AckSeer: A Repository and Search Engine for Automatically Extracted Acknowledgments from Digital Libraries

Madian Khabsa
Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802
madian@psu.edu

Pucktada Treeratpituk
Information Sciences and Technology
The Pennsylvania State University
University Park, PA 16802
pxt162@ist.psu.edu

C. Lee Giles
Information Sciences and Technology
Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802
giles@ist.psu.edu

ABSTRACT

Acknowledgments are widely used in scientific articles to express gratitude and credit collaborators. Despite suggestions that indexing acknowledgments automatically will give interesting insights [9], there is currently, to the best of our knowledge, no such system to track acknowledgments and index them [1]. In this paper we introduce *AckSeer*[2], a search engine and a repository for automatically extracted acknowledgments in digital libraries. *AckSeer* is a fully automated system that scans items in digital libraries including conference papers, journals, and books extracting acknowledgment sections and identifying acknowledged entities mentioned within.

We describe the architecture of *AckSeer* and discuss the extraction algorithms that achieve a F1 measure above 83%. We use multiple Named Entity Recognition (NER) tools and propose a method for merging the outcome from different recognizers. The resulting entities are stored in a database then made searchable by adding them to the *AckSeer* index along with the metadata of the containing paper/book. We build*AckSeer* on top of the documents in CiteSeerx digital library yielding more than 500,000 acknowledgments and more than 4 million mentioned entities.

Categories and Subject Descriptors

I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text analysis*; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Linguistic processing*; H.3.1 [**Information Storage and Retrieval**]: Digital Libraries—*Collection*

General Terms

Algorithms, Search engines

Keywords

Acknowledgments, Information Extraction, Disambiguation

[1]An early acknowledgement indexing system was built in CiteSeer but was not refactored into the new CiteSeerX
[2]`http://ackseer.ist.psu.edu`

1. INTRODUCTION

While author information and affiliation are considered to be sufficient metadata to describe the creators of items in digital libraries, there is still plenty of information available from acknowledgment sections of papers and books, much of which is not readily accessible. Acknowledgments are rich with information about indirect contributors and collaborators of the work which lead one researcher to call acknowledgments "Super Citations" [13]. Cronin went further and argued that the first acknowledged person might have contributed to the work more than the last author[9]. Acknowledged contributors maybe who funded the work.

An early study dissected the different types of acknowledgments found in scholarly work, and found that the most common types were:"

- Moral support
- Financial support
- Access (facilities, data ..etc.)
- Clerical support
- Technical support (programming, statistics.. etc)
- Peer interactive communication " [10, 12]

These categories of acknowledgments may indicate significant contribution to the published work, given that such an entity was mentioned in the acknowledgments. While the promotion and tenure process in academic institutions is mostly affected by citations and authorship, acknowledgments can be argued to have enough credit that could possibly affect promotion and tenure decisions [9]. In fact, the previously mentioned work argues that the the most significant reason keeping acknowledgments out of consideration for promotion and tenure can be traced back to the lack of a central repository that keeps track of acknowledged entities in scientific articles. Such a repository, which would be similar to the Institute for Scientific Information (ISI), would be imperative for studying acknowledgments on a larger scale, and could possibly contribute to the measuring academic influence.

Acknowledgments are domain dependent, in the sense that different communities tend to have different forms and structure of constructing the list of acknowledged entities. The venue plays a significant role in determining what should be acknowledged and what should not. For example, a PhD dissertation would probably have larger section for acknowledgments than a journal article or a conference paper. In the medical domain, studies can be a result of multicentral clinical trials where the number of authors can go beyond double digits as a result of collaboration. In this case, a significant number of contributors cannot be listed as authors, and hence their contribution is credited in the acknowledgments section. Another trend in medical papers is to list the primary investigators as authors with the name of the study group as a co-author and the members of the study group are listed in the acknowledgments section. In 1991, *The New England Journal of Medicine* published an editorial to describe the guidelines of acknowledgments and authorships after accepting a manuscript in which five pages of the article's twelve were acknowledgments [18].

To the best of our knowledge, there is no active system that keeps track of acknowledgments in digital libraries. Therefore, our work is an important step towards making acknowledgments more accessible for researchers, academic evaluators, and economists for many uses such as promotion decisions and the effectiveness of funded research[15]. The *AckSeer* architecture is built with domain independence in mind and can therefore be used to index acknowledgments from various domains even though our experiments are limited to documents primarily in computer and information science.

In addition, analyzing and understanding the the co-authorship network would benefit a great deal by incorporating acknowledgment information within the graph, since persons and organizations being acknowledged have interacted with the authors and contributed their share to the published work. Hence, a bigger graph comprising both authors and acknowledged entities would be very interesting to study, providing an additional feature to a co-authorship network that has already been extensively studied.

Our contributions are several. We design and describe the architecture of acknowledgments repository and search engine. We introduce methods for extracting acknowledgment sections as well as the entities mentioned within, and list the top acknowledged entities in computer and information science. We use a novel approach to cluster entities based on search engine results. Finally, we introduce different ranking functions that can be used in an acknowledgments search engine.

The remainder of this paper is organized as follows. In section two we survey related work. Section three describes the architecture of *AckSeer* and the system components. The extraction method along with the disambiguation algorithm and ranking functions are described in section four. In section five we discuss the experiments and the statistics obtained. We conclude in section six and identify directions for future work.

2. RELATED WORK

The related work can be categorized into two different components, first studying acknowledgments in scientific publications whether it has been extracted manually or automatically. The second component is concerned with building repositories and search engines for sub-objects of information in digital libraries such as tables, figures and acknowledgments that can be found inside articles and books.

Acknowledgments have received a fair amount of interest from sociologists, information scientists, and to some level from computer scientists. The attention given to acknowledgments has not been comparable to that given to citations and authorship. However, it was suggested far before automatic citation indexing became feasible (i.e. Google Scholar[3], CiteSeer[14]).

Early work on acknowledgments was carried out manually. Cronin et al. [9] examined prominent journals of sociology for 10 years period manually extracting acknowledgments. As part of their work they conducted the same study but on psychology and philosophy journals for a 100 years span [11]. Other fields received attention as well including history [26] , information sciences [7], and humanities [8].

Recent work [6, 15] proposed extracting acknowledgments automatically and indexing them using the CiteSeer digital library as an example. That work is most similar to ours as they extract acknowledgments along with their entities and make them searchable. However, our work builds a standalone repository and search engine for acknowledgments, using improved entity extraction techniques. Furthermore, our passage extraction algorithm achieves a higher F1 measure due to much better recall score of our algorithm (90% on average) while they achieved a recall around 67% [4]. As mentioned, we use multiple state-of-the-art Named Entity Recognition (NER) tools to identify entities in the acknowledgments section. We introduce a method to utilize results from multiple NER instances and merge them. This is very useful as some NERs are better in extracting person names while others are superior in identifying organizations. Their NER was based on hand written rules that are only capable of extracting certain organization names but not person names. In both cases we are able to outperform their work and also propose a novel approach for entity merging and clustering.

Extracting acknowledgments from digital libraries can be considered another variation of information and metadata extraction where multiple approaches have been proposed including regular expressions, rule based extractors, and machine learning methods. Regular expressions and rule-based systems can perform well in certain problems as long as the data does not change but are tedious to maintain when new data becomes available. As in that work [6, 15] we use regular expressions to identify acknowledgment passages since this extraction can be robust. But in the case of NERs, those

[3]http://scholar.google.com
[4]The values were achieved on different datasets because the data they used in experiments was lost along with the code. Hence, we used a different dataset that includes the top cited papers of all time which is more likely to be in their dataset

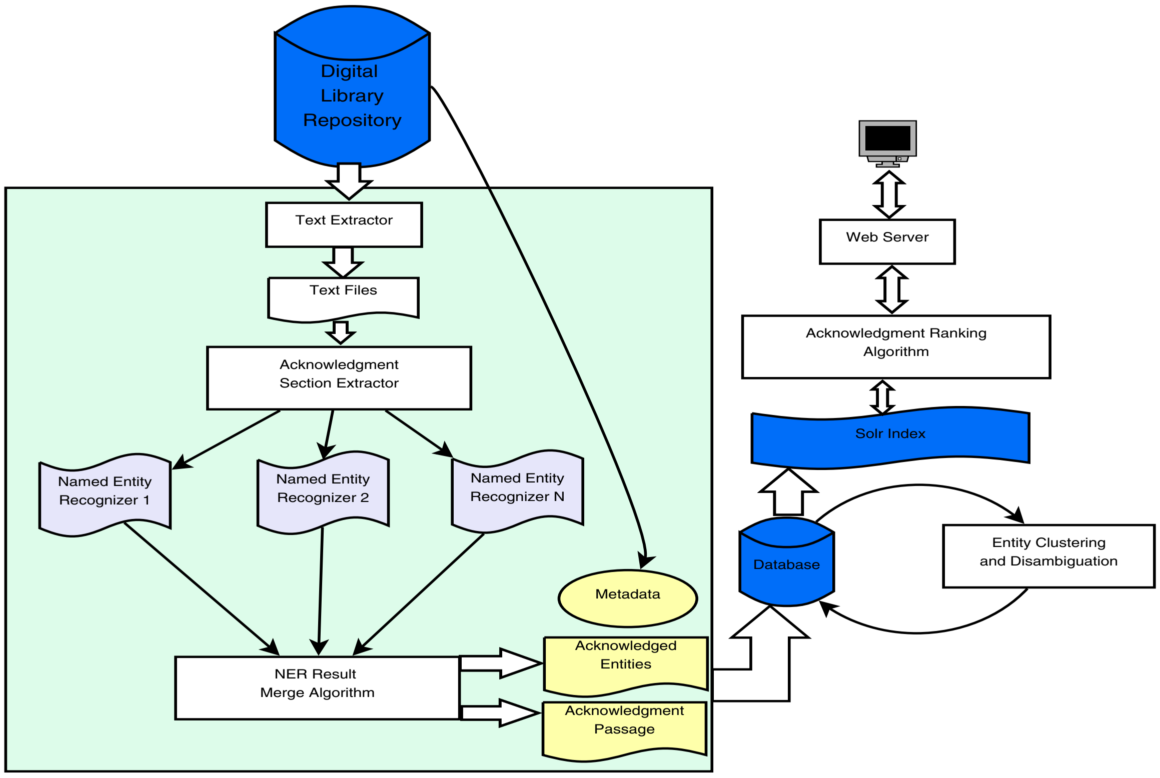

Digital Library Repository
Text Extractor
Text Files
Acknowledgment Section Extractor
Named Entity Recognizer 1
Named Entity Recognizer 2
Named Entity Recognizer N
Metadata
NER Result Merge Algorithm
Acknowledged Entities
Acknowledgment Passage
Web Server
Acknowledgment Ranking Algorithm
Solr Index
Database
Entity Clustering and Disambiguation

Figure 1: The Architecture of AckSeer

based on machine learning methods such as Hidden Markov Models (HMM)[32] and Conditional Random Fields (CRF) [27] have been applied successfully to multiple domains reducing the need to hand-craft rules and to manually extract entities. Other machine learning techniques like Support Vector Machines (SVM) are very popular in classifying high dimensional data , and information extraction can be cast as a classification problem [5]. Han et al. [16] used SVMs to automatically extract metadata in building digital libraries, while others have applied SVM for chunk identification [22] and named entity recognition [17, 28].

Repositories of sub-objects are gaining popularity due to the need to browse and search these sub-objects independently of their parent-objects (i.e. paper, book) and since they might represent summaries of the work discussed within the article. Searching for figures, tables, algorithms and chemical formulas are examples of sub-object repositories made possible. Tables have been extracted from digital libraries and repository was built that indexs tables and ranks them with a novel ranking algorithm [24, 23]. Along with tables, plots and figures are often used to provide measures, comparisons and results of experiments, which has led to an interest in extracting these plots and associated data [19, 25]. While summaries are usually reported in tables and figures, computer scientists and software developers are also interested in algorithms and pseudo code. Bhatia et al. [3, 4] built a search engine for extracting and indexing algorithms found in the computer science literature.

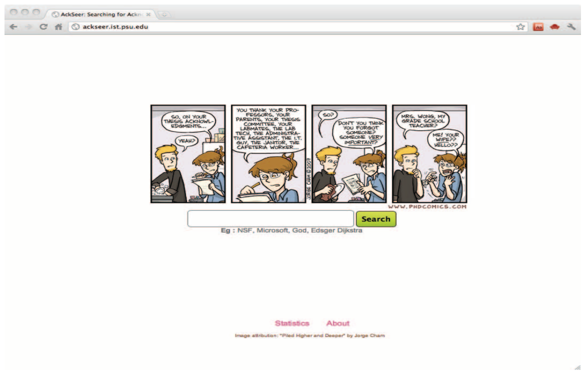
3. ARCHITECTURE

3.1 System Overview

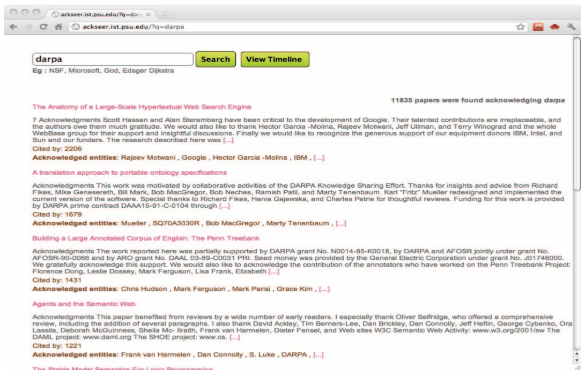
Figure 1 shows the architecture of *AckSeer*. The system can be built for any digital library that can provide documents in PDF format along with their metadata. These files are then converted to text using a converter such as PDFbox and the output is passed to the section extraction algorithm which is discussed in section 4.1. If acknowledgments are found within the documents, the extracted passage would be passed to multiple NERs to extract the entities mentioned. Currently, we use OpenCalais[2] and AlchemyAPI[1] as NERs but there is no limit on the number of NERs that can be used simultaneously. The results from the NERs are aggregated and merged together removing duplicates and ignoring entities that are contained in other NERs results. The resulting acknowledgment passage along with the entities and the metadata generated by NERs and given by the digital library are stored in the database. The entity clustering and disambiguation process is run in batches on the entities within the database to merge variations of the same entity. For example, NSF and National Science Foundation are merged into the same cluster. The entities are then indexed with Solr/Lucene along with metadata of the containing document and the acknowledgment passage and become searchable through the web interface.

3.2 User Interface

The user interface resembles most standard search engines in order to keep the system familiar and easy to use. Figure 2 shows different screenshots from the user interface. The


AckSeer: Searching for Ackn
ackseer.ist.psu.edu
SO, ON YOUR THESIS ACKNOWLEDGMENTS...
YEAH?
YOU THANK YOUR PROFESSORS, YOUR PARENTS, YOUR THESIS COMMITTEE, YOUR LABMATES, THE LAB TECH, THE ADMINISTRATIVE ASSISTANT, THE I.T. GUY, THE JANITOR, THE CAFETERIA WORKER...
JORGE CHAM © 2007
SO?
DON'T YOU THINK YOU FORGOT SOMEONE? SOMEONE VERY IMPORTANT?
MRS. WONG, MY GRADE SCHOOL TEACHER?
ME! YOUR WIFE?? HELLO??
WWW.PHDCOMICS.COM
Search
Eg : NSF, Microsoft, God, Edsger Dijkstra
Statistics
About
Image attribution: "Piled Higher and Deeper" by Jorge Cham

(a) The landing page of AckSeer


ackseer.ist.psu.edu/?q=darpa
darpa
Search
View Timeline
Eg : NSF, Microsoft, God, Edsger Dijkstra

11835 papers were found acknowledging *darpa*

The Anatomy of a Large-Scale Hypertextual Web Search Engine

7 Acknowledgments Scott Hassan and Alan Steremberg have been critical to the development of Google. Their talented contributions are irreplaceable, and the authors owe them much gratitude. We would also like to thank Hector Garcia -Molina, Rajeev Motwani, Jeff Ullman, and Terry Winograd and the whole WebBase group for their support and insightful discussions. Finally we would like to recognize the generous support of our equipment donors IBM, Intel, and Sun and our funders. The research described here was [...]

Cited by: 2206

Acknowledged entities: Rajeev Motwani , Google , Hector Garcia -Molina , IBM , [...]

A translation approach to portable ontology specifications

Acknowledgments This work was motivated by collaborative activities of the DARPA Knowledge Sharing Effort. Thanks for insights and advice from Richard Fikes, Mike Genesereth, Bill Mark, Bob MacGregor, Bob Neches, Ramish Patil, and Marty Tenenbaum. Karl "Fritz" Mueller redesigned and implemented the current version of the software. Special thanks to Richard Fikes, Hania Gajewska, and Charles Petrie for thoughtful reviews. Funding for this work is provided by DARPA prime contract DAAA15-91-C-0104 through [...]

Cited by: 1679

Acknowledged entities: Mueller , SQ70A3030R , Bob MacGregor , Marty Tenenbaum , [...]

Building a Large Annotated Corpus of English: The Penn Treebank

Acknowledgments The work reported here was partially supported by DARPA grant No. N0014-85-K0018, by DARPA and AFOSR jointly under grant No. AFOSR-90-0066 and by ARO grant No. DAAL 03-89-C0031 PRI. Seed money was provided by the General Electric Corporation under grant No. J01746000. We gratefully acknowledge this support. We would also like to acknowledge the contribution of the annotators who have worked on the Penn Treebank Project: Florence Dong, Leslie Dossey, Mark Ferguson, Lisa Frank, Elizabeth [...]

Cited by: 1431

Acknowledged entities: Chris Hudson , Mark Ferguson , Mark Parisi , Grace Kim , [...]

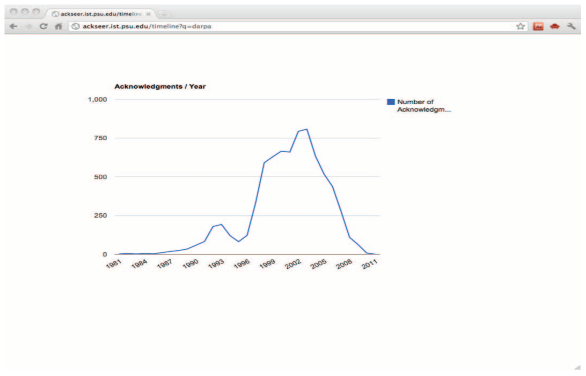
Agents and the Semantic Web

Acknowledgments This paper benefited from reviews by a wide number of early readers. I especially thank Oliver Selfridge, who offered a comprehensive review, including the addition of several paragraphs. I also thank David Ackley, Tim Berners-Lee, Dan Brickley, Dan Connolly, Jeff Heflin, George Cybenko, Ora Lassila, Deborah McGuinness, Sheila Mc- Ilraith, Frank van Harmelen, Dieter Fensel, and Web sites W3C Semantic Web Activity: www.w3.org/2001/sw The DAML project: www.daml.org The SHOE project: www.cs. [...]

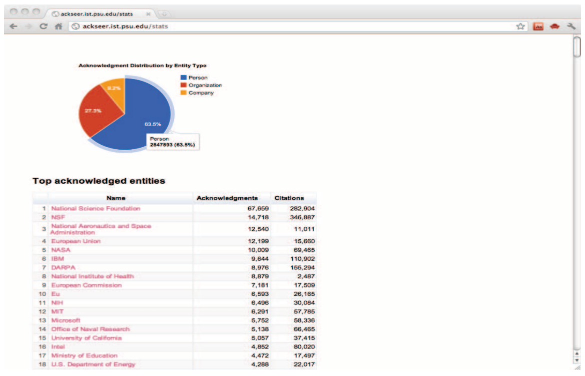
Cited by: 1221

Acknowledged entities: Frank van Harmelen , Dan Connolly , S. Luke , DARPA , [...]

(b) The results page for a specific query



(c) Timeline view for the query DARPA.


ackseer.ist.psu.edu/stats
Acknowledgment Distribution by Entity Type
Person
Organization
Company
9.2%
27.3%
63.5%
Person
2847893 (63.5%)

Top acknowledged entities

	Name	Acknowledgments	Citations
1	National Science Foundation	67,659	282,904
2	NSF	14,718	346,887
3	National Aeronautics and Space Administration	12,540	11,011
4	European Union	12,199	15,660
5	NASA	10,009	69,465
6	IBM	9,644	110,902
7	DARPA	8,976	155,294
8	National Institute of Health	8,879	2,487
9	European Commission	7,181	17,509
10	Eu	6,593	26,165
11	NIH	6,496	30,084
12	MIT	6,291	57,785
13	Microsoft	5,752	58,336
14	Office of Naval Research	5,138	66,465
15	University of California	5,057	37,415
16	Intel	4,852	80,020
17	Ministry of Education	4,472	17,497
18	U.S. Department of Energy	4,288	22,017

(d) The statistics page of Ackseer

Figure 2: Screenshots of AckSeer

landing page shown in figure 2(a) [5] is where a query can be submitted to the system. There are multiple types of queries which the system is capable of handling. The default query mode would try to match the terms of the query against the words of the acknowledgment passage in a digital document. Alternatively, the user may specify the field against which the query is matched by preceding the query terms with the field name followed by colon. Available fields are *entities* and *title*. The query "entities:NSF"would search for NSF in the list of extracted entities of each acknowledgment passage, while the query "title:Google" would search for acknowledgments appearing in documents that have the word Google in its title. The results page for a query is shown in figure 2(b). For each matching acknowledgment the metadata shown includes the title of the containing paper, the number of citations the paper received, the list of extracted entities, and the acknowledgment passage itself.

For a specific query q it is possible to view the timeline of papers that contain q in the acknowledgements section. The timeline is a plot where the x axis represents the year and the y axis represents the number of papers published in that year where q appeared in the acknowledgments. Figure 2(c) shows the timeline for the query *DARPA*. The last interface of *AckSeer* is the statistics page which shows a pie chart of the distribution of entity types in the repository. Further, we maintain four lists showing the top acknowledged entities, persons, companies, and organizations where each list presents the entities along with the number of times they have been acknowledged and the number of citations these papers acknowledging them have received.

[5]The image on AckSeer homepage is courtesy of PhD Comics `http://www.phdcomics.com/comics/archive.php?comicid=870`

4. ACKNOWLEDGMENT EXTRACTION

In this section we describe the methods deployed in extracting acknowledgment passages and entities. After that we introduce a novel approach to cluster and disambiguate entities appearing in acknowledgments. Eventually we propose ranking functions that can be used while searching for acknowledgments.

4.1 Passage Extraction

Identifying acknowledgment sections is a crucial part of *AckSeer* and is the first step towards building the repository. Errors encountered during the extraction would carry through the indexing pipeline; therefore, we try to mitigate the effect of accumulated errors by building robust passage extractors. Previous work on acknowledgments extraction used regular expressions achieving a high precision , 99%, but relatively moderate recall, 67% [6, 15]. Inspired by that work we devised an algorithm to identify the different sections and chapters of digital documents mainly though regular expressions, since most of documents in digital libraries including papers and books are semi-structured.

To evaluate an extraction algorithm we need a tagged dataset or gold standard on which we can measure the performance of different extraction methods. To the best of our knowledge, there is no tagged dataset for acknowledgments extraction. Our constructed dataset contains the top 200 cited papers/books in the CiteSeerX[6] digital library. Highly cited papers were chosen so that the performance of the extractors would be biased towards higher quality papers and books, assuming that the number of citations is correlated with the quality of the work. Among the 200 papers that we manually examined, 130 of them had an acknowledgments section accounting for about 65% of the dataset. We use the following tags to identify objects appearing in the acknowledgment passage, while of course others could be considered:

- Acknowledgment: the acknowledgments section
- Person: indicates a person being acknowledged
- Entity: indicate an agency or company being acknowledged
- Scholarship: acknowledging a scholarship
- Project: acknowledging a project
- GrantNum: grant number
- ContractNum: contract number

Listing 1: Example of a tagged acknowledgment

```
<acknowledgment>
ACKNOWLEDGEMENTS W.M. and Z.Z. are
supported by grant <grantNum> LM05110
</grantNum> from the <entity>National
Library of Medicine </entity>. We thank
Dr <person> Warren Gish</person> for
helpful conversations , Dr <person> Eugene
 Koonin</person> for assistance with the
examples , and Dr <person> Gregory Schuler
</person> for producing several of the
figures . </acknowledgment>
```

After examining numerous papers and books in the CiteseerX digital library to understand how and where authors tend to write the acknowledgments section, we devised an algorithm that utilized regular expressions to segment each paper/book into sections (chapters for books). The header or the title of each section is then filtered, and headers referring to acknowledgments are stored. This is a modified version of the section identification method used in [31]. Algorithm 1 shows the pseudocode of the section extraction process. Briefly, the algorithm checks each line in the document against a predefined set of regular expressions that detect section headers. If the line is considered a match, a new section is detected and all the lines following the header are considered section content. A new section is started whenever a content line matches the header regular expressions. Eventually, all identified section headers are examined and if acknowledgments were to be found between them, the content is extracted. The algorithm is linear in both time and space making it quite scalable on large numbers of documents.

The regular expression based extraction algorithm yields hard precision and hard recall values at 83.7% and 83.8% respectively. The hard precision and hard recall is defined such that an extracted acknowledgment is only correct if it

[6] `http://citeseerx.ist.psu.edu`

Algorithm 1: Section extraction algorithm

```
input  : A text document $D$
output: Acknowledgment section
$Sections \longleftarrow HashTable$;
$header \longleftarrow$ read line ;
$content \longleftarrow \emptyset$;
while not at end of $D$ do
    read line $l$;
    if $l$ matches new section regex then
        Add $header$ and $content$ to $Sections$;
        $header \longleftarrow l$;
    else
        Append $l$ to $content$;
    end
end
for $header, content \in Sections$ do
    if $header$ matches acknowledgment regex then
        return $header, content$;
    end
end
```

Table 1: Performance of the acknowledgment extractor on the training dataset of 200 papers

Measure	Value
Hard Precision	83.7%
Hard Recall	83.8%
Hard F1 score	83.7%
Precision	92.3%
Recall	91.6%
F1 score	91.9%
Precision without books	98.3%
Recall without books	97.6%
F1 score without books	97.9%

is a verbatim match of the manually tagged one. So if the extractor extracted an extra dot or semicolon, the acknowledgment is not considered to be successfully extracted. We found that the extractor sometimes extracted an extra line such as the title of the journal article along with the issue that sometimes appears at the header or footer of the page. This small line doesn't make the extracted acknowledgment necessarily incorrect, as the original acknowledgment is extracted completely but with an extra line. A similar scenario can occur when journal articles list the date on which the paper was received. Based on our definition above, when there is more than one extra line beyond the original acknowledgment, the extracted passage is considered incorrect. We relaxed this restriction and now consider such a passage to be a correctly extracted acknowledgment. Under this relaxed assumption, the value of precision and recall are 92.3% and 91.6% respectively. It's worth noting that our dataset comprises both books and papers, and the errors which our extractor makes are mostly on books, which is expected since books have more variation in their acknowledgement format than papers. If we exclude books, our precision and recall jumps to 98.3% and 97.6% respectively. Table 1 summarizes the performance of the section extractor on the top 200 cited papers in computer science in CiteSeerX digital library.

4.2 Entity Extraction

The next phase in building the acknowledgments repository is to extract the entities inside the acknowledgment section. Named Entity Recognition (NER) has been studied extensively in the literature, and systems have been able to achieve 90% accuracy in many competitions. Instead of building our own NER system, we decided to leverage what is available in open source and free web services. We have examined many open source tools: OpenNLP, NLTK, Apache UIMA, Illinois Named Entity Tagger, OpenCalais[2] and AlchemyAPI[1]. Upon examining the different NERs it became clear that certain extractors perform better on specific types of entities and poorly on other types. For example, some NERs are more suited to extract celebrity names and organizational entities, while they fail to recognize normal person names. Others, however, might do a better job on person names due to their large dictionary of names, but perform poorly on organizations and company entities.

Guided by this, we designed a name extraction module that is capable of consuming results from different NERs and allowing them to work in parallel. The outputs from all the NERs are collected and merged together to generate a single list of entities. The merger is performed as follows: for an acknowledgment passage p, a set of NERs $N_1 ... N_k$ is ran on p, with each N_i generating a list of entities E_i. Then, all the lists $E_1 ... E_k$ are concatenated into single list L. For each entity $e_i \in L$ we remove all verbatim duplicates e_j such that $e_i = e_j$ and $i \neq j$ from the list L. After that, we find and remove all entities e_k such that $e_k \in e_i$ and e_i starts with e_k. In other words we eliminate all substrings of the entity e_i that share the same prefix. This step helps remove sub-identified entities that are not extracted in full form. The problem occurred quite often in our experiments and the suggested solution helped mitigate the issue. For example, while extracting *National Research Council Canada* extractor A successfully identified the entity, while extractor B would only extract *National Research Council*.

In *AckSeer* we chose to use OpenCalais and AlchemyAPI because they provide a web service that can be easily called and provide quality extraction and disambiguation results. OpenCalais is a web service developed by Reuters that provides many knowledge extraction services, but we only use the entity extraction part. AlchemyAPI is another system which provides a plethora of text extraction services including text stripping, language detection, and last but not least Named Entity Extraction. Similar to OpenCalais, we only use the NER part of AlchemyAPI.

In our experiments with both OpenCalais, and AlchemyAPI we found OpenCalais to be superior to AlchemyAPI in identifying and extracting organizations and companies, while AlchemyAPI is better at extracting names of persons. As such, we use both systems to extract the entities and then merge them based on the name of the entity (as it appears in the text). If the entities are well known, OpenCalais and AlchemyAPI would provide disambiguated canonical names, along with links to dbpedia[7] and freebase[8]. However, our experiments with the disambiguation results of both APIs showed clearly that the results are not accurate enough and a standalone disambiguation method is needed. The above mentioned section identification method is similar to what was used in our previous work[20].

[7]`http://dbpedia.org/`

[8]`http://www.freebase.com/`

4.3 Instance Merging and Disambiguation

OpenCalais and AlchemyAPI tag the extracted entities into three categories: person, company, and organization. In the following sections we propose methods to disambiguate and merge instances of the same entity.

4.3.1 Entity Names

Authors have different ways of writing their acknowledged entities; some prefer to use the full name of the entity while others tend to use abbreviations and acronyms. For example, *National Science Foundation*, the most acknowledged entity in our dataset, is frequently written in full form but the acronym *NSF* is also as frequent but never in lower case. To have accurate statistics about each entity we should handle all naming variations and cluster them together. The variations are not only acronyms v.s. full form; sometimes entities are written with a shorter name than the full form. For example, *"Natural Sciences and Engineering Research Council of Canada"* and *"Engineering Research Council"* appear to refer to the same entity, and both are frequently used in acknowledgments. Similar examples include: *"Stanford"* and *"Stanford University"*; *"U.S Federal Reserve"*, *"The Federal Reserve"*, *"United States Federal Reserve"*. Therefore, disambiguating entities is far more complicated than just using a dictionary of acronyms along with their long form. Even popular acronym databases are ineffective in disambiguating acknowledged entities because they are built by matching a letter from the acronym with the beginning of a word in the full form. These methods would fail to recognize many acronyms especially if the acronym stems from languages other than English. For example *German Research Foundation*, which is the German NSF equivalent, is acknowledged in three different formats: 1) *DFG* , 2) *Deutsche Forschungsgemeinschaft*, 3) *German Research Foundation*. Predefined acronym databases fail to recognize that all the three forms refer to the same entity.

Earlier work on acknowledgment indexing used manually crafted dictionaries to match and merge instances of the same entity [6]. Building a dictionary is a very labor intensive task, and simply not feasible because our repository contains more than 4 million mentioned entities. Given the aforementioned challenges in disambiguating entities we explored other approaches that utilize outside world information for clustering, and at the same time scale to the size of our data. Hence, we propose an approach that relies on search engine results to cluster entities. The basic idea is that variations of the same entity should have similar search results when querying a search engine like Google or Bing. For example, searching Bing for "NSF" or "National Science Foundation" returns *www.nsf.gov* in both queries. Search engines invest a lot of resources in disambiguating queries and clustering them to return the best results. Using a search engine API would allow us to access this information and use it for clustering. In this work Bing API[9] is used to create a profile for each entity, where the profile contains the top ten results returned by Bing when querying for that entity. A similar approach [29] to our work uses search engine results to disambiguate authors of research papers. We differ in how these results are used.

[9]`http://www.bing.com/toolbox/bingdeveloper/`

For each entity e_i in our repository where $i \in 1..n$, we create a list l_i containing the top ten urls returned by Bing API when querying for e_i. For each domain d appearing as the first result on l_i we create a new cluster d and add e_i to it. If that cluster already exists, we just add e_i to it. The resulting cluster d contains all the entities whose first search result is on on the domain d. Algorithm 2 shows the pseudo-code of our approach. The variable k in algorithm 2 controls the number of clusters that e_i get assigned to. Currently we use $k = 1$ to use the first search result only.

Algorithm 2: Entity Disambiguation Algorithm

```
input  : Set of entities $Entities$ and list of domains $L_e$
$Clusters \longleftarrow HashTable$;
for $e \in Entities$ do
    $u \longleftarrow L_e[0]$;
    if $u \in Clusters$ then
        Add $e$ to $Clusters[u]$;
    else
        Create new cluster $Clusters[u]$;
        Add $e$ to $Clusters[u]$;
    end
end
for $e \in Entities$ do
    for $i \leftarrow 1$ to $k$ do
        if $L_e[i] \in Clusters$ then
            Add $e$ to $Clusters[L_e[i]]$;
        end
    end
end
return $Clusters$;
```

Our search engine driven disambiguation method showed promising results, and is very fast since it only needs to scan the list of entities once, yielding linear run time, assuming the search results are available. We generate the list of url results in a separate batch job before running the disambiguation method. This linearity allows us to process the entire repository of more than 4 million entities in few minutes.

Future work could explore the number of URLs that can be used for each entity (the variable k in the algorithm). Currently, we only use the first search result for efficiency, but in many cases we found that the correct cluster for the url is not the first one. Our estimate suggests that using the first url increases the precision but decreases the recall. Other potential future work is to consider the results as features in a high dimension space and apply a hierarchical clustering algorithm.

4.3.2 Person Names

Person disambiguation in the case of acknowledgments can be considered a form of author disambiguation. The assumption is reasonable given the patterns of acknowledgments described in the introduction. As of now we do not perform any person specific disambiguation however one could treat acknowledged persons as coauthors and use random forest based author disambiguation [30]. Errors here do occur and *John Wiley* is classified as a person.

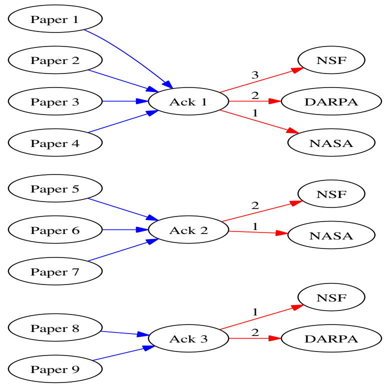


Figure 3: Graph of citations and acknowledgments used for ranking

4.4 Searching and Ranking

AckSeer provides a search interface to access millions of acknowledged entities that were extracted from more than 500,000 papers and books. An effective ranking algorithm is required to navigate through the hundred of thousands of acknowledgments. Traditional ranking methods based on term frequency and inverse document frequency are not effective here because an entity appears only once inside the acknowledgment passage. TF-IDF based ranking would be no different than a database query returning all matching records in no specific order. We propose an approach to rank acknowledgments which takes advantage of the number of citations received by the document containing the acknowledgment. The intuition is that citations can be considered as incoming credit to the paper, while acknowledgments are outgoing credit. By distributing the incoming credit to the outgoing credit, we would be able to rank the important acknowledgment passages. This method is inspired by the HITS algorithm [21] and earlier rankings of acknowledgements in CiteSeer. Figure 3 shows three acknowledgment passages receiving citations from papers 1 through 9 and acknowledging the entities NSF, DARPA and NASA. The blue edges denote citations while red edges denote acknowledgment. The number on the red edges denotes the order of appearance in the acknowledgment passage.

4.4.1 Citations

Given a query Q, a list of matching acknowledgments L is retrieved from the inverted index and the returned result is ranked by the inverse order of the citations count. With this method citations are distributed evenly on all acknowledged entities. In the example shown in figure 3, given the query *NSF* the returned result will be $[Ack1, Ack2, Ack3]$.

4.4.2 Citations Inverse Entity Count Boost

The Citations Inverse Entity Count Boost ($CIECB$) ranks acknowledgments by citations by first boosting the matching entities with the reciprocal of the entities count in the acknowledgment. So for a given query Q and matching acknowledgment passage A, the similarity score is computed:

$Sim(Q, A) = Citations(A) * \frac{1}{|entities \in A|}$

In the same example of figure 3 and the query *NSF*, the resulting order would be $[Ack2, Ack1, Ack3]$

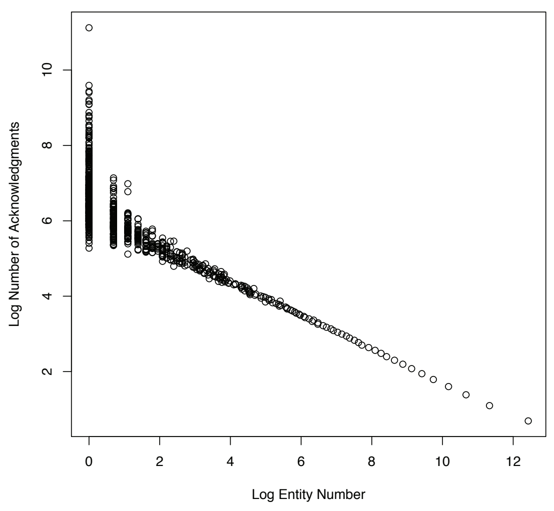


Figure 4: A scatter plot in log log scale for number of acknowledgments against the number of entities

4.4.3 *Citations Inverse Entity Count with Order Boost*

The Citations Inverse Entity Count with Order Boost ($CIECOB$) assumes that acknowledged entities are similar to authors, hence the order of appearance matters. The first acknowledged entity should receive higher credit than the subsequent entities, just like in the case of authorship. The similarity function from $CIECB$ is augmented as follows:

$Sim(Q, A) = Citations(A) * \frac{1}{|entities \in A|} * \frac{1}{Order(Q) in A}$

Going back to our example in figure 3 and the query NSF, the result in this case would be $[Ack3, Ack2, Ack1]$. Using $CIECOB$ we are able to capture the importance of order, count, and citations in a single ranking scheme.

5. EXPERIMENTS AND ANALYSIS

5.1 Repository

In our experiments of *AckSeer* we use the repository of CiteSeerX digital library which for this work contained more than 1.5 million documents in computer science, math and statistics. The section extractor identified acknowledgments in 526,930 of the papers in the repository. From the extracted acknowledgments we extracted 4,486,134 entity mentions, 1,648,013 of which were unique textually. The distribution of entity types was as follows: 2,847,893 persons, 1,225,947 organizations, and 412,294 companies. We ranked the top acknowledged entities, and our results are similar to [15, 6] in showing that National Science Foundation is the top acknowledged entity. Table 2 shows the top acknowledged entities with manual disambiguation of top results. It also shows the number of citations received from by the papers that acknowledge these entities. The last column is the result of dividing the citations by the acknowledgments. These statistics are an extension of what we found in our previous work [20].

As noted in table 2, the top acknowledged entities are all organizations and companies. We extracted the top acknowledged persons in our dataset, and noticed that there

Table 2: Top acknowledged entities in CiteSeerX

Entity	# Acks	# Cites	Cites/Acks
NSF	67659	282904	4.18
NASA	12540	11011	0.88
European Union	12199	15660	1.28
IBM	9644	110902	11.50
DARPA	8976	155294	17.30
NIH	8879	2487	0.28
European Commission	7181	17509	2.44
MIT	6291	57785	9.19
Microsoft	5752	58336	10.14
Office of Naval Research	5138	66465	12.94

Table 3: Top acknowledged persons in CiteSeerX

Entity	Number of Acks	H-Index
Oded Goldreich	471	37
Olivier Danvy	350	24
Avi Wigderson	325	29
Vern Paxson	325	36
David Wagner	313	32
Jim Gray	309	21
Paul Taylor	292	14
Sally Floyd	288	40
Jon Crowcroft	270	25

is a relation between the h-index[10] of the person and the number of time she/he has been acknowledged. We use the h-index values computed by CiteSeerX. We leave further analysis of this observation to future work.

5.2 Disambiguation

We evaluate our disambiguation algorithm by constructing a dataset that consists of the top 500 acknowledged entities in *AckSeer*. As our baseline, we implement the longest common subsequence algorithm (LCS). The LCS algorithm is a common method for sequence alignment. It uses dynamic programming to compute optimal alignment between two sequences of characters. We use simple heuristics to separate acronyms from non-acronym entity names. For each non-acronym name we generate its potential acronym, by taking the first letter of each word in the name. To determine if an acronym name s matches a target name t, which can be either another acronym or a full name, we compute the longest common subsequence between s and t (or the potential acronym of t, if t is a full name). If the length of the longest common subsequence is greater than 0.7 of the length of the longer acronym, they are considered to match, , for instance, between 'USAF' and 'US Air Force.' On the other hand, for two non-acronym names to match, one has to be a substring of the other, e.g. 'US Department of Energy' and 'Department of Energy.' We compute the pairwise LCS score between all the elements in our dataset. The clusters are created such that all the entities in cluster c have pairwise LCS score > 0.7. A cluster c is considered correct if all the entities inside that cluster refer to the same entity. In our evaluation we only consider clusters that have more

[10]An author would have h-index = X if he has X papers each of which has been cited at least X times

than one entity. LCS generates 89 clusters that have *support*[11] > 1. We manually inspected these clusters and found out that 49 of them are correct resulting in 55% accuracy.

Next we evaluate the accuracy of our search engine based algorithm. The search results are retrieved for entities in the dataset and we run algorithm 2 with default parameter values. We manually examine the generated clusters for accuracy, that is the purity within the cluster. Overall, the algorithm detects 345 clusters with 70 of them having *support* > 1. Among the 70 clusters, 65 of them were correct resulting in an accuracy of 92.8%. That is a significant improvement over the base line.

The following list shows some of the correctly detected clusters:

- **www.bmbf.de**: {BMBF , German Federal Ministry of Education and Research, German Ministry of Education}
- **www.defense.gov**: { Department of Defense, U.S. Department of Defense, DoD}
- **www.usa.gov**: {U.S. government, United States Government, US Government, federal government }

Some of the mistakes that the algorithm makes can be traced back to the generality of the entity. For example our algorithm clusters the entity **"Department of Electrical Engineering"** with the *princeton.edu* cluster because their department of electrical engineering appears as the first result on Bing, and the url of the result is on the princeton domain (not a subdomain). Future work would research the trade-off between using domain name and full URL in building the clusters.

5.3 Acknowledgments Social Graph

In our previous work we have studied a small graph [20] of the acknowledgment network where the nodes are authors and acknowledged entities. The edges are created between an author a whose paper p acknowledges entity e. Our experiment was conducted on the graph resulting from the top 1000 cited papers in CiteSeerX. We examined a few social network measures on the small graph such as degree distribution, centrality and clustering coefficients where we found out that the in-degree and out-degree follow a power law distribution.

6. CONCLUSIONS AND FUTURE WORK

In this paper we introduced *AckSeer*, the repository and search engine for automatically extracted acknowledgments. We described the architecture of an acknowledgment search engine that is domain independent and repository independent. Our section extraction algorithm is based on regular expressions but it is robust enough with average F1 score 85%. We extract entities from the acknowledgments using multiple NERs and introduce a method to merge their results. In addition, we disambiguate and merge instances of the same entity using a novel search engine based disambiguation algorithm. The nature of acknowledgments and entity frequencies lead us to devise new ranking functions that utilize the importance of the paper and its citations.

[11]The support of the cluster denotes the number of elements in it

AckSeer was built on top of CiteSeerX digital library with more than 500,000 acknowledgments and 4 million entities. Future work could take many directions, i.e. exploration of the semantics of acknowledgments in order to understand the reasoning behind mentioned entities. Other disambiguation and NER techniques could be investigated due to current limitations in accuracy and disambiguation. Better entity clustering could explored and large heterogeneous acknowledgement graphs might give social insights.

7. ACKNOWLEDGMENTS

We thank Pradeep Teregowda and Sharon Koppman for the useful discussions and suggestions. We gratefully acknowledge AlchemyAPI for increasing our requests limit, and partial funding from NSF.

8. REFERENCES

[1] AlchemyAPI. `http://www.alchemyapi.com/`.

[2] OpenCalais. `http://www.opencalais.com/`.

[3] S. Bhatia, P. Mitra, and C. L. Giles. Finding algorithms in scientific articles. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 1061–1062. ACM, 2010.

[4] S. Bhatia, S. Tuarob, P. Mitra, and C. L. Giles. An algorithm search engine for software developers. In *Proceedings of the 3rd International Workshop on Search-Driven Development: Users, Infrastructure, Tools, and Evaluation*, SUITE '11, pages 13–16, 2011.

[5] H. Chieu and H. Ng. A maximum entropy approach to information extraction from semi-structured and free text. In *Proceedings of the National Conference on Artificial Intelligence*, pages 786–791. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2002.

[6] I. Councill, C. Giles, H. Han, and E. Manavoglu. Automatic acknowledgement indexing: expanding the semantics of contribution in the citeseer digital library. In *Proceedings of the 3rd international conference on Knowledge capture*, pages 19–26. ACM, 2005.

[7] B. Cronin. Acknowledgement trends in the research literature of information science. *Journal of Documentation*, 57(3):427–433, 2001.

[8] B. Cronin, G. McKenzie, and L. Rubio. The norms of acknowledgement in four humanities and social sciences disciplines. *Journal of Documentation*, 49(1):29–43, 1993.

[9] B. Cronin, G. McKenzie, L. Rubio, and S. Weaver-Wozniak. Accounting for influence: Acknowledgments in contemporary sociology. *Journal of the American Society for Information Science*, 44(7):406–412, 1993.

[10] B. Cronin, G. McKenzie, and M. Stiffler. Patterns of acknowledgement. *Journal of Documentation*, 48(2):107–122, 1992.

[11] B. Cronin, D. Shaw, and K. La Barre. A cast of thousands: Coauthorship and subauthorship collaboration in the 20th century as manifested in the scholarly journal literature of psychology and philosophy. *Journal of the American Society for Information Science and Technology*, 54(9):855–871, 2003.

[12] C. Davis and B. Cronin. Acknowledgments and intellectual indebtedness: A bibliometric conjecture. *Journal of the American Society for Information Science*, 44(10):590–592, 1993.

[13] D. Edge. Quantitative measures of communication in science: A critical review. *History of Science*, 17:102–134, 1979.

[14] C. Giles, K. Bollacker, and S. Lawrence. Citeseer: An automatic citation indexing system. In *Proceedings of the third ACM conference on Digital libraries*, pages 89–98. ACM, 1998.

[15] C. Giles and I. Councill. Who gets acknowledged: Measuring scientific contributions through automatic acknowledgment indexing. *Proceedings of the National Academy of Sciences of the United States of America*, 101(51):17599, 2004.

[16] H. Han, C. L. Giles, E. Manavoglu, H. Zha, Z. Zhang, and E. A. Fox. Automatic document metadata extraction using support vector machines. In *Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries*, JCDL '03, pages 37–48, Washington, DC, USA, 2003. IEEE Computer Society.

[17] H. Isozaki and H. Kazawa. Efficient support vector classifiers for named entity recognition. In *Proceedings of the 19th international conference on Computational linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[18] J. Kassirer and M. Angell. On authorship and acknowledgments. *New England Journal of Medicine*, 325(21):1510–1512, 1991.

[19] S. Kataria, W. Browuer, P. Mitra, and C. L. Giles. Automatic extraction of data points and text blocks from 2-dimensional plots in digital documents. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 2*, AAAI'08, pages 1169–1174. AAAI Press, 2008.

[20] M. Khabsa, S. Koppman, and C. Giles. Towards building and analyzing a social network of acknowledgments in scientific and academic documents. *Social Computing, Behavioral-Cultural Modeling and Prediction*, pages 357–364, 2012.

[21] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[22] T. Kudoh and Y. Matsumoto. Use of support vector learning for chunk identification. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational natural language learning-Volume 7*, pages 142–144. Association for Computational Linguistics, 2000.

[23] Y. Liu, K. Bai, P. Mitra, and C. Giles. Tablerank: A ranking algorithm for table search and retrieval. In *Proceeding Of The National Conference On Artificial Intelligence*, volume 22, page 317. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

[24] Y. Liu, K. Bai, P. Mitra, and C. L. Giles. Tableseer: automatic table metadata extraction and searching in digital libraries. In *Proceedings of the 7th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '07, pages 91–100, New York, NY, USA, 2007. ACM.

[25] X. Lu, J. Wang, P. Mitra, and C. Giles. Automatic extraction of data from 2-d plots in documents. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 1, pages 188 –192, sept. 2007.

[26] L. Scrivener. An exploratory analysis of history students' dissertation acknowledgments. *The Journal of Academic Librarianship*, 35(3):241–251, 2009.

[27] B. Settles. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, pages 104–107. Association for Computational Linguistics, 2004.

[28] K. Takeuchi and N. Collier. Use of support vector machines in extended named entity recognition. In *proceedings of the 6th conference on Natural language learning-Volume 20*, pages 1–7. Association for Computational Linguistics, 2002.

[29] Y. F. Tan, M. Y. Kan, and D. Lee. Search engine driven author disambiguation. In *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '06, pages 314–315. ACM, 2006.

[30] P. Treeratpituk and C. Giles. Disambiguating authors in academic publications using random forests. In *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, pages 39–48. ACM, 2009.

[31] P. Treeratpituk, P. Teregowda, J. Huang, and C. Giles. Seerlab: A system for extracting key phrases from scholarly documents. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 182–185. Association for Computational Linguistics, 2010.

[32] G. Zhou and J. Su. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 473–480. Association for Computational Linguistics, 2002.