abstract for any matches of surnames adjacent to any names that could expand any initials in the preferred author list. In all cases, we kept any unselected alternate forms of any duplicate attributes with the record in the special alt attribute (see Appendix A). We did not include full document texts in our corpus; instead, we assumed that URLs served as proxies to copies of the full document text online.

2.3.4 Filter Records

We define a valid record as one that has a title, at least one author, at least one URL, and at most one DOI. Further, we require documents to have unique titles and URLs as a standard of quality control. To support duplicate records while removing spurious records with non-unique titles like “Errata” or non-unique URLs pointing to journal home pages, we limited the number of primary source records in a single merged record to 4. We removed any record that did not meet these criteria. After filtering, 2,737,377 valid records remained. Finally, we assigned each record a unique
alphanumeric ID. We published this filtered collection of valid, merged, unique records as the “FULL” set.

2.4 Corpus

```
for line in open(FILE_NAME):
    record = eval(line)
```

Figure 2.4: Example Python code for on-line record handling from file.

We published our compiled corpus and related files at [http://goo.gl/K01gVX](http://goo.gl/K01gVX) for free anonymous downloading over the Internet. For ease of use, each line is in a self-contained Python string format (“pickle” format), one line per record. See Figure 2.4 for a code usage example in Python. We also shuffled the line order from the processing order so that any block of lines represents a valid random sample of the entire corpus.

2.4.1 Corpus Statistics

The compiled corpus has a wide coverage of unique journals, authors, and languages (see the list below). Of records not associated with a journal, most (95.5%) are arXiv preprints without publication records and are presumably unpublished. We flagged unpublished articles for easy filtering, as they may indicate low-quality records (see Appendix A).

**Journals:** 13,197

**Publishers:** 5,187
Figure 2.5: Top 20 journals by document count; 529,435 documents either are not published in a journal or have no journal included in their records.

Authors: 3,916,783

Reported Languages: 70

Has DOI: 1,449,783 (52.0%)

Has Abstract: 2,534,770 (90.9%)

Has Tags (Keywords): 1,966,508 (70.5%)

Associated with a Journal: 2,270,960 (81.4%)

As shown in Figure 2.6, BioMed Central and the American Physical Society are the top two publishers of open access academic works in this corpus; this is as expected, since these two publishers are highly represented in PubMed Central OA and arXiv, respectively. The Public Library of Science is perhaps one of the most well-known open access academic publishers and the third most common publisher in OAAC; however, PLoS ONE is by far the most frequently used journals as shown in Figure 2.5 above.
Figure 2.6: Top 10 publishers by document count; 530,449 documents have no publisher.

2.4.2 English Language Filtering

While some authors and publications include document language labels, we find that the reported language, if any, frequently mismatch the actual language of the document. In DOAJ records particularly, language labels are frequently mixed together within the same record and even within the same block of text. Also, some record attributes contain mostly XML mark-up and little or no plain text. To mitigate these issues, we used English language spell check software to identify records with a majority of ordinary English words. While this approach sometimes includes documents comprised of mixed languages while excluding some jargon-dense English ones, it does ensure that each record contains a substantial fraction of computationally recognizable English text. We provided this subset of 2,039,079 documents (74.5% of FULL) as a separate corpus (ENGLISH).
2.5 Discussion of OAAC

The OAAC contains a large collection of open access academic publications from a wide variety of disciplines. As a corpus of text samples, it is particularly useful in computer science research related to content analysis on the web, because it is diverse in document topics, yet relatively consistent in quality, sophistication, and format. Nevertheless, many opportunities for future work on OAAC exist. First, the OAAC aggregates the records of several other directories, thus the quality and completeness of OAAC records must depend on the quality and completeness of these directories. While compiling the directories produced a “wisdom of crowds” aggregated record corpus of sorts, there is much potential for processing the full publication texts themselves, which we did not do.

Second, we only produced a snapshot of OA Academia at the time we compiled OAAC (Spring 2014). Ideally, new publications are automatically added to OAAC as they are published. Finally, we produced OAAC primarily for computer science researchers who would be most interested in downloading the corpus and processing it themselves. However, other tools could be built around OAAC, like a database-driven web interface to encourage usage by a wider audience and to support features like search and database queries.
Chapter 3: A Method of Categorizing Big Content

In the age of big content, methods are needed to make sense of large collections of rich content, like scholarly texts that are both computationally feasible and practically effective. In this chapter, we describe our novel method, which produces an optimal hierarchical categorization of any content type, given a method to decompose content items into concept tokens. First, we review the literature of automated content categorization with a mind for its relation to our concept topology method. We then present our method workflow step-by-step as it could be followed by a software engineer. Finally, we introduce the theory of our clustering method, stochastic blockmodeling, and justify its use, particularly for concept topology modeling. We discuss our method here from a computer systems engineering approach; we describe our method’s particular adaptation and application to our academic texts clustering problem in Section 4.

3.1 Work Related to Concept Topology Modeling

Clustering to discover topics or unsupervised categorization of content, particularly text, is a well-studied field with a rich history (see Aggarwal and Zhai [1] for a review). Big content, like most data, can be clustered using variants of classical clustering techniques like K-means or hierarchical clustering. However, unsupervised
topic modeling in text is most popularly based on two methods, Probabilistic Latent Semantic Indexing (PLSI) [19], and Latent Dirichlet Allocation (LDA) [7]. In these methods, a corpus is modeled as a function of hidden random variables, or topics, which are then estimated from corpus items represented as collections of features, typically words.

One advance in these techniques is to include additional information about the corpus to supplement existing topic models. Newman et al. improve unsupervised topic clustering on low-quality content by including information derived from external texts [33]. Likewise, Mei et al. use social network topology to inform topic modeling [30]. Conversely, topic similarity has been included as a supplementary source to improve community detection in a network of hyperlinks [40]. Other approaches reduce the complexity of the problem, using sampling methods like random projections [3] or intelligent edge filtering [40].

An alternate direction is to improve the way features in content are mathematically represented. The simplest models include the “bag of words,” in which each document is represented as a set of its words, and the “word count vector,” in which each dimension is a unique word and its value is the count of that word in the document. Recent improvements on these simplistic representations include Paragraph Vector [24] and Skip-gram model with extensions [31].

Finally, ideas related to concept topology modeling are not limited to text. For example, Jing and Baluja use the network structure of an inferred visual similarity graph to improve the image search results [21].
3.2 Workflow

Our workflow has five steps, each of which has a meaningful output that can be understood by humans (see Fig. 3.2). This is in contrast to many unsupervised categorization methods that function like a “black box” and are based on intermediate numerical representations intelligible only by computers, like variations of word or phrase vectors and their abstract distributions. Our approach of breaking our workflow with meaningful outputs has two major advantages. First, each step can be validated by expert review and statistical sampling. This makes customizing, tuning, and implementing the method much more practical, particularly in an industry setting. Second, the output of each step has applications itself. For example, the
top-K most similar adjacency list produced in Step 3 (see Section 3.2.3) can be used as a topical and ranked recommendation table, given a content item of interest. The five steps are as follows: 1) Decompose content into concept tokens; 2) Count token frequency and compute specificity score; 3) Construct top-K most similar list; 4) Construct similarity network; and for the last step, 5) Cluster using hierarchical blockmodeling.

3.2.1 Step 1: Decompose Content into Concept Tokens

Content items like text, music, video, and images are typically more challenging to analyze and organize than quantitative data because unlike numbers, how to represent rich content in a computer so that its meanings can be analyzed is not straightforward. In the past, content analysis required human reviewers to annotate content based on a human understanding of the concepts in content. However, this approach is frequently infeasible due to the volume of content to be reviewed and the expertise required to do so. More problematically, human review introduces human bias and batch effects into corpus annotations, a fundamental flaw in distributed human annotation. Indeed, without accounting for such bias, subsequent big data analysis may return spurious
results based on how the study was conducted, in this case, who did which batch of annotations. For example, such “batch effects” plague modern computational biology research in high-profile gene expression research [25].

We approach this challenge using a “standardized concept token set” model of representing the meaning of content items. A “concept token” is a discrete unit of meaning than can be represented in the computer as a set item. Here, we implement concept tokens as standardized keywords produced using natural language processing (NLP); however, other approaches customized for different types of media are equally applicable within the concept token set criteria. For example, images and video can be decomposed into concept token sets using the “bag of features” technique [20].

**Concept Token Set Generation Criteria**

- Apply the same method to all content items equally and independently. This reduces the risk of introducing feature bias and batch effects. Additionally, this allows parallel and distributed computation of concept token sets from content items.

- Bound the token set size to be a small, finite number. For example, in text processing, a single publication of hundreds to thousands of words should be represented as a concept token set of a dozen or so keyword concept tokens. This constraint exists chiefly to constrain the computational complexity of comparing large numbers of concept token sets in future steps.

- Bound the specificity of concept tokens so that tokens are not so unique that none are shared with other content items, but not so general that they don’t distinguish content items.
• Standardize the form of concept tokens so that they can be compared meaningfully and without ambiguity. For example, if using NLP keywords, “black cat,” “Black cat,” and “black_cats” are superficially different forms of the same concept and should all be represented as the same token. Conversely, “virus_(Computer)” and “virus_(Immunology)” refer to different concepts and should be represented as different tokens.

3.2.2 Step 2: Count Token Frequency and Compute Specificity Score

![Figure 3.3: Step 2: More common concept tokens (circles) have lower specificity scores (numbered stars)](image)

Figure 3.3: Step 2: More common concept tokens (circles) have lower specificity scores (numbered stars)

Not all concept tokens are equally important. For example, in an academic text corpus, consider the tokens “HIV” and “biology.” Intuitively, texts that share the token “HIV” should be more similar in topic than texts that share the token “biology” because “HIV” is a more specific term than “biology.”

To estimate concept specificity, count the frequency of each concept token in the corpus. Compute a specificity score per token that increases with inverse frequency; we propose the formula in Equation [3.1] where $S_i$ is the specificity score for token
$t_i$, $c_i$ is the number of times token $t_i$ appears in the corpus where $c_i > 1$, and $B$ is a user parameter. We cache the value of $S_i$ in a hash table $SPECIFITY$ indexed by the token $t_i$ for future use.

$$S_i = \frac{1}{\log_2(c_i + 2^B - 2) - B + 1}$$  \hspace{1cm} (3.1)

$$SPECIFITY[t_i] \leftarrow S_i$$ \hspace{1cm} (3.2)

The shape of the specificity score function is controlled by the user parameter $B$. $B$ ranges from 1 to $\infty$, and as it increases, it “smoothes” the specificity score function. When $B = 1$, rare tokens have a much higher relative specificity score compared to more common tokens. As $B$ increases, this difference is reduced; for $B = \infty$, $S = 1$ for all tokens regardless of frequency (i.e., each token counts equally).

Counting is straightforward to implement and is computationally efficient for even very large concept token sets. By this method, we avoid the complexities of inferring a concept ontology by relying on a big content assumption that the more frequently a concept is found in a large corpus, the less specific it is. Likewise, the less specific a concept is, the less closely related two content items should be that share it. We build on this last intuition in the next step.

### 3.2.3 Step 3: Construct Top-K Most Similar List

Create an adjacency list where two content items are adjacent if they share at least one content token. Optionally, preprocess the token set to discard unique tokens and “stop” tokens that appear too frequently and are not informative despite
Figure 3.4: An adjacency list with each content item (lettered box) and its corresponding list of top $K = 2$ content items with the highest sum specificity score (numbered star)

Dramatically increasing computation time. Order each list by decreasing sum specificity, or “similarity score” ($SS$) computed from the concept token set intersection of two content items as in Equation 3.3 where $SPECIFITY[t_i]$ is a constant-time table look-up for the specificity score of term $t_i$ and $\{\text{tokens}_r \cap \text{tokens}_s\}$ is the set of tokens shared by content items $r$ and $s$.

$$SS = \sum SPECIFITY[t_i] \mid t_i \in \{\text{tokens}_r \cap \text{tokens}_s\} \quad (3.3)$$

From a systems perspective, similarity score can be computed per item in parallel using read-only shared memory of the token sets of all content items in the corpus. From a computational complexity perspective, in the worst case, computing all-pairs set intersection is $O(|T| n^2)$, where $T$ is the set of all tokens and $n$ is the number of content items. In practice, this computation can be dramatically reduced by limiting the number and frequency of tokens Top $K$ can be solved in $O(m)$ time per target content item using quickselect [18], where $m \leq n$ is the number of other content items that share at least one token with the target.
3.2.4 Step 4: Construct Similarity Network

![Undirected concept similarity network](image)

Figure 3.5: Undirected concept similarity network corresponding to the adjacency list in Figure 3.4.

The top-K adjacency list in Step 3 maps directly to an undirected network, where vertices are content items and an edge represents a concept-similarity relationship. Optionally, one can add integer edge weights (edge multiplicities) that increase with the similarity score; we use the one in Equation 3.4, where $W_{rs}$ is the number of edges between vertices $r$ and $s$ as a function of their similarity score, $SS_{rs}$. Note that while real-valued weights are theoretically possible, our blockmodel clustering technique as formulated in this thesis requires integer values (see Section 3.3).

$$W_{rs} = \max(\lceil SS_{rs} - 1 \rceil, 1) \tag{3.4}$$

3.2.5 Step 5: Cluster Using Hierarchical Blockmodeling

Apply hierarchical blockmodeling in order to cluster items into a hierarchy by the topology of the concepts they share. The resulting hierarchy clusters very closely related content items into leaves and consecutively merges these leaves into clusters based on connections between them, until the entire corpus is merged. Label the hierarchical categorization structure, using a standard hierarchical addressing scheme.
Figure 3.6: Content items organized into an associative hierarchy based on the topology of the network in Figure 3.5.

*Conceptual distance* can be computed between any two hierarchically addressed content items by the maximum distance from the lowest common ancestor in the hierarchical categorization tree. Content items not included in the original clustering can be categorized by decomposing them into concept tokens, finding the top-K most similar content items in the corpus, and selecting a categorization based on the lowest common ancestor of the top-K most similar content items. We go into detail about hierarchical blockmodeling in Section 3.3.

### 3.3 Introduction to Stochastic Blockmodels

Fundamentally, a “stochastic blockmodel” is a statistical network model, where each of \( n \) vertices belongs to one of \( K \) “blocks” and the probability of an edge between two vertices is a function of blocks to which those vertices belong \[36\]. (See Newman \[34\] for a review of community mining and blockmodeling in particular.) This can be intuitively illustrated using an adjacency matrix, where the rows and columns are ordered so that vertices in the same block are adjacent. When the distribution of values in the adjacency matrix is determined by block membership, a characteristic block pattern becomes apparent (see Fig. 3.7). A “cluster” is a special case of a
blockmodel pattern, where vertices in the same block are more likely to be connected with each other than with vertices in different blocks. “Blockmodeling” then is the statistical inference technique of fitting empirical data to a stochastic blockmodel. That is, what is the most likely blockmodel that produced given data? The most likely blockmodel is then returned as the clustering result.

3.3.1 Selecting Blockmodels for Concept Topology Modeling

We choose our variant of stochastic block modeling over modularity maximization [8], random walk compression [39], and a variety of other techniques [13, 41] not only for its demonstrated superior results [38] and computational tractability [37], but also
because it best fits our problem. When fitting cluster assignments, we would not only like to consider the density of edges within the same cluster, but also the pattern of less dense cross edges between other potential clusters.

For example, consider our motivating application, categorizing academic publications. Say we were trying to fit a cluster of papers on cancer research, as in Figure 3.7. Such papers would be highly connected with each other, but they also would share some topical connections with papers on topics in medicine in general, like public health. However, they would probably have very few connections with papers in unrelated disciplines, like quantum physics or materials science. A stochastic block-model models not only the connectivity within a block, but the different probabilities between all blocks, and so considers this relevant evidence in fitting.

### 3.3.2 Blockmodel Formulation

Let $A_{ij}$ be entry in the adjacency matrix between vertices $i$ and $j$, which is 1 if there is an edge between $i$ and $j$, and 0 otherwise. Further, let $b_i$ be the block to which vertex $i$ belongs and let $p_{b_ib_j}$ be the probability of an edge between blocks $b_i$ and $b_j$. The probability that an observed network was generated by a blockmodel is expressed in Equation 3.5.

$$P = \prod_{i<j} p_{b_i b_j}^{A_{ij}} (1 - p_{b_i b_j})^{1-A_{ij}}$$

(3.5)

We then maximize this function, given the data to solve for the most likely or “optimal” blockmodel. While elegant, this approach in practice tends to perform poorly on empirical networks. Fortunately, many highly successful yet elegant extensions to address these issues have been published recently. To model the large differences in
vertex degrees found in most real networks, we apply the degree-corrected multigraph variant as in Karrer and Newman 2011 [22]. To improve the resolution of the method to detect small but well-defined clusters in big networks and to produce a hierarchical clustering, we apply the hierarchical extension as in Peixoto 2014 [38].

3.3.3 Degree-Corrected Multigraph Extension

Rather than depending on the binary presence or absence of an edge, it may be desirable to include edge weights so that some edges “count” more than others. Weights can be used to represent differences in statistical confidence, importance, flow, or in our case, a measure of similarity. To support edge weights, rather than modeling edges to be drawn from a Bernoulli distribution where values can only be 1 (present) or 0 (absent), we draw them from a Poisson distribution. A Poisson distribution produces non-negative integer values and can model any number of edges between two vertices. Such a network where multiple edges can be incident to the same two vertices is called a “multigraph” and is analogous to non-negative integer edge weights. As shown by Karrer and Newman [22], this can be formulated in terms of vertices and edge counts as in Equation 3.6 to solve for the maximally likely undirected graph model $G$ given the set of observations $g$

$$\mathcal{L}(G|g) = \sum_{rs} m_{rs} \log \frac{m_{rs}}{n_r n_s} \quad (3.6)$$

where $r$ and $s$ are blocks, $m_{rs}$ is the number of edges between blocks $r$ and $s$, and $n_r$ is the number of vertices in block $r$.

In practice, this model tends to produce poor results when fit to real networks. This is because in this model, vertices are assumed to have roughly the same degree,
but in real networks, the vertex degree may vary quite a bit. Thus, results from using this method tend to fit vertices into blocks by similar degrees, rather than by assortative clustering as intended. To produce a better model, we must also account for the observed degree sequence when fitting the most likely block assignments.

To model the large differences in vertex degrees found in most real networks, we apply the degree-corrected multigraph variant as in Karrer and Newman [22]. In this formulation, we replace vertex counts $n$ with the sum of vertex out-degrees $\kappa$, where $\kappa_r$ is the number of edges incident to a vertex in block $r$ (see Equation 3.7).

$$\mathcal{L}(G|\theta) = \sum_{rs} m_{rs} \log \frac{m_{rs}}{\kappa_r \kappa_s}$$  \hspace{1cm} (3.7)

3.3.4 Hierarchical Extension

One weakness of blockmodeling is that when the network grows large, the minimum detectable cluster size may grow larger than the size of the true clusters in the network. For blockmodels, this size scales with $O(\sqrt{N})$, where $N$ is the number of vertices in the network. As a result, small but true clusters with fewer than $O(\sqrt{N})$ vertices will not be detected. As shown by Peixoto [38], this limitation can be mitigated by an order of magnitude, to $O(\log N)$, by fitting a hierarchy of blockmodels. This approach is based on the observation that a blockmodel can itself be fit to a higher-level blockmodel, where the blocks map to vertices, edges between blocks map to multiedges, and edges within a block map to self-multiedges. Hierarchical blockmodeling improves detectable cluster size resolution to $O(\log N)$. Additionally, and of particular importance to our topic categorization application, this approach
produces a hierarchical rather than disjointed clusters. It is non-parametric, computationally tractable, and mitigates overfitting by accounting for the complexity of the model description during optimization. We use Peixoto’s graph_tool\textsuperscript{35} Python package implementation, minimize_nested_blockmodel_dl, which incorporates both the degree-corrected multigraph and hierarchical extensions, and is computed using an efficient Markov chain Monte Carlo (MCMC) implementation\textsuperscript{37}.
Chapter 4: A Map to All of Academia

In this chapter, we apply our method proposed in Chapter 2 on the corpus we have published as described in Chapter 1. The result is an optimal categorization of all open access academic publications, which we describe in this chapter.

4.1 Concept Token Generation Using Wisdom-of-Crowds NLP

When no particular method is necessarily the best, a popular and effective approach can be to combine results from multiple independently designed and implemented sources [29], an approach known as “wisdom of crowds.” Here, we apply this approach to produce concept tokens from text using two independent natural language processing (NLP) services, AlchemyAPI (http://www.alchemyapi.com) and Aylien (http://aylien.com). By combining results, we increased our confidence in tokens produced by both services and, given that both services operate independently, dramatically mitigated any poor performance of one service with results from the other. The complete set of resulting concept tokens per document is available at the OAAC webpage at http://goo.gl/KOlgVX.
4.1.1 NLP Computation on OAAC

As input to both services, we formatted document title, abstract, and author-provided keywords, if any, into HTML strings that annotated the document title. From AlchemyAPI, we used the *Entity Extraction* and *Concept Tagging* services; from Aylien, we used only the *Concept Extraction* service. Entity extraction is context- and grammar-sensitive automated detection of important proper nouns like the names of people, places, and things. (See [http://www.alchemyapi.com/products/features/entity-extraction](http://www.alchemyapi.com/products/features/entity-extraction) for more details.) Concept extraction / tagging associates the inferred meaning of text with terms in ontological databases, for example, DBpedia [4]. Concept extraction is capable of identifying concepts that are not explicitly referenced in text. (See [http://www.alchemyapi.com/products/features/concept-tagging](http://www.alchemyapi.com/products/features/concept-tagging) and [http://aylien.com/text-api-doc#concepts](http://aylien.com/text-api-doc#concepts) for more details.)

NLP computation took approximately one month of daily batch computations. We only computed concept tokens for a random subset of 1,502,473 out of 2,039,079 articles (73.7%) from the OAAC corpus ENGLISH.jul20.2014 due to computational constraints.

4.1.2 Producing Concept Tokens from NLP Results

Concept token production from raw NLP results occurred in two steps: reduce raw NLP service results into scored token lists, and combine scored token lists into a final concept token set per document.

To produce scored token lists per service, first we selected a string to represent the “raw” token depending on the NLP results returned. Entity extraction returns words as they appear in the text; we used this string as a raw token. Concept extraction
returns URLs to ontology databases like DBpedia; we used the string as it appeared in the database URL as a raw token. We then standardized raw token strings by stemming words, standardizing the letter case, and replacing whitespace and special characters with underscores. Finally, we filtered low quality or irrelevant results using thresholds customized per type of raw NLP results returned.

Step one produced two lists of scored tokens, one from Aylien, and one from AlchemyAPI. Scores were between 0 and 1 and were derived from service-specific “relevance” or “confidence” scores. To combine these two lists, first we “boosted” the best score of tokens returned by both services as in Equation 4.1. We then sorted tokens by decreasing score (highest score first) and choose up to the first 30 tokens.

\[
\text{MaxScore} = \max(TokenScore_{Aylien}, TokenScore_{AlchemyAPI})
\]

\[
\text{BoostedScore} = \text{MaxScore} + \frac{1 - \text{MaxScore}}{2}
\]  

(4.1)

4.2 NLP Concept Token Method Results

We validated the quality of our NLP concept token procedure using a manual review of 300 randomly selected results. An example result can be found in Appendix B.1; the complete set of manual review results are offered at: [https://gist.github.com/andrewdyates/d88cdf6c0d28820e2cc8](https://gist.github.com/andrewdyates/d88cdf6c0d28820e2cc8). We evaluated the validity of both individual concept tokens and the overall quality of the set of concept tokens. Overall, we found that individual concept token errors were infrequent (3%), and that overall concept token set quality was quite high, with no observed errors.
4.2.1 Scoring Rubric: Concept Token Score

A concept token score represents the reviewer’s judgment of how well a token individually represents an important concept in its corresponding document.

1: Good: Concept token captures an idea relevant to this work. Example: “Neuroplasticity in mood disorders” – “Mood_disorder”

0: Error: Concept token is tangentially related to this work but is not in the work. An argument could be made about why this error is not as “bad” as a -2 error. Example: “Building a model for scoring 20 or more runs in a baseball game” – “St._Louis_Cardinals”

-1: Error: Concept token word is correct, but it has been disambiguated incorrectly. Example: “Neuroplasticity in mood disorders” – “Plasma_(physics)”

-2: Error: Concept token is an error and has no relevance to the work. Example: “Effect of N Fertilizer Application on Growth and Yield of Inoculated Soybean” – Data_storage_device

4.2.2 Scoring Rubric: Overall Document Score

An overall document score represents the reviewer’s judgment of how well the associated concept token set faithfully represent all of the important concepts in a document. These scores account for cases like when all individual concept tokens are high quality, but major concepts are missing.

2: Good: All of the main ideas are captured by the inferred concept set; the concept set satisfactorily represents the ideas in this work. The concept set does not
have to be perfect, but the reviewer must strongly consider downgrading the
document score if there is more than one error.

1: Fair: Many ideas of the work are captured by the concept set, but some key ideas
are not represented and / or there are multiple errors. A human reviewer would
judge the concept set to be useful, but not entirely satisfactory.

0: Null: Only 0 to 2 concepts are inferred. Idea coverage is insufficient, but the re-
view is not mislead. These results are flagged as NLP failures in post-processing
and removed; so, they do not distort downstream results in the workflow.

-1: Error: The inferred concept set misleads the reviewer about the true ideas in
this work. Some true ideas may be represented. Such results cannot be au-
tomatically removed in post-processing and may distort downstream results in
the workflow.

4.2.3 Manual Review Results

<table>
<thead>
<tr>
<th>Min</th>
<th>Mean</th>
<th>Median</th>
<th>Std Dev</th>
<th>Max</th>
<th># Terms</th>
<th># Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12.5</td>
<td>12</td>
<td>4.9</td>
<td>36</td>
<td>3739</td>
<td>300</td>
</tr>
</tbody>
</table>

Table 4.1: Concept Token Counts per Document

<table>
<thead>
<tr>
<th>Score</th>
<th>-1</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>0</td>
<td>2</td>
<td>30</td>
<td>268</td>
<td>300</td>
</tr>
</tbody>
</table>

Table 4.2: Document concept token set scores
Remarkably, we observed no document-level errors (score of -1 or -2) in our observed sample; the corpus document error rate is less than 1%, with 95% confidence by a one-sided binomial test. Document level “Good” rate, or the rate that the human reviewer felt the results were not merely relevant but comparable with human annotation, is 89.3% (85.3% to 92.6% with 95% confidence by binomial test, (see Tables 4.1 and 4.2). At the concept token level, errors (score: < 1) were more frequent, but still rare (3.4%, 2.8% to 4.0%, with 95% confidence by binomial test; see Table 4.3), with all three error types being roughly equally frequent (see Table 4.4).

<table>
<thead>
<tr>
<th>Score</th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>54</td>
<td>31</td>
<td>42</td>
<td>3025</td>
<td>587</td>
<td>3739</td>
</tr>
</tbody>
</table>

Table 4.3: Concept Token Scores

<table>
<thead>
<tr>
<th>Score</th>
<th>% of Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2</td>
<td>42.5%</td>
</tr>
<tr>
<td>-1</td>
<td>24.4%</td>
</tr>
<tr>
<td>0</td>
<td>33.1%</td>
</tr>
</tbody>
</table>

Table 4.4: Concept Token Error Fractions

4.3 Top-K Edge Results

We validated the top-K most similar document results by a manual review of 100 randomly selected documents and their corresponding top $K = 5$ most similar
documents. Included in this review was a justification for our selection of the user parameter $B$, which controls the “smoothness” of the specificity score function. We produced results for three different values of $B$: 1, 10, and $\infty$ (token count). Overall, we found that errors were rare for all three values of $B$ (best: 0.8%, 0.2% to 2.0%, with 95% confidence by binomial test), and that $B = 10$ had the best performance. An example result is in Appendix B.2; complete manual review results can be found at: https://gist.github.com/andrewdyates/fd7203271af064bd870e

4.3.1 Scoring Rubric and Method

We manually reviewed 100 randomly selected documents and their top $K = 5$ most similar results for $B \in \{1, 10, \infty\}$ (15 total results reviewed per target document). For each top $K = 5$ set for each value of $B$, we recorded the number of errors (0 to 5) and a 1 if that value of $B$ produced the best results, otherwise 0. We allowed duplicate “best set” scores; frequently, two or three sets were either the same or indistinguishably good.

4.3.2 Manual Review Results

$B = 10$ performed significantly better (confidence 90% binomial test) than both $B = 1$ and $B = \infty$ in “best set” scoring and also better in error frequency, but not significantly (see Tables 4.5 and 4.6; * indicates a significant result with 90% confidence per two-sided binomial test). On inspecting errors, in general, the deleterious effects of erroneous concept tokens tended to average out with other tokens. Instead, similarity-level errors seemed to have two primary causes: 1) a single rare but tangentially or unrelated concept was given more weight than the sum of evidence from other shared concepts; or 2) key specific concepts were not given more weight than
the sum of many other common and possibly less important shared concepts. This was especially aggravated when a target document’s concept list was short (6 or fewer concepts). These two failure cases tended to correspond to the extremes of $B = 1$, where rare tokens have the highest relative specificity score, and $B = \infty$, where the token specificity score is constant and not related to token frequency. $B = 10$ blended these two approaches by weighting rare tokens more than common tokens, but at a less extreme scale than $B = 1$. Thus, $B = 10$ mitigated these two failure cases and tended to produce superior results.

<table>
<thead>
<tr>
<th>$B$</th>
<th># “Best”</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>75/100</td>
</tr>
<tr>
<td>10</td>
<td>89/100*</td>
</tr>
<tr>
<td>$\infty$</td>
<td>64/100</td>
</tr>
</tbody>
</table>

Table 4.5: Number of “Best List” Designations

<table>
<thead>
<tr>
<th>$B$</th>
<th># Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7/500</td>
</tr>
<tr>
<td>10</td>
<td>4/500</td>
</tr>
<tr>
<td>$\infty$</td>
<td>8/500</td>
</tr>
</tbody>
</table>

Table 4.6: Number of Similarity Errors
4.4 Hierarchical Clustering Results

We generated an optimal, hierarchical categorization using hierarchical block-clustering. From 1,502,473 records in the OAAC corpus ENGLISH.jul20.2014 with computed NLP concept tokens, we removed records with with < 5 tokens and limited token set sizes to at most 30 tokens. As mentioned earlier, we used $B = 10$ to compute document similarity scores and Equation 3.4 to compute edge multiplicities with a ceiling multiplicity of 6. We set top $K = 15$ based on informal tests of computation time and resulting mean nats per edge categorization optimization results, and we set implementation user parameter $E = 0.001$ to improve the computation time for large networks. In total, we fitted a network of 1,459,110 documents and 18,041,069 similarity edges with an average edge multiplicity of 1.6 (see Table 4.7).

Optimization computation took 59 compute hours on an m2.4xlarge Amazon EC2 instance. The final hierarchy contains 84,190 leaf clusters and 37 levels, and the resulting minimum description length is -5.96 nats per multiedge, which is very strong numerical evidence of a good fit to a hierarchical block structure. For comparison, the same network without multiedges results in a minimum description length of -4.91 nats per edge. (See Appendix C for table of the complete categorical tree structure.) We also include an example tree diagram of a 20,000 document sample as shown in Figure 4.1.

<table>
<thead>
<tr>
<th>Multiedge</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>9,506,586</td>
<td>6,686,319</td>
<td>1,492,151</td>
<td>275,930</td>
<td>53,738</td>
<td>26,345</td>
</tr>
<tr>
<td>Percent</td>
<td>52.7%</td>
<td>37.1%</td>
<td>8.3%</td>
<td>1.5%</td>
<td>0.3%</td>
<td>0.1%</td>
</tr>
</tbody>
</table>

Table 4.7: Multiedge Frequency
4.5 Ongoing Work

While the production of a map of all open access academic publications is exciting, work is still required to validate, label, illustrate, and apply the results of this optimization. This work is in progress and will be included in publications regarding these results. Also, due to resource constraints, only one observer performed manual review and validation. Ongoing work includes expanding manual validation to use multiple observers.
Chapter 5: Conclusions

In this thesis, we have achieved three things. First, we assembled and published a corpus of all open access academic publications as they are freely available on the Internet. Second, we proposed a novel method for hierarchically categorizing content by topic, using modern statistical techniques. Finally, we applied our method on our corpus using NLP, producing an optimal map of all the sciences as they are available on the Internet.

5.1 Recommendations for Future Work

In this line of research, more validation, illustration, and application of our “Map to All of Academia” is needed. We are currently performing a systematic manual review and producing visualizations. At a web service based on these results, we will also include this information at http://open-academia.org. We also plan to apply our method to a better known corpus, an already well-categorized text corpus like Wikipedia and compare our categorical results with human-curated ones.

In new research directions, we plan to adapt our method for other types of media; in particular, image databases. Doing so will require new research into how best to decompose images into “visual concept tokens” and properties of the “visual concept topology” our method would produce when applied as such. Finally, the theory and
demonstrated applicability of hierarchical blockmodeling to adjacency matrices is exciting, and we plan to apply it to optimally cluster the “signed and biased” adjacency matrices produced in our previous research about the Glyph SPLOM method [44].
Appendix A: OAAC File Format

A.1 Description of Record Attributes

Each record in OAAC is a Python dictionary of key and value pairs. Any unknown attribute has the value None. We list all keys here with a brief description of the values that may be assigned to each key.

A.1.1 Document

id: A 16 alphanumeric ID that uniquely identifies the record in the corpus

title: Document title

authors: A list of the document authors in the order of publication. Each name is a single string with given names first and then surname.

abstract: The document abstract.

url: The web link to the document’s full text.

tags: A list of any author or journal provided keywords, tags, or categories.

A.1.2 Publication

journal: Full name of journal in which the document is published.
**publisher:** Full name of the publisher that published the document.

**doi:** Digital Object Identifier, a unique ID assigned to published works by the International DOI Foundation (IDF). See doi.org for more information. DOIs refer to individual documents. DOIs begin with “10.”

**issns:** Set of International Standard Serial Numbers (ISSNs) associated with the document. ISSNs refer to journals and periodical publications rather than individual documents. ISSNs are in the form of the regular expression:

\d{4}\-\d{3}[\dX]

**year:** earliest recorded publication year.

**page:** page number in publication.

**issue:** issue of publication.

**volume:** volume of publication.

**language:** list of languages claimed by the journal and the document author. Note that the actual language of the record and its associated document may differ.

**A.1.3 Meta**

**referenced:** A dictionary of references to supplementary records used to create the merged record.

**doaj.journal:** DOAJ Journal meta record; see [doaj.org/faq#metadata](http://doaj.org/faq#metadata)

**issn:** JournalTOC ISSN record

**doi:** [dx.doi.org](http://dx.doi.org) DOI record
**merged_info:** Set of tuples of the parsed records included in this merged record. In each tuple, the letter indicates the source, while the number indicates the record number from that parsed source file. Letter Codes: A: ArXiv, O: OCLC-provided subset, D: DOAJ, P: PubMed.

**spellcheck:** A dictionary of spell check results for the title, abstract, and tags attributes.

**eng_pct:** The fraction of words recognized by an English spell checking program

**xml_pct:** The fraction of common XML characters like <, >, and &

**accept_text:** Indicates whether the record passed the “sufficient English” test field: title, abstract (abstr), and tags. None indicates a failure of the entire record to pass the “sufficient English” test. Note that the record as a whole can pass the test despite that some of the fields have failed the test.

**alts:** A dictionary of attribute key names to a list of alternate values for that attribute. We save these alternate values rather than discarding them after selecting the best attribute version.

**is_published:** A boolean flag indicating whether the document record has a known journal or publisher associated with it.
Appendix B: Experimental Validation Results

B.1 Example Annotated NLP Concept Token Result

An example concept token result for an OAAC record. Concept token results are provided in tuples (token, ranking_score, review_score) where the review_score is manually annotated according to the method described in Section 4.1.

**TITLE:** Pre-exposure prophylaxis: where are we in Europe?


**OVERALL RESULTS SCORE:** 2 (Good)

**RANKED AND SCORED CONCEPT TOKENS:** (Tenofovir, 1.00, 1), (Emtricitabine, 1.00, 1), (Serodiscordant, 1.00, 1), (CAPRISA_004, 1.00, 1), (Vertically_transmitted_infection, 1.00, 1), (Drug_delivery, 1.00, 1), (AIDS, 1.00, 2), (Pre-exposure_prophylaxis, 1.00, 2), (Post-exposure_prophylaxis, 1.00, 1), (Sub-Saharan_Africa, 1.00, 1), (Europe, 1.00, 1), (HIV, 0.99, 2), (Pharmacokinetics, 0.97, 1), (Clinical_trial, 0.83, 1), (Antiretroviral_drug, 0.81, 1), (Developed_country, 0.67, 1), (Sexual_intercourse, 0.61, 1)

**AUTHOR PROVIDED KEYWORDS:** None

**OAAC ARTICLE ID:** 4KQZCF2OHBH00RG

**ABSTRACT:** Despite major advances in HIV therapy, the number of new HIV infections remains very high, even in high-income countries where resurgence among
men who have sex with men (MSM) has been witnessed. New prevention strategies have therefore to be assessed in order to curb the incidence of HIV infection. Recent studies have explored the effectiveness of antiretroviral therapy (ART) for HIV prevention and have generally yielded encouraging results. ART has been used successfully to prevent mother-to-child transmission of HIV, HIV acquisition following occupational or sexual exposure to HIV (post-exposure prophylaxis), and more recently, to reduce the risk of HIV transmission within a serodiscordant couple by treating the HIV-positive partner (HPTN 052 study). Another possible use of ART in prevention is pre-exposure prophylaxis, where ART is taken by an HIV-seronegative individual before HIV exposure. This PrEP strategy has been validated in animal models and more recently assessed in clinical trials in humans. The results of six large efficacy trials of PrEP are now available, but results have been inconsistent. The use of tenofovir gel in women at higher risk in Sub-Saharan Africa has shown efficacy when given before and after sex in the Caprisa 004 study (reduction of 39% of the incidence of HIV), whereas no efficacy was shown with daily use in the VOICE trial. Similarly, daily oral PrEP with tenofovir or tenofovir and emtricitabine has proved effective in the Iprex trial in MSM (reduction of 42% of HIV incidence), in the Partners PrEP study (reduction of 67 to 75% in HIV incidence) and in the TDF-2 trial (reduction of 63% in HIV incidence), but not in the Fem-PrEP or the VOICE trials in women. There are many potential explanations for these apparently conflicting results, such as the populations in which these strategies have been assessed, the differential pharmacokinetics of ART in the male and female genital tracts and most likely the high level of adherence which is required to confer protection against HIV.
acquisition. These results have also generated a lot of controversy about the imple-
mentation of PrEP. Some think that the data are good enough to rollout PrEP in
key populations at higher risk. Others think more research is needed before PrEP is
implemented because of concerns around safety, emerging resistance, cost and change
in sexual behaviour that might offset the benefit of PrEP. Safety is indeed a major
concern in healthy individuals. New studies are underway to address these issues and
are assessing PrEP regimens in open-label studies (Iprex-OLE in MSM), intermittent
PrEP regimens to try to improve adherence, new ART classes and new modalities
of drug delivery. PrEP is therefore a promising biomedical intervention that might
be used in the near future in addition to current prevention methods to prevent HIV
infection and help control the spread of this infection.

B.2 Example Top $K = 5$ Most Similar Results

Computed using the OAAC corpus 2014; similarity scores computed using $B = 10$.
In manual review, all top 5 most similar records were judged to be valid results in
this example.

Target Record

**TITLE:** Global Expression Profiling in Atopic Eczema Reveals Reciprocal Expres-
sion of Inflammatory and Lipid Genes

**URL:** [http://www.ncbi.nlm.nih.gov/pmc/articles/pmc2603322/](http://www.ncbi.nlm.nih.gov/pmc/articles/pmc2603322/)

**OAAC ARTICLE ID:** 0000I3SKQ1BEQMQH

**CONCEPT TOKENS:** Dermatitis, Malassezia, Genetic_linkage, Gene_expression_profiling,
Lipid, Constellation, Homeostasis, DNA, DNA_microarray, Gene, Cutaneous_condition,
Atopic_dermatitis, Genetics, Gene_expression, Voluntary_association, Skin_biopsy
Most Similar Record #1

**TITLE:** Inhibition of Transcription Factor Specificity Protein 1 Alters the Gene Expression Profile of Keratinocytes Leading to Upregulation of Kallikrein-Related Peptidases and Thymic Stromal Lymphopoietin

**URL:** [http://www.ncbi.nlm.nih.gov/pmc/articles/pmc3193562/](http://www.ncbi.nlm.nih.gov/pmc/articles/pmc3193562/)

**OAAC ARTICLE ID:** WPF7VNJNIAJ36QZG

**SIMILARITY SCORE:** 2.61

**CONCEPT TOKENS:** Keratinocyte, Protease, Gene, Protein, Dermatitis, Kallikrein, Atopic_eczema, Gene_expression_profiling, Gene_expression, Thymic_Stromal_Lymphopoietin, Transcription_(genetics), Small_interfering_RNA, Epithelium, DNA, Atopic_dermatitis, RNA, DNA_microarray, Genetics, Messenger_RNA

Most Similar Record #2

**TITLE:** Nanovesicles from Malassezia sympodialis and Host Exosomes Induce Cytokine Responses Novel Mechanisms for Host-Microbe Interactions in Atopic Eczema

**URL:** [http://www.ncbi.nlm.nih.gov/pmc/articles/pmc3142114/](http://www.ncbi.nlm.nih.gov/pmc/articles/pmc3142114/)

**OAAC ARTICLE ID:** 97C9YWHSQFYBC57V

**SIMILARITY SCORE:** 2.20

**CONCEPT TOKENS:** Dermatitis, Differential_centrifugation, Malassezia, Flow_cytometry, Exosome_(vesicle), Atopic_eczema, T_cell, ELISPOT, Fungus, Inflammation, ELISA, Western_blot, Dendritic_cell, Immune_system, Endogeny, Yeast, M.sympodialis, Plasma(phys), Pathogenesis, Electron_microscope, Asthma, Atopic_dermatitis, Bacteria, Dendrite, Cell_biology, Malassezia_sympodialis
Most Similar Record #3

**TITLE:** Malassezia Folliculitis versus Truncal Acne Vulgaris (Clinical and Histopathological Study)


**OAAC ARTICLE ID:** 9VKUHYJ03HR0EDWW

**SIMILARITY SCORE:** 2.07

**CONCEPT TOKENS:** Hair_follicle, Acne_vulgaris, Malassezia, Antifungal_medication, Folliculitis, Basra, Hospital, Histopathology, Cutaneous_condition, Sebaceous_gland, Inflammation, Scar, Skin_biopsy

Most Similar Record #4

**TITLE:** Levels of adiponectin, a marker for PPAR-gamma activity, correlate with skin fibrosis in systemic sclerosis: potential utility as a biomarker?

**URL:** [http://www.ncbi.nlm.nih.gov/pmc/articles/pmc3446479/](http://www.ncbi.nlm.nih.gov/pmc/articles/pmc3446479/)

**OAAC ARTICLE ID:** N2EOYOZPW2FK31WT

**SIMILARITY SCORE:** 2.06

**CONCEPT TOKENS:** Peroxisome_proliferator-activated_receptor_gamma, Longitudinal_study, Skin_biopsy, Transforming_growth_factor_beta, Fibroblast, Gene_expression_profiling, Adiponectin, Systemic_sclerosis, Gene, Agonist, Fibrosis, Biomarker, Peroxisome_proliferator-activated_receptor, DNA, Ligand, Subcutaneous_tissue, Scleroderma, Transforming_growth_factor, Genetics, Cell_nucleus, DNA_microarray, Gene_expression, Microarray, Mesenchymal_cells, Adipocyte
Most Similar Record #5

**TITLE:** Genome-wide linkage study of atopic dermatitis in West Highland White Terriers

**URL:** [http://www.ncbi.nlm.nih.gov/pmc/articles PMC3111396](http://www.ncbi.nlm.nih.gov/pmc/articles/pmc3111396/)

**OAAC ARTICLE ID:** VKWJ2XE5Q815OEV

**SIMILARITY SCORE:** 1.89

**CONCEPT TOKENS:** Itch, Chromosome 7 (human), West Highland White Terrier, Statistical power, Environmental Modification Convention, Genetic linkage, Mutation, Atopic eczema, Cutaneous condition, Histopathology, Atopic dermatitis, Candidiasis, Microsatellite, Gene, Asthma, Chromosomal crossover, Voluntary association, Genetic association
Appendix C: Hierarchical Categorization Tree Table

<table>
<thead>
<tr>
<th>Level</th>
<th>Vertices</th>
<th>Edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>37 (Root)</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>36</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>35</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>34</td>
<td>7</td>
<td>25</td>
</tr>
<tr>
<td>33</td>
<td>9</td>
<td>39</td>
</tr>
<tr>
<td>32</td>
<td>13</td>
<td>78</td>
</tr>
<tr>
<td>31</td>
<td>19</td>
<td>148</td>
</tr>
<tr>
<td>30</td>
<td>29</td>
<td>239</td>
</tr>
<tr>
<td>29</td>
<td>40</td>
<td>408</td>
</tr>
<tr>
<td>28</td>
<td>44</td>
<td>759</td>
</tr>
<tr>
<td>27</td>
<td>51</td>
<td>1003</td>
</tr>
<tr>
<td>26</td>
<td>60</td>
<td>1321</td>
</tr>
<tr>
<td>25</td>
<td>66</td>
<td>1587</td>
</tr>
<tr>
<td>24</td>
<td>84</td>
<td>2392</td>
</tr>
<tr>
<td>23</td>
<td>99</td>
<td>3809</td>
</tr>
<tr>
<td>22</td>
<td>142</td>
<td>6796</td>
</tr>
<tr>
<td>21</td>
<td>170</td>
<td>9696</td>
</tr>
<tr>
<td>20</td>
<td>185</td>
<td>14495</td>
</tr>
<tr>
<td>19</td>
<td>225</td>
<td>19612</td>
</tr>
<tr>
<td>18</td>
<td>281</td>
<td>29222</td>
</tr>
<tr>
<td>17</td>
<td>344</td>
<td>37581</td>
</tr>
<tr>
<td>16</td>
<td>396</td>
<td>45944</td>
</tr>
<tr>
<td>15</td>
<td>461</td>
<td>54889</td>
</tr>
<tr>
<td>14</td>
<td>533</td>
<td>65016</td>
</tr>
<tr>
<td>13</td>
<td>586</td>
<td>84948</td>
</tr>
<tr>
<td>12</td>
<td>661</td>
<td>96030</td>
</tr>
<tr>
<td>11</td>
<td>714</td>
<td>104718</td>
</tr>
<tr>
<td>10</td>
<td>772</td>
<td>112642</td>
</tr>
<tr>
<td>9</td>
<td>905</td>
<td>138829</td>
</tr>
<tr>
<td>8</td>
<td>1492</td>
<td>297197</td>
</tr>
<tr>
<td>7</td>
<td>2544</td>
<td>552969</td>
</tr>
<tr>
<td>6</td>
<td>3921</td>
<td>770816</td>
</tr>
<tr>
<td>5</td>
<td>6786</td>
<td>1111844</td>
</tr>
<tr>
<td>4</td>
<td>10876</td>
<td>1674881</td>
</tr>
<tr>
<td>3</td>
<td>16570</td>
<td>2224590</td>
</tr>
<tr>
<td>2</td>
<td>37831</td>
<td>3559807</td>
</tr>
<tr>
<td>1</td>
<td>84190</td>
<td>5107954</td>
</tr>
<tr>
<td>0 (Document)</td>
<td>1,459,110</td>
<td>18,041,069</td>
</tr>
</tbody>
</table>

Table C.1: Node and edge counts per level of the optimal OAAC topic category tree
Bibliography


