

CITEREP

-

JOURNAL CITATION STATISTICS FOR LIBRARY COLLECTIONS USING DOCUMENT REFERENCE EXTRACTION TECHNIQUES

Author

Steven Verkuil
s.verkuil@alumnus.utwente.nl

Supervisors

Dr. ir. D. Hiemstra
Dr. T. De Schryver

Master Thesis

Submitted July 2016

*Faculty of Electrical Engineering,
Mathematics and Computer Science*

*University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands*

ABSTRACT

Providing access to journals often comes with a considerable subscription fee for universities. It is not always clear how these journal subscriptions actually contribute to ongoing research. This thesis provides a multistage process for evaluating which journals are actively referenced in publications.

Our software tool for journal citation reports, CiteRep, is designed to aid decision making processes by providing statistics about the number of times a journal is referenced in a document set. Citation reports are automatically generated from online repositories containing PDF documents. The process of extracting citations and identifying journals is user and maintenance friendly. CiteRep allows to filter generated reports by year, faculty and study providing detailed insight in journal usage for specific user groups.

Our software tool achieves an overall weighted precision and recall of 66,2% when identifying journals in a fresh set of PDF documents. While leaving open some areas of improvement, CiteRep outperforms the two most popular citation parsing libraries, ParsCit¹ and FreeCite² with respect to journal identification accuracy. CiteRep should be considered for creation of journal citation reports from document repositories.

¹ <http://aye.comp.nus.edu.sg/parsCit/>

² <http://freecite.library.brown.edu/>

ACKNOWLEDGEMENTS

This thesis, along with the CiteRep software, is the result of my Computer Science graduation project at the University of Twente. The past six months have been an absolute pleasure to work on CiteRep. I am very happy with the final software tool and hope it might benefit many; now and in the future.

I would like to thank dr. ir. Djoerd Hiemstra and dr. Tom De Schryver for their supervision and excellent support. We have had many meetings over the past few months and their guidance has made CiteRep into the great tool it now is.

I would also like to thank the librarians at the University of Twente for providing valuable feedback at my first presentation session. Thanks to your feedback I was able to improve the end-user experience of working with the CiteRep web interface.

Many thanks go to dr. ir. Maurice van Keulen and Dennis Vierkant for their supervision in the graduation committee. Our conversations have been very valuable to me. Without Dennis I would not be able to access all the necessary metadata from the document repositories on which CiteRep relies.

Finally, I would like to thank my family and friends for supporting me during the final months of my master's research. They were always available for a motivational talk and provided good distractions when I needed them.

Above all I am grateful to God, the creator of heaven and earth, for knowing Him and providing me with strength for every day!

Enschede, 6 july 2016

Steven Verkuil

CONTENTS

INTRODUCTION.....	1
1.1 CONTEXT	1
1.2 PROBLEM STATEMENT.....	2
1.3 RESEARCH QUESTIONS	3
1.4 APPROACH.....	3
1.5 STRUCTURE.....	4
BACKGROUND.....	5
2.1 RELATED WORK	5
2.2 ADOPTED TECHNIQUES	8
2.3 OPEN-SOURCE CITATION TOOLS.....	8
RESEARCH APPROACH.....	10
3.1 SPECIFICATION OF REQUIREMENTS.....	10
3.2 CITEREP PROCESS ARCHITECTURE	11
3.2.1 Document Acquisition	11
3.2.2 Citation Extraction	12
3.2.3 Journal Identification	13
3.2.4 Journal Normalization.....	13
3.3 EVALUATION METHOD.....	14
3.3.1 CiteRep Document Sets	14
3.3.2 Elsevier Document Sets	15
3.3.3 Cora Dataset	15
3.4 SOFTWARE ARCHITECTURE.....	16
CITATION EXTRACTION	19
4.1 EXTRACTING THE REFERENCE SECTION	19
4.1.1 TrimCorrection	22
4.1.2 NumberedListCorrection	25
4.1.3 BlockListCorrection.....	28
4.1.4 BigTextCorrection.....	29
4.2 CITATION ACCURACY	31

JOURNAL IDENTIFICATION	32
5.1 OVERVIEW	32
5.2 REFPARSE LIBRARY	33
5.3 CITEREP ALGORITHM	34
5.3.1 <i>Select RefParse Candidates</i>	34
5.3.2 <i>Database of Known Journals</i>	36
5.4 IDENTIFICATION ACCURACY	38
JOURNAL NORMALIZATION	39
6.1 OVERVIEW	39
6.1.1 <i>SimpleJournal</i>	39
6.1.2 <i>NormalizeJournal</i>	41
6.2 NORMALIZATION ACCURACY	42
EVALUATION.....	43
7.1 CITEREP OVERALL ACCURACY	43
7.2 JOURNAL USAGE AT THE UNIVERSITY OF TWENTE	44
CONCLUSION	48
FUTURE WORK.....	49
APPENDIX A – COMMUNICATION PROTOCOL.....	51
BIBLIOGRAPHY.....	54

LIST OF FIGURES

Figure 1 Structure of an academic paper.....	2
Figure 2 Illustration of worker and dashboard interaction	16
Figure 3 Desktop interface for reviewing publications in CiteRep	17
Figure 4 Mobile interface for reviewing publications in CiteRep.....	17
Figure 5 Worker performing tasks on a remote machine	18
Figure 6 Corrections used by CiteRep to improve reference extraction	21
Figure 7 Distribution of character counts in repository papers	22
Figure 8 Distribution of citation character counts in repository papers	23
Figure 9 Schematic working of the TrimCorrection	23
Figure 10 Schematic working of the NumberedListCorrection	25
Figure 11 Pseudocode for the NumberedListCorrection.....	26
Figure 12 Schematic working of the BlockListCorrection.....	28
Figure 13 Schematic overview of the BigTextCorrection.....	30
Figure 14 Schematic overview of journal identification and normalization processes in CiteRep's architecture.....	32
Figure 15 Example journal database entry	36
Figure 16 Example of splitting a citation into searchable parts	37
Figure 17 Example of the SimpleJournal method	40
Figure 18 Action chain for the NormalizeJournal procedure	41
Figure 19 Distribution of documents in UT repositories per year.....	45
Figure 20 Overall top 20 journals at the University of Twente.....	45
Figure 21 Distribution of documents per faculty	46
Figure 22 Top 10 journals for the CTW faculty between 2000-2015	46
Figure 23 Top 10 journals for the TNW faculty between 2000-2015	47
Figure 24 Top 10 journals for the EEMCS faculty between 2000-2015....	47
Figure 25 Top 10 journals for the BMS faculty between 2000-2015.....	47
Figure 26 Schematic overview of the worker-dashboard communication protocol	51
Figure 27 Example worker-dashboard network requests	52

LIST OF TABLES

Table 1 List of delimiters depicting the start of a bibliography section	20
Table 2 List of delimiters depicting the end of a bibliography section.....	20
Table 3 Baseline performance of CiteRep for journal identification in the ElsevierSet	22
Table 4 List of words which never occur in a citation.....	23
Table 5 Impact of TrimCorrection on CiteDataSet and ElsevierSet.....	24
Table 6 Impact of NumberedListCorrection on CiteDataSet and ElsevierSet	27
Table 7 Measurements for ElsevierSet when the NumberedList-Correction functionality is removed entirely	27
Table 8 Impact of BlockListCorrection on CiteDataSet and ElsevierSet..	29
Table 9 Impact of BigTextCorrection on CiteDataSet and ElsevierSet.....	30
Table 10 Measurements when all corrections are enabled	31
Table 11 Journal invalidators.....	35
Table 12 Common words in a journal notation	39
Table 13 Effect of disabling SimpleJournal and NormalizeJournal	42
Table 14 CiteRep accuracy using the Elsevier test set.....	43
Table 15 CiteRep accuracy using our manually created test set.....	43
Table 16 Comparison of CiteRep journal identification performance to ParsCit, FreeCite and Refparse.....	44

Chapter 1

INTRODUCTION

The exploration and analysis of scientific literature collections is an important task for effective knowledge management.

[1] CiteRivers: Visual Analytics of Citation Patterns

1.1 Context

Access to scientific journals is essential for universities and often comes with a considerable subscription fee [2]. University Libraries attempt to minimize costs by looking for vendors which provide full solutions for their needs [3]. Also, in light of recent open access initiatives, there is a need to determine which journals are referred to most often as a measure of importance to a specific research field [4].

The library of the University of Twente (UT) would like to obtain insight in how many times journals are being referred to in publications from students and staff. Knowledge about the actual usage of journals is deemed important for decision making strategies [5]. CiteRep is concerned with gathering information for decision making by providing insight in which journals a university should provide in order to accommodate the needs of students and staff.

Currently there are little options available to obtain information about journal usage in a university context. The information that is available is often not fine-grained or reliable [6]. Journal citation counts provided by publishers are mostly on annual basis and only at institutional level. CiteRep provides detailed journal citation reports by automatically counting journal references from the bibliography sections inside documents obtained from university repositories. Journal citation statistics can be used to choose between various journal subscription packages and provide detailed insight in which journals are popular for specific faculties and studies.

1.2 Problem Statement

Students and staff refer to publications from various journals the University of Twente is subscribed to. Publications are downloaded from publisher sources and may be used for teaching, reading interests, or as a reference in new research. CiteRep provides insight in the usage of journals for scientific publications written at the UT. Detailed journal citation reports are generated by automatically processing documents from library repositories.

CiteRep relies on extracting the reference list inside a PDF document for obtaining a count of the number of journal references in a scientific publication. The reference list is a structured list of citations often near the end of a document (Figure 1). Documents are obtained from online library repositories which are automatically queried and processed. Repositories contain publications from both students and staff accompanied with some basic metadata. CiteRep provides fine grained journal citation reports by using metadata to relate a document to a specific faculty or study.

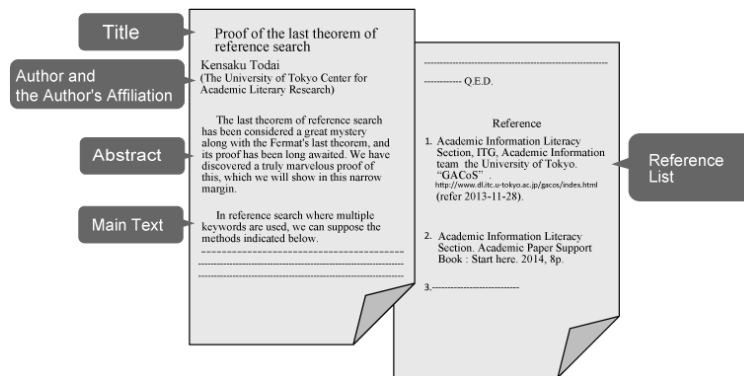


Figure 1 Structure of an academic paper

Sources providing documents for CiteRep at the University of Twente are not designed for citation analysis. Citation data is not available in a structured way and differs per document based on the reference style that is used. Some documents are incomplete or are actually unreadable image files. CiteRep is limited to the documents which can be processed as machine readable text. The total repository consists of 60,700 entries of which 42,122 could be processed for citation analysis.

Different citation styles also have different ways of citing a journal. Some styles refer to the full journal title whereas other styles use an abbreviated format. CiteRep uses normalization techniques to map different journal notations to a uniform format. Journals are uniformly identified independent from the citation style used.

Permission was granted by the ICT department to query the online APIs to obtain the PDF documents needed for this research. The university APIs make use of the Open Archives Protocol standard [7]. We argue that if other universities follow the methodology of this research they could also benefit from this procedure, although exact API implementation might differ slightly.

1.3 Research Questions

The goal of this research is to enable automatic generation of journal citation reports from university document repositories. CiteRep incorporates a three stage process to generate journal citation reports from a set of PDF documents. The first stage is to extract the citation section from each of the documents in the document set. In the second stage, for each citation in the bibliography, the journal reference is identified. Finally, differences in journal notation are normalized into one uniform notation.

The extraction, identification and normalization stages include unique challenges to be tackled. To this end the following research questions are answered in this thesis:

- RQ1 How accurately can a list of citations for a publication be extracted from the reference section when given a PDF document as input?
- RQ2 How accurately can the journal be identified inside a citation?
- RQ3 How accurately can journal titles be normalized to compensate for abbreviations and spelling differences?

CiteRep is our answer to the main research question from the University of Twente:

How can existing document repositories be used to provide insight in the usage of journals in publications by university students and staff?

1.4 Approach

Much research has already been put into automatic citation analysis assessing the impact and quality of scholarly publications [8]. CiteRep builds on established document processing technologies as identified during a study on related literature in the field of knowledge management. The literature study as summarized in the next chapter relates CiteRep to existing approaches.

Common elements identified from related literature are represented by CiteRep's extraction, identification and normalization phases. A software framework was developed to support these phases, keeping usability and extensibility in mind.

CiteRep's journal citation report accuracy greatly depends on the accuracy of each of the individual extraction, identification and normalization phases. For each phase accuracy was optimized using a validation set as measure. The overall accuracy of CiteRep was assessed by creating an open-source test set for journal citation performance evaluation.

The structure of this thesis reflects the approach as summarized here. The next page provides the reader with an overview of the contents of the remaining chapters.

1.5 Structure

The remainder of this document is structured as follows. Chapter 2 provides background information on the concepts on which CiteRep builds from related work in the field of knowledge management. Chapter 3 provides a general overview of the research approach, outlining CiteRep's system requirements, process architecture and evaluation criteria. Chapter 4 describes the process of extracting the reference section from a PDF document to find citations; answering the first research question. Chapter 5 discusses the methodology used to identify the journal in a citation reference text; answering the second research question. Chapter 6 evaluates different normalization measures taken for disambiguation handling journal abbreviations and spellings; answering the third research question. Chapter 7 provides the reader with an evaluation of CiteRep's overall accuracy and provides citation counts based on the repositories of the University of Twente. Chapter 8 presents the conclusions of this research. Future work is discussed in Chapter 9.

Chapter 2

BACKGROUND

Today many online applications such as Google Scholar³, Scopus⁴ and Web of Science⁵ allow to search through scientific publications, lookup authors and list which journals contain which publications [9]. Many of these features are made possible because of intelligent algorithms automatically indexing and processing references from published papers. Hence a lot of work has already been put in extracting metadata from bibliographies.

CiteRep cannot rely on online systems for various reasons. Most online algorithms are proprietary and inner workings are not disclosed. Also the nature of this research prohibits papers to be submitted to online resources because of access restrictions. Some papers are only published in the local university network and contain sensitive information which might not be published outside the university network. There are some citations parsing libraries that are open-source and can be run locally. We provide a brief overview of existing techniques in this chapter and conclude with listing the most popular open-source citation processing implementations.

2.1 Related Work

One of the first attempts to build a citation indexing system was CiteSeer in the late 90's. CiteSeer is an autonomous citation indexing system for academic literature following a clear procedure of document acquisition, document parsing and citation identification [10]. CiteSeer utilizes extraction based on heuristics. Regular features of a citation are automatically identified and used to predict whether certain pieces of information, such as a journal, title, or author inside a citation exists. CiteRep's extraction and identification phases closely resemble the parsing and identification procedures from CiteSeer. CiteRep's document acquisition procedure differs from CiteSeer. CiteSeer uses various document sources which are crawled for data retrieval. CiteRep uses university library web repositories for document acquisition.

Most tools processing bibliographies rely on the analysis of citation patterns. The CiteRivers [1] software tool for instance is a visual analytics program for finding citation patterns in scientific publications. CiteRivers uses the reference section and additional metadata from a paper to link it to other publications and run statistics on a given dataset. For actual reference extraction it submits the full title of a paper to the DBLP database⁶ using the response to identify elements within citations. The

³ <https://scholar.google.com>

⁴ <https://www.scopus.com>

⁵ <https://www.webofknowledge.com>

⁶ <http://dblp.uni-trier.de>

DBLP database contains open bibliographic information on major computer science journals and proceedings [11]. CiteRivers achieves an average field extraction accuracy of 70%, meaning it is 70% accurate in identifying which parts of a citation are for example the author, the journal, the volume something else. The CiteRivers approach does show that external sources can be used to obtain additional information given a publication title as input. To obtain citation counts the CiteRivers tool uses AMiner⁷, which is a database that contains all DBLP entries enriched with additional information extracted from academic social networks and websites [12]. CiteRep, analogous to CiteRivers, makes use of a database of known journals to aid with journal identification. The database of known journals used by CiteRep is specially crafted, open-source⁸, and can be used without requiring an internet connection.

Other approaches to citation extraction are based on knowledge representation frameworks capable of matching complicated template structures to known citation structures. The knowledge-framework developed by Min-Yuh Day et al [13] achieves an average field accuracy of 97.8% for citation extraction. Their Reference Metadata Extraction method builds upon the INFOMAP knowledge representation framework [14]. CiteRep also makes extensive use of known structures to find the citation section in a document and to identify the journal for each reference found.

Fuchun Peng and Andrew McCallum [15] attempt to improve the accuracy of research paper search engines such as CiteSeer by using conditional random fields for citation labeling. Conditional random fields are models which can encode knowledge of relationships between observations; enabling to label data the model has not seen before in a consistent way [16]. It is suggested by Min-Yuh Day et al. [17] that when compared to other techniques, the conditional random fields prove to have a better overall accuracy and are better suited for labeling elements inside references [13]. Citation processing using conditional random fields requires a learning phase in which the models are trained on known citation structures. CiteRep however has to work on documents from several languages and work on citation styles that are unknown to the system. The great diversity in languages and citation styles in the university repositories render it impractical for CiteRep to train conditional random fields for journal identification.

Labeling elements in citations can be solved by machine learning algorithms borrowed from other fields of research as Chen et al. [18] illustrate. They recognize that amongst all current citation parsing techniques those based on Conditional Random Fields currently achieve the best performance. However, these methods require extensive training for the vast majority of different scientific fields and their differences in citation notation. Their proposed software tool, BibPro, attempts to

⁷ <https://aminer.org>

⁸ <https://github.com/SVerkuil/CiteRep/tree/master/client/CiteRepData/journals.txt>

overcome this problem and borrows from protein sequencing technology by converting citation strings into protein sequences. For this they use a readily trained database, BLAST [19]. BLAST is used as a metadata extraction tool to find a candidate citation template by matching feature indices. It is able to parse a wide variety of citation styles and achieves an average field accuracy of 95% for the most difficult citation style. The BibPro source code is freely and publicly available on the internet⁹. BibPro however consistently has a low accuracy for extraction of the journal field. The authors argue that variability in punctuation is responsible for this behavior [18]. CiteRep needs to excel at identifying the journal field specifically and cannot rely on the BibPro sequence alignment approach.

RefParse is an generic approach to bibliographic reference parsing [20]. It is independent of any specific reference style. Its mechanism looks at regularities within multiple references to deduce its style and can recognize entities such as author, journal, title and many more. RefParse works on an entire list of references together, enabling it to infer the style at runtime. It works based on the assumption that all references within a bibliography are formatted using the same reference style. This assumption holds true after investigating over 1,000 real world documents [20]. In addition to known patterns, RefParse also makes extensive use of lexicons to look up author names and other attributes based on known values from the past. Combining previous knowledge and inferring the reference style based on common features, RefParse achieves 94% field accuracy on average. When looking specifically at journal identification RefParse outperforms the most popular open-source tool ParsCit [21]. The source code of RefParse is freely available¹⁰ and written in Java. RefParse is incorporated in the CiteRep architecture to provide guidance as to where the journal can be found in a list of citations.

⁹ <https://github.com/ice91/BibPro>

¹⁰ <https://github.com/VBRANT/refparse>

2.2 Adopted Techniques

CiteRep has adopted several strategies for citation processing from related work in the field of knowledge management. CiteRep reflects existing technologies on the following accounts:

- CiteRep’s extraction and identification phases closely follow the proven document parsing and citation identification procedures from CiteSeer [10].
- A database of known journals¹¹ is used to aid with journal identification. *Different from the approach taken by CiteRivers [1], the database used by CiteRep is available locally; enabling the identification process to run without relying on online resources.*
- CiteRep uses a knowledge-based [13] approach to citation extraction. Known citation characteristics are used by CiteRep to eliminate parts of a citation which do not refer to a journal, increasing the accuracy of journal identification.
- CiteRep is able to process bibliographies independent from specific reference styles. RefParse [20] is included within CiteRep to aid the identification process with finding the journal information inside a citation.

2.3 Open-Source Citation Tools

We conclude this section by listing the most popular open-source citation parsing libraries. Some of these parsers also have an online public API. Analogous to CiteRep, the software tools listed below adopt similar techniques from previous work in the field of knowledge management. CiteRep is only concerned with accurately extracting the journal from a citation, ignoring all other fields in a citation. CiteRep incorporates the unique ability to correct for journal spelling differences and abbreviations. This feature of CiteRep is not supported by any other parser.

List of related Open-Source Citation Applications

- | | |
|--------------------------|--|
| - RefParse ¹² | - ParsCit ¹⁵ |
| - FreeCite ¹³ | - Biblio Citation Parser ¹⁶ |
| - ParaCite ¹⁴ | |

The RefParse library is one of the more recent libraries, created in 2014 at the Karlsruhe Institute of Technology. RefParse is incorporated into CiteRep to help improve journal extraction accuracy.

¹¹ <https://github.com/SVerkuil/CiteRep/tree/master/client/CiteRepData/journals.txt>

¹² <https://github.com/VBRANT/refparse>

¹³ <http://freecite.library.brown.edu>

¹⁴ <http://paracite.eprints.org>

¹⁵ <http://aye.comp.nus.edu.sg/parsCit>

¹⁶ <http://search.cpan.org/~mjewell/Biblio-Citation-Parser-1.10>

Before actual citation processing can be performed on any PDF document, the bibliography section itself needs to be identified within the document text. One open-source library for this specific purpose is PDFExtract¹⁷. PDFExtract looks at visual positioning of text inside a document to identify where the reference section starts. We discovered that PDFExtract has difficulties with PDF documents that have additional markup in the header or footer. The software is also not mature; the latest version having more than 20 open issues which are still unresolved at the time of writing. PDFExtract cannot reliably be used within CiteRep as a method for extracting the reference section from a PDF document.

The open-source citation parsing library ParsCit also supports extracting the reference section from a PDF document. However, when ParsCit was given a random sample of 20 documents it was able to find the reference section in 75% of the cases, which is deemed insufficient for CiteRep.

CiteRep incorporates its own approach for finding the bibliography instead of using existing libraries. The CiteRep extraction procedure can correctly identify the citation section with an accuracy of 84% upon evaluation of the complete document set. CiteRep is the most accurate implementation for extracting the bibliography section from a PDF document and can handle many different document formats.

The next chapter outlines the research approach taken by CiteRep for extracting the citation section from PDF documents, identifying journals inside citations and normalize journals for spelling differences.

¹⁷ <https://github.com/CrossRef/pdfextract>

Chapter 3

RESEARCH APPROACH

CiteRep is our answer to the request for automatic journal citation reports set out by the business faculty at the University of Twente. CiteRep provides insight in how often UT students and staff refer to journals in their publications. Journal citation reports generated by CiteRep can be used for decision making processes concerning journal subscriptions, providing insight in how journals are applied by students and staff.

This chapter provides the user requirements and system requirements for CiteRep and outlines the process architecture following from these requirements. We conclude with an overview of the evaluation methods used to assess the accuracy of individual components and CiteRep as a whole.

3.1 Specification of Requirements

A brief overview of the most important user and system requirements is provided. Requirements originate from discussion with stakeholders and the original thesis project description provided by the University of Twente.

The user requirements for CiteRep are:

- User-friendly and easy to understand interface.
- Concurrent use of CiteRep by multiple users.
- Simple and minimal installation procedure.
- Connections with repositories are easily created and managed.
- Intermediate results of extraction, identification and normalization phases are made available to the end-user for manual inspection.
- Detailed journal citation reports are provided and can be filtered based on year, faculty and study.

The system requirements for CiteRep are:

- Work in connection with university repositories through API calls.
- Document processing is done within the local UT network.
- PDF documents are automatically processed into plain text.
- The citation section inside given text is automatically identified.
- Different document languages and citation styles are supported.
- Journals in citations are automatically identified and normalized.
- Documents are processed unattended and in batch.
- CiteRep is easily extensible with new functionality.
- Platform independent.

The university repositories are dynamic and change over time based on the current state of research at the UT. It is required that CiteRep's reports are easily accessible and easy to customize by authorized university employees. The CiteRep software framework allows multiple users to access journal citation reports directly. CiteRep does not require difficult system setup and runs directly on any system architecture. The software is well documented and can easily be extended in the future.

Access to some publications at the UT is restricted to the universities internal network. CiteRep's document processing functionality runs on a separate worker on a standalone machine inside the university network. This worker is written in Java and requires no setup or installation. CiteRep provides a web interface to view the journal citation reports. The web interface does not necessarily have to run on the same physical machine as where the worker is processing PDF documents.

The CiteRep software prototype processes the document repositories that are available at the university network. Documents are provided by repositories in XML format, following the Open Archives OAI specification [22]. OAI is a well specified and frequently used format. CiteRep architecture uses the XPath [23] language to allow for any XML source to be used as document input to CiteRep in the future.

3.2 CiteRep Process Architecture

CiteRep automatically acquires journal citation reports from university repositories using a multistage process of extraction, identification and normalization. The process details of document acquisition, citation extraction, journal identification and journal normalization are provided in the following subsections.

3.2.1 Document Acquisition

There are two main repositories of publications at the University of Twente. These sources are accessible via an open API and contain basic metadata such as year of publication, faculty, authors and abstract. Document entries are accompanied by a pointer to the location of the corresponding PDF document for CiteRep to download.

The first repository, which can be found at *doc.utwente.nl*, contains publications from PhD students, professors and other university staff. The second repository, *essay.utwente.nl*, contains material created by students, including bachelor projects and master theses. After investigation of the total document set we discovered some documents that were written by students and were later revised and in collaboration with a professor published again. In such cases these documents exist, possibly being slightly modified, in both repositories. CiteRep processes these documents twice in total, once from each repository, and hence the journal citation reports are slightly biased towards journals cited in these publications.

We argue that the relevance of work that is revised twice is high, and therefore the importance of the referenced journals is also high. Data deduplication between the *doc* and *essay* repositories is omitted within CiteRep. If the document resides in both sources, possibly with minor improvements, it will be indexed and scheduled for processing twice, counting the referenced journals twice.

Documents are sometimes retracted after publication and CiteRep needs to reflect upon changes in the document set. The repositories *doc* and *essay* make use the Open Archives Protocol [24] and keep persistent information about deletions. Querying the API is similar to processing a changelog. For each entry representing a publication its creation, alteration and possible deletion date is known. Querying the API from the first entry to the last yields create, modify and delete actions in a chronological order; always resulting in a consistent document set upon completion. This also means that the results of this research could be easily replicated when provided with access to the university network. If the API is queried till upon the time our measurements, May 23th 2016, the resulting dataset is exactly the same as used during this master thesis research and hence outcomes can be easily verified.

3.2.2 Citation Extraction

A PDF document, annotated with metadata obtained from one of the repositories, is converted to machine readable text by CiteRep. Reading and processing PDF documents are performed using the open-source library Apache PDFBox¹⁸. PDFBox has a build-in text extraction which is implemented within CiteRep. CiteRep then stores the full text extraction enabling end-users to visually inspect the extraction phase of the process and aid with debugging. The next difficult part is to automatically identify the piece of text which contains the bibliography section in the document full text.

CiteRep attempts to identify the reference section by looking at common attributes such as headings and lists. There are many different lay-outs in which PDF documents are written, ranging from simple one page documents to multi column page layouts. References are automatically extracted by CiteRep from all types of PDF documents with a small fault margin. If for example a certain document layout would consistently fail, and a department uses that layout all the time, CiteRep would be biased. If the citation section is not correctly extracted, the journal identification step will also fail. For each paper each intermediate output is logged and made available for inspection within the CiteRep web interface. The implementation details of CiteRep's citation extraction procedure are provided in Chapter 4.

¹⁸ <https://pdfbox.apache.org>

3.2.3 Journal Identification

Journal identification is the process used by CiteRep to identify the piece of text inside a citation that contains the journal information. The journal identification process assumes the citation extraction procedure was successful and the produced citation list is used as input to the identification process. There are multiple ways of referring to the same journal, either by full name, abbreviation, short code or some other non-standard format [25]. Challenges with processing journal information were already present as early as 1994 when the first journal citation maps were computer generated [26].

For example, the journal of the Association for Information Science and Technology is sometimes referred to using the abbreviation *J Am Soc Inf Sci Technol* whilst other papers refer to it as simply *JASIST*. This example also does not show up in the list of known journals maintained by Web of Science [27]. CiteRep has to handle writing differences to provide accurate journal citation reports.

Multiple lists exist on the internet which contain journals and their official abbreviations. Publisher price lists are also well maintained lists of known journals and their official abbreviations. Twenty such lists were found on the internet are used by CiteRep to improve the identification process, matching pieces of the reference text against known journals. A simple one-time procedure was created enabling to quickly merge lists from various sources together into one list without duplicates. The resulting list of journals and known abbreviations is made available open-source for others to use and include in their software¹⁹.

CiteRep makes use of the RefParse library and includes additional techniques to aid with finding the journal in a citation. RefParse library is a recent open-source²⁰ project and outperforms current citation parsers with respect to journal accuracy [20]. CiteRep uses additional algorithms to further improve journal identification accuracy. CiteRep is unable to process other parts in a citation and is tailored to extracting only journal information from citations. Chapter 5 provides with detailed insight the inner workings of CiteRep's journal identification procedure.

3.2.4 Journal Normalization

Journal normalization is the process used by CiteRep to rewrite different notations of the same journal into one uniform notation. This procedure ensures that unique journals are counted, preventing treating each spelling difference as a new journal in citation reports. CiteRep is the first and only open-source analytics tool providing a journal normalization procedure.

¹⁹ <https://github.com/SVerkuil/CiteRep/tree/master/client/CiteRepData/journals.txt>

²⁰ <https://github.com/VBRANT/refparse>

The list of known journals and their alternate notations was used to compile a list of common journal abbreviations. The compiled list contains over 300 known journal abbreviations such as *proc*, which stands for *proceedings*, and *j*, which stands for *journal*. CiteRep uses this list of common abbreviations to normalize journal titles into one generic representation. Applying this procedure to various journal notation styles always yields the same result. Chapter 6 further details the full journal normalization process used by CiteRep.

3.3 Evaluation method

Accuracy is assessed throughout this thesis using precision, recall and f-score; commonly used as metric in the field of information retrieval [28]. CiteRep searches the citation section in a document for journal references and the resulting set of journals is then compared to a list of known journals from a test set. Precision denotes which fraction of the retrieved journals is relevant to the document that was processed. Recall indicates the fraction of the relevant journals that was retrieved. The f-score is the harmonic mean of the precision and recall scores.

$$precision = \frac{|\{known\ journals\} \cap \{found\ journals\}|}{|\{found\ journals\}|}$$

$$recall = \frac{|\{known\ journals\} \cap \{found\ journals\}|}{|\{known\ journals\}|}$$

$$fscore = \frac{2 \cdot precision \cdot recall}{(precision + recall)}$$

For each paper from the test set its precision and recall is calculated. The precision and recall scores of all documents combined are divided by the number of documents in the test set to obtain the weighted precision and weighted recall. These weighted values are used in this thesis to assess the accuracy of CiteRep on a test set.

CiteRep’s main purpose is to generate journal citation reports. CiteRep identifies all journals in a citation section even if the citation section is malformed or multiple citations are concatenated together during text processing. CiteRep is not always able to keep track of the relation between a journal and the citation it was found in. CiteRep can be seen as a search engine, searching for all journals inside a set of publications. For the generation of journal citation reports only the document with associated journal references and metadata is known.

3.3.1 CiteRep Document Sets

During this research we use five datasets for evaluation. The first dataset is a validation set called the *CiteDataSet*. The dataset consists of 250 randomly picked papers from the set of all published papers available at the university repositories. The *CiteDataSet* is used for benchmarking individual components in the identification phase of CiteRep, helping to determine various threshold values in our architecture and supporting claims that are made about the population.

The second dataset is a test set which is simply called *TestSet* and is the main test set used to make claims about the overall accuracy of CiteRep. The 40 papers were left untouched until the CiteRep development was complete, ensuring that CiteRep is not biased towards documents from this dataset. The test set was hand crafted and is open-source²¹. It can be used by other researchers from inside the university network for benchmarking and validation of the CiteRep research.

3.3.2 Elsevier Document Sets

Elsevier graciously provided us with two document sets which contain PDF documents and citation sections annotated with journals. We have observed that Elsevier uses methods of their own to normalize journal notations, presumably to relate journals to other databases in their infrastructure. The precision and recall calculations of CiteRep are influenced by these normalizations and hence compared to other datasets the Elsevier benchmarks have lower scores. However, the relative change in performance is still measurable. Because there are not many datasets available for benchmarking we are happy to use the ones provided by Elsevier.

The third dataset, a validation set called the *ElsevierSet*, consists of 50 papers which exists in both the Elsevier Scopus database and in our CiteRep database. We use this set to optimize the journal identification phase threshold values within CiteRep. Because we iterate multiple times over this dataset and actively use it to improve our methodology, this set cannot be used for final performance evaluation.

We have a fourth set of papers consisting of 80 entries that are both in the Elsevier database and in our CiteRep database. We call this test set the *ElsevierStandard* and is used to calculate the precision, recall and f-score of the overall system. The dataset is only used for evaluating the CiteRep architecture.

3.3.3 Cora Dataset

The fifth and final test set is the standardized *CoraDataSet* created by Andrew McCallum [29]. This dataset consists of 500 annotated citations, not papers. These citations are used to benchmark the performance of CiteRep on standalone citation parsing. Because other citation parsers such as ParsCit and RefParse also have used the Cora dataset for benchmarking the test set is used to compare CiteRep to existing citation parsers. The test set was left untouched until CiteRep was completed to prevent our system from being biased towards the citations in the Cora dataset.

We conclude this chapter by providing an outline of the prototype software architecture of CiteRep. The software architecture of CiteRep satisfies the user and system requirements and provides procedures for document extraction, journal identification and journal normalization.

²¹ <https://github.com/SVerkuil/CiteRep/tree/master/client/TestSet/>

3.4 Software Architecture

CiteRep has a modular software framework enabling the automatic generation of journal citation reports from PDF documents. The framework connects to online university repositories to download documents from students and staff. These documents are automatically processed and journal statistics provided with a user-friendly interface.

We have chosen to keep the software architecture of CiteRep simple. The primary focus is on obtaining insight in how journals are used at the University of Twente. CiteRep primarily support this cause and has little other functionality. CiteRep was developed with easy end-user accessibility in mind. CiteRep is modular and can easily be extended with new functionality in the future.

The CiteRep framework is divided into two main components. The first is called the *worker*. A worker performs tasks such as downloading and parsing PDF documents, extracting citations and identifying journals in citations. The second component, the *dashboard*, is an online web interface and database which stores the outcomes of the workers and displays them in a graphical and understandable way. The dashboard is used to create and manage remote sources. The dashboard creates *tasks* for document extraction and journal identification which are performed by a remote worker. The worker and dashboard communicate using frequent heartbeat messages carrying the tasks to be completed and triggers to store data in the database once the task is completed. A schematic overview of this setup is shown in Figure 2.

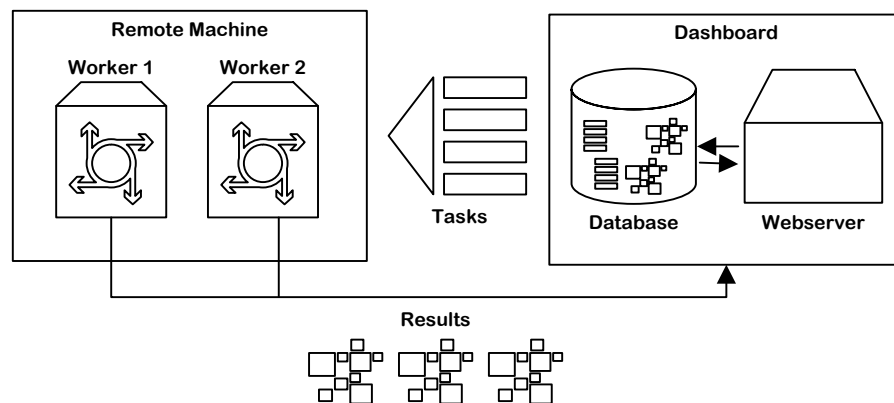


Figure 2 Illustration of worker and dashboard interaction

The dashboard web interface front-end is designed to be simple in use and is based on an open-source well-documented responsive interface developed by Twitter called Bootstrap²². The web interface dynamically scales to desktop computers and mobile devices. Figure 3 shows the CiteRep web interface for browsing through publications indexed from the university repositories on a desktop computer. Figure 4 shows the same interface as rendered on mobile devices.

²² <https://getbootstrap.com>

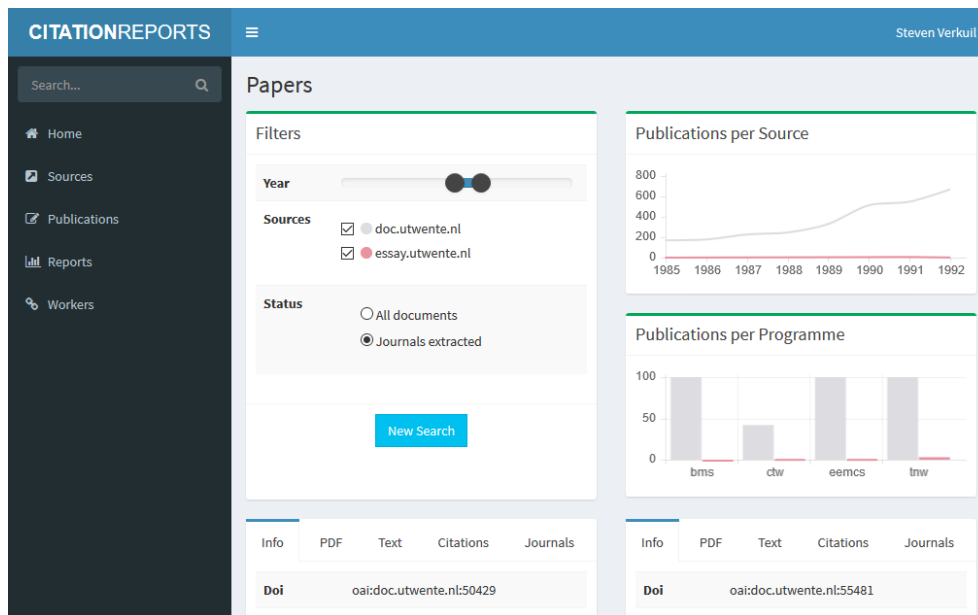


Figure 3 Desktop interface for reviewing publications in CiteRep

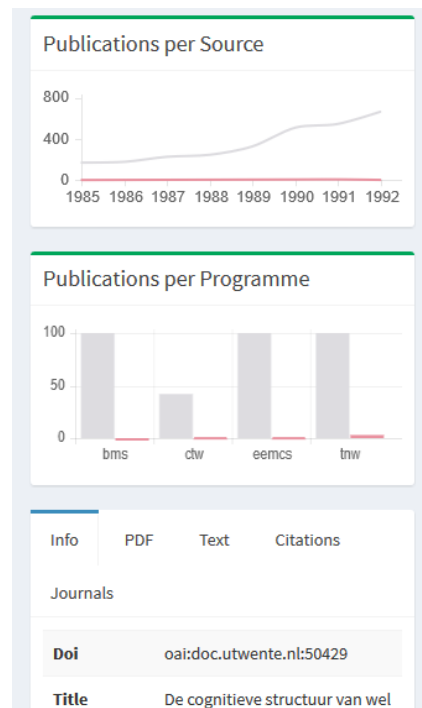


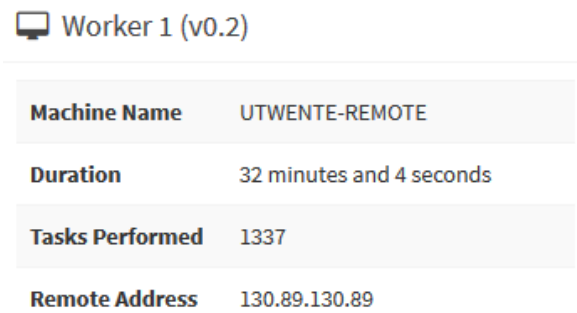
Figure 4 Mobile interface for reviewing publications in CiteRep

The dashboard backend is based on a popular open-source web framework written in PHP called Laravel²³. Both the frontend and backend architecture frameworks are well documented. Ample learning examples are available online in case the software needs to be extended in the future by people not familiar with these frameworks.

²³ <https://laravel.com>

The worker application is programmed in Java. The main benefit of using Java is its capability to run cross platform, maximizing application portability [30]. The worker is packaged as a jar executable file which can be executed using a simple command on the command line. The worker can be left alone on a remote server inside the university network automatically performing tasks once they become available. Using Java allows support for a lot of standardized and open-source libraries. A good example of a library that is used by CiteRep is Apache PDFBox. PDFBox is used to extract text from PDF documents.

CiteRep’s software architecture allows for multiple workers to be run and connected to the dashboard at the same time. CiteRep aims for even work distribution amongst separate machines when performing CPU intensive tasks such as extracting plain text from PDF documents and identifying a journal in a citation text. Workload distribution is especially helpful when new repositories have to be processed. Basic information about connected workers is shown to the end-user in the dashboard as seen in Figure 5.




 Worker 1 (v0.2)	
Machine Name	UTWENTE-REMOTE
Duration	32 minutes and 4 seconds
Tasks Performed	1337
Remote Address	130.89.130.89

Figure 5 Worker performing tasks on a remote machine

Communication between the worker and the dashboard is performed using a simple JSON [31] messaging protocol. CiteRep has easy to understand protocol handlers for the worker client and dashboard server. Details about the CiteRep messaging protocol are provided in Appendix A.

The CiteRep framework is user-friendly and supports the citation extraction, journal identification and journal normalization phases using tasks performed by workers. The next three chapters provide insight in the algorithms used by CiteRep for extraction, identification and normalization supported by the software framework.

Chapter 4

CITATION EXTRACTION

Citation extraction within CiteRep is considered with finding and extracting the citation section in the plain text of a PDF document. The bibliography section is identified using known citation characteristics. CiteRep uses four text correction techniques to further improve the accuracy and readability of extracted citation sections. This chapter provides a detailed outline of the procedures used by CiteRep. For each correction its working is explained and accuracy measured using two validation datasets. The reader is invited to read on and discover how CiteRep extracts and processes document citations.

4.1 Extracting the Reference Section

A scientific PDF document can have numerous different layouts and the reference section is not always at the end of the document. CiteRep cannot use a single universal approach to document processing. Some documents have a two column layout with the reference section at the end of the document. Other documents have a single page layout and a reference section at the end of each chapter. Sometimes the bibliography section is not clearly marked with a caption, but for instance with a visual pointer such as a horizontal line. The bibliography section itself could be a numbered list, a list in alphabetical order, or some other layout. Different professions each have their own favorite way of displaying references and standardized formats such as ACM, ABNT, APA, IEEE, Chicago and many more exist [25]. CiteRep adopts a knowledge-based approach in order to facilitate citation extraction from varying citation styles.

Knowledge about the university document set enables CiteRep to extract bibliographies with high accuracy. Manual inspection of a sample of publications at the University of Twente yields the following observations.

- Documents are written in English, German or in Dutch.
- The bibliography section is often found at the last 1/3th part of the document.
- Most bibliographies are numbered. There are many variations in number notations. For example, [1], (1) or 1. are commonly used list styles.
- If there is more text after the bibliography section inside a document, it is often the case that there is a clear title preceding the section. For instance, the texts “appendix”, “summary” and “motivation” often appear as new sections after the bibliography section.
- All citations in a document follow the same citation style. We have found no case in which a single document contains multiple reference styles.

The open-source Apache PDFBox library is used to convert a PDF document to plain text. The TextStripper class that comes with this library automatically detects paragraphs, lines and delimiters in a document.

The library supports setting a threshold value of whitespaces that must occur before a new paragraph is detected. We have discovered that, because of the wide diversity of documents that need to be processed, there is no uniform whitespace setting that correctly identifies paragraphs for all types of text. As a result, CiteRep cannot rely on paragraph detection as performed by PDFBox. The reference section cannot be visually identified using paragraph spacing and has to be found in the text itself.

CiteRep uses a list of common keywords to find the start of a bibliography section in the document text. Such a keyword needs to be preceded by the beginning of a newline character and followed by another newline character. By doing so we are explicitly looking for titles (single word sentences), and not words that are part of a regular sentence. The delimiters that are identified as a start of the reference section, both in Dutch, German and English language are shown in Table 1.

References	Bronvermelding	Bronverwijzing	Literatur
Bibliography	Bronnen	Reference list	Literatuur
Bibliografie	Referenties	Bronverwijzingen	Literature
Literature Cited	Literaturhinweise	Resource guide	Literatuurlijst
References and notes			

Table 1 List of delimiters depicting the start of a bibliography section

Similarly, a list of delimiters was identified depicting the end of a reference section. CiteRep assumes that the text processor at this point already found the start of the bibliography section and started to capture the lines that come next. A line beginning with one of the words from Table 2 is presumably no reference and hence we have left the reference section and CiteRep stops capturing text.

Appendix	Bijlage	Bijlagen	Index
Chapter	Authors	Afbeeldingen	Acknowledgement
Appendices	Summary	Motivation	Notes
Table	Figure	Fig.	Noten
Samenvatting	Summary	Section	

Table 2 List of delimiters depicting the end of a bibliography section

It is important to note that CiteRep’s procedure of finding the bibliography section based on fixed delimiters is prone to errors. The citation section is badly extracted in about 33% of the cases upon evaluation using 250 papers from the CiteDataSet. Whenever this procedure fails the result is either an empty set of citations or a large piece of irrelevant text is added to the presumed reference section. In both cases either a starting delimiter or ending delimiter was missed during citation extraction.

Based on our measurement it was almost never the case that the reference section of a well formatted document was processed partially, meaning that the text scanner started at the right point in the text, but stopped before the end of the reference section. The only case in which this occurred was when the PDF document has a two column layout and Apache PDFBox first outputted the second column contents instead of starting text processing with the first column. This resulted in text being mixed up and numbered list being out of order. CiteRep uses a *NumberedListCorrection* to compensate for this behavior for numbered lists specifically. If this behavior occurs within non-numbered lists CiteRep might return an incomplete list of references.

Methods were defined to perform another extraction method if the citation section came up empty. Text corrections attempt to fix the resulting output if text was added to the citation section that is not actually a citation. Each such a correction within CiteRep has three stages. In the first stage the correction validates if it is applicable to the given input. If the correction is applicable, it will be executed. Applicability of a correction does not necessarily mean it alters the input provided. It simply means that the correction in a second stage is allowed to take a look at the citation section to see if it can improve its contents. The third stage calculates if further corrections are allowed or if the returned result is deemed final. Corrections are chained and executed in order. The corrections defined in CiteRep are shown in Figure 6.

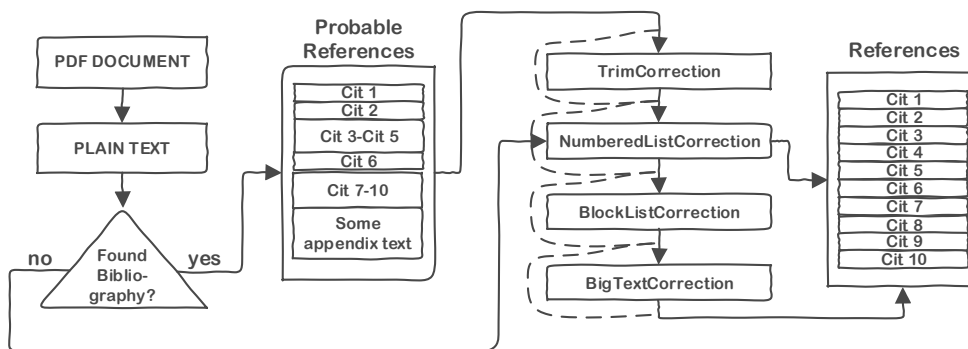


Figure 6 Corrections used by CiteRep to improve reference extraction

CiteRep first attempts to identify the reference section based on known delimiters. The output is fed into a series of corrections. Each correction chains to the next correction, or in the case of the *NumberedListCorrection* branches directly to the end state preventing other corrections from being applied. If no reference delimiter was found, or if many such keywords were found, there is no single piece of text reliably

identified as being the bibliography section. CiteRep then calls a special procedure in which the raw text of the PDF document is directly fed into the NumberedListCorrection. The NumberedListCorrection is very powerful and finds all numbered lists directly from plain text. If the citation list was ordered alphabetically or using some other format CiteRep is unable to process the document. For the CiteDataSet, 26% of the 250 papers contain no identifiers that indicate the start of a reference section. When the documents without bibliography keyword identifiers were fed directly into the NumberedListCorrection, 61% was processed correctly. CiteRep’s NumberedListCorrection significantly increases the number of documents for which the citation section can be automatically processed.

The remainder of this section explains each of the corrections in more detail. For each correction a rationale is provided for choices that have been made based upon observations we did using the CiteDataSet. The consequences of our choices were evaluated by looking at the overall system performance impact using the ElsevierSet. Each time an individual correction was benchmarked, all other corrections in the correction chain were disabled. A baseline measurement without having any text correction enabled is provided in Table 3.

Precision	Recall	F-score
0.575	0.570	0.573

Table 3 Baseline performance of CiteRep for journal identification in the ElsevierSet

4.1.1 TrimCorrection

The TrimCorrection removes additional non relevant text from the beginning or ending of the citation section. Determining if arbitrary text has been added before or after the reference section is done by checking the length of the presumed citation section and the total length of the PDF text. We found that for the CiteDataSet, containing 250 randomly sampled papers from university repositories, a paper contains on average 82,619 characters (median 45,506 characters). When looking at the citation sections, we found that a citation section contains an average of 5,849 characters (median 4,612 characters). We provide a plot of the distribution of character counts of papers and citation sections in Figure 7 and Figure 8 respectively.

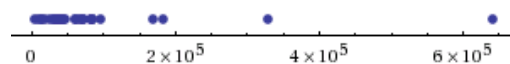


Figure 7 Distribution of character counts in repository papers

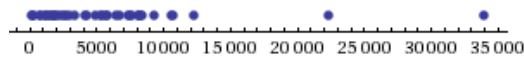


Figure 8 Distribution of citation character counts in repository papers

On average 7% of all text of a paper is contributed to the citation section. CiteRep expect that the TrimCorrection should be applied if the citation section makes up a significant larger portion than 7% of all characters. Applying the correction does not necessarily mean that there is actual text to trim, but at least the correction is given the chance to improve the citation text.

CiteRep also applies the TrimCorrection if there is a presumed citation which contains one of the word sequences from Table 4. These word sequences were found by manually sampling papers from the CiteDataSet and often occur in an appendix or non-citation text following right after a citation section. When CiteRep finds any of these word sequences in a citation, the entire citation section is scheduled to be processed by the TrimCorrection.

Was born in	Received the	The m.s.c.	m.s.c. degree
From the university	At the department	This appendix	The proof of

Table 4 List of words which never occur in a citation

The TrimCorrection can perform two kind of corrections. It removes additional text that precedes the citation section and it removes text that follows right after. It is not possible to feed a full text PDF file to the trim function as the trim function assumes that a search for bibliography keyword identifiers has already been performed when the document was first processed to machine readable text. The TrimCorrection works on the assumption that if a delimiter depicting the start of the bibliographic section was found by the text scanner, the text before that delimiter was already discarded. Figure 9 shows the working of TrimCorrection schematically.

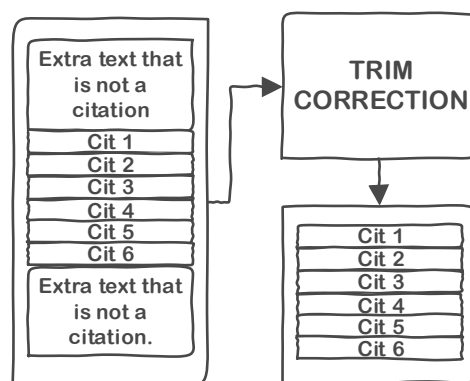


Figure 9 Schematic working of the TrimCorrection

To determine if there is additional text before the citation section the TrimCorrection uses the delimiter list of Table 1. The TrimCorrection splits the text in case the start of the reference section was concatenated with another sentence in the document, hence the text processor missed to identify the beginning of the bibliography section.

To check if there is additional text after the citation text is a bit harder. Additional text behind a citation section means that there is no reliable ending delimiter for the TrimCorrection to use. The TrimCorrection identifies additional text after a citation section by processing the list of presumed citations from first to last. For each identified citation the ratio of common citation-characters such as ; ,] [) (- / and digits is checked. We have found that inside a citation a minimum of 7% of the characters is a special character or a digit upon inspection of the CiteDataSet. If the TrimCorrection finds a citation that does not comply with the 7% special character threshold, all other presumed citations after that point are marked as additional text and trimmed.

The TrimCorrection has a global threshold value that determines if the correction applies to the input citation list. The TrimCorrection is applied if at least x% of all characters in a document were contributed to the citation section. For the CiteDataSet on average 7% of the characters belong to the citation section. We have varied the threshold values for the TrimCorrection to find the optimal threshold value.

For each threshold value we show for the CiteDataSet to how many papers the TrimCorrection *applies* (meaning the threshold value is met or the text contains a special word from the table above). We also show how many citations of the dataset were actually *altered* by the correction. The effect of the TrimCorrection on the CiteDataSet and the influence on the accuracy of journal extraction on the ElsevierSet is displayed in Table 5.

Thresh.	CiteDataSet (250 papers)		ElsevierSet (50 papers)		
	Applies	Alters	Prec.	Rec.	F-Score
0.0	76.0%	10.0%	0.577	0.570	0.573
0.05	60.0%	8.0%	0.577	0.570	0.573
0.07	46.8%	5.6%	0.577	0.570	0.573
0.1	30.0%	5.2%	0.577	0.570	0.573
0.2	8.4%	3.2%	0.577	0.570	0.573
0.5	3.6%	3.2%	0.577	0.570	0.573
0.7	3.6%	3.2%	0.577	0.570	0.573
1.0	2.8%	2.8%	0.577	0.570	0.573

Table 5 Impact of TrimCorrection on CiteDataSet and ElsevierSet

There are a quite a few interesting observations drawn from Table 5. The first observation being whatever the threshold value of the TrimCorrection is, it does not influence the accuracy of journal extraction on the ElsevierSet. We explain this as follows. The TrimCorrection is designed to remove additional text which is not part of the citation section. If a citation has additional text, it does not influence the fact that there is still a single journal in each citation. The journal is still found by CiteRep using the identification procedures explained in chapter 5. CiteRep's journal identification procedure is proven to be robust enough to compensate for additional non-citation text in bibliography sections.

CiteRep stores intermediate values of the overall process, keeping record of the original PDF document, the extracted text, the identified citations and the final list of journals. Having the citation list as clean as possible is good for human understanding and readability when investigating intermediate results. Based on the observations from Table 5 CiteRep has set the threshold value of the TrimCorrection to 0.0 in its architecture, making the TrimCorrection effectively part of our main algorithm. The TrimCorrection is applied every time, improving human readability for 10% of all papers on average. Besides having a positive impact on readability of the extracted citation section, the TrimCorrection does not influence the accuracy of journal citation extraction.

4.1.2 NumberedListCorrection

The NumberedListCorrection is the most powerful correction in the CiteRep architecture. It is capable of finding numbered lists in any given piece of text, supporting various numbering patterns such as [1], (1) and 1. It gracefully handles mistakes made by the PDF to text conversion process. It could for instance be the case that citations 4-8 precede citations 1-3 if document columns were processed out of order by the text processor. The NumberedListCorrection automatically corrects for mixed up column ordering for multi-column documents. The correction also corrects for missing opening or closing brackets for numbers in numbered lists. The flexibility of the NumberedListCorrection allows it to work directly on the full text of a PDF document. The workings of the NumberedListCorrection are shown schematically in Figure 10.

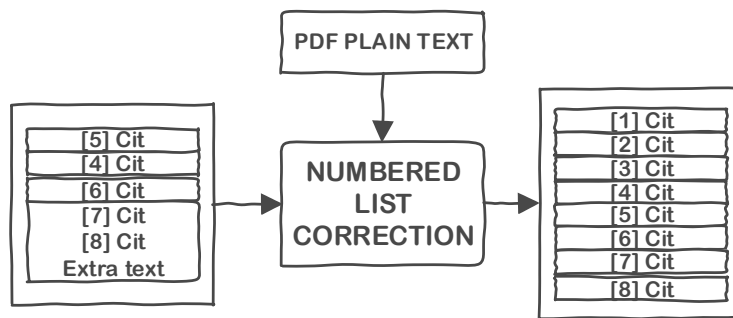


Figure 10 Schematic working of the NumberedListCorrection

The NumberedListCorrection is able to fix the order of the citation section, find citations that are accidentally concatenated together, and remove extra text that is not a numbered citation. It also finds missing citations due to column ordering issues. It uses the full document plain text as additional input in an attempt to find numbers missing from the numbered list. The procedure for finding all numbered lists in a document is rather complex. In basic the procedure runs through all lines in the text until it finds a line that starts with the number one. If this line is found, all consecutive lines are captured, until certain conditions are invalidated. The capture stops and starts again from that point in the text. An outline of the key concepts of the procedure in pseudocode are shown in Figure 11.

FindNumberedLists(text, start)

Set **count** to the value of **start**

Create an empty list to hold **citations**

Split the **text** into **lines**

For each **line** in the list of **lines**

Check if the **line** starts with a number equal to **start**, if so, store this line as the current **citation**

Else if the **line** does not start with a number, append it to **citation**

Else if the **line** has number, increase **count**, store **citation** in a list of **citations**, and store the current **line** as the new value of **citation**

If we have appended 9 times without finding a **line** that starts with a number, add the current **citation** to the list of **citations** and call the procedure **Validate(citations)**

Validate the **citations** that are not already validated by calling **Validate(citations)**

Since the **text** can be out of order, if **count** is higher than **start**, call **FindNumberedLists(text, count)** and see if it yields any more results and append these results to the **citations** list.

Return the unique aggregate of all validated **citations** lists and sub-calls. If this aggregate is an empty set, leave the original input unaltered.

Validate(citations)

If the list of **citations** contains a citation which has no letters in it, discard the entire list.

If the list of **citations** contains a citation which contains text from the papers chapter outline, discard the entire list.

If more than half of the **citations** contain not enough characters that are common in a citation, discard the entire list.

If more than one third of the **citations** contains characters that are commonly present in a formula, discard the entire list.

If we reach this point, leave the **citations** unaltered

Figure 11 Pseudocode for the NumberedListCorrection

Although the NumberedListCorrection succeeds in finding almost any numbered list in even very badly formatted text, it sometimes also returns a false positive. Some documents contain numbered lists with formulas, chapter indexes or other information. These lists sometimes pass all citation validation checks. We have decided to leave the NumberedListCorrection as it is instead of adding more validation methods in an attempt to further improve the precision of this procedure. Already a lot of time and effort has been put into this method and it has greatly improved the overall system accuracy of CiteRep. Time did not allow for further optimizations on this part of the system.

For CiteRep to apply the NumberedListCorrection, there is a threshold value which has to be met. If more than a certain percentage of the input citations start with an (encapsulated) number, the correction applies. The correction itself is then applied to the full text of the source PDF document since columns that are out of order could have resulted in missing citations. Table 6 shows how the NumberedListCorrection performs on papers from the CiteDataSet and the ElsevierSet.

Thresh.	<i>CiteDataSet (250 papers)</i>		<i>ElsevierSet (50 papers)</i>		
	Applies	Alters	Prec.	Rec.	F-Score
0.0	76.0%	39.2%	0.565	0.575	0.570
0.05	50.0%	34.4%	0.565	0.580	0.573
0.07	48.0%	34.4%	0.565	0.580	0.573
0.1	47.6%	34.4%	0.565	0.580	0.573
0.2	46.0%	33.6%	0.565	0.580	0.573
0.5	42.8%	32.4%	0.565	0.580	0.573
0.7	40.0%	30.8%	0.565	0.580	0.573
1.0	22.8%	18.8%	0.568	0.576	0.572

Table 6 Impact of NumberedListCorrection on CiteDataSet and ElsevierSet

Varying the threshold value for the NumberedListCorrection is shown to minimally influences the f-score of the overall system performance. If CiteRep cannot find the bibliography section inside a PDF document, the entire PDF documents text bypasses the correction chain and is fed directly through the NumberedList-Correction. In retrospect, the baseline measurements of Table 3 already benefitted indirectly from this corrections functionality and no significant difference in f-score is observed using the correction chain. The functionality of this correction is of vital importance to CiteRep to process numbered citations when the bibliography section could not be identified. Table 7 shows the performance decrease of our architecture if all functionality of the NumberedListCorrection is removed from CiteRep.

Precision	Recall	F-score
0.486	0.517	0.501

Table 7 Measurements for ElsevierSet when the NumberedList-Correction functionality is removed entirely

Looking at Table 6 we find the most optimal threshold values are between 0.05 and 0.7, meaning that of the citation list that is provided as input, at least 7% or 70% respectively needs to start with a number before this correction is applied. We have chosen set the threshold value to the lower half of this range, namely to 0.1. This threshold value enables the correction to alter 34.4% of the input, improving user readability of the citation section upon manual inspection without compromising precision or recall. The threshold values 0.0 and 1.0 decrease the accuracy of CiteRep as numbered lists are either processed when there are none, or not processed at all.

In conclusion CiteRep mainly benefits from the NumberedListCorrection as a mechanism to find citation lists in documents where no bibliography indicator was found. It allows to process documents which otherwise would be left out of journal citation reports, increasing the percentage of documents that CiteRep can process.

4.1.3 BlockListCorrection

The BlockListCorrection is the simplest of all corrections in the CiteRep system. It is applicable to a list of citations, if and only if more than a certain threshold value of these citations starts with a '['. Please note that if such a list is a numbered list, it was already processed by the NumberedListCorrection and CiteRep branched directly to the end state, bypassing this correction.

The BlockListCorrection works by looping through the list of presumed citations to fix errors in splitting up the citation list. These errors often occur because the PDF to plaintext conversion misinterprets when a new citation starts. When it makes this mistake it accidentally identifies a sentence that actually belongs to the citation the line above as being a new citation. We fix this by looping through all citations, and if the citation does not start with a '[' it is appended to the previous citation as is shown schematically in Figure 12.

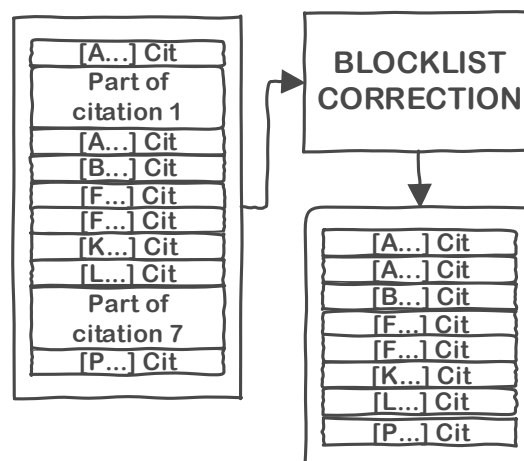


Figure 12 Schematic working of the BlockListCorrection

The BlockListCorrection does not under any circumstance discard any pieces of its input. It only merges parts of the list to correct for wrong “splits” that were made earlier in the citation extraction process.

The threshold value for this correction determines the amount of sentences in the citation list that must start with an '['-character before the BlockListCorrection is applied. As can be observed from Table 8, the precision, recall and f-score for journal extraction using the ElsevierSet are not influenced by this correction. Incorrect splits in block list bibliographies have no influence on the overall accuracy. The journal identification procedure of CiteRep is robust enough to correct for wrong splits. However human readability in the CiteRep web interface does improve when citations are split correctly.

Thresh.	<i>CiteDataSet (250 papers)</i>		<i>ElsevierSet (50 papers)</i>		
	Applies	Alters	Prec.	Rec.	F-Score
0.0	76.0%	57.2%	0.575	0.570	0.573
0.05	2.4%	2.0%	0.575	0.570	0.573
0.07	2.4%	2.0%	0.575	0.570	0.573
0.1	2.4%	2.0%	0.575	0.570	0.573
0.2	0.8%	0.4%	0.575	0.570	0.573
0.5	0.4%	0.0%	0.575	0.570	0.573
0.7	0.4%	0.0%	0.575	0.570	0.573
1.0	0.0%	0.0%	0.575	0.570	0.573

Table 8 Impact of BlockListCorrection on CiteDataSet and ElsevierSet

We have decided to set the threshold value for this correction to 0.1 in our architecture. Although there is no influence on the f-score for any of the threshold values, setting the threshold value to 0.0 would mean that even citation sections without a '['-character would be processed. However, this correction is specifically designed to work on block-list citations. Since not always a block-list character is present, all citations will be concatenated into one large string, lowering human readability and possibly confusing the identification phase for larger document sets. CiteRep enforces that at least 10% of the citation strings should start with a '['-character before the BlockListCorrection is enabled.

4.1.4 BigTextCorrection

The BigTextCorrection is a correction that is applied if the input contains a citation consisting of more than 1,000 characters, indicating a single big piece of text. This piece of text consists of multiple actual citations. No author or number identifications were found that could be used to split it up in previous corrections. The BigTextCorrection has as main purpose to increase human readability by splitting a series of concatenated citations into a list of separate citations. A big text piece of 1,000 characters can also confuse the citation parser in a later stage because known citation patterns cannot be matched to the input. Figure 13 provides a schematic overview of how the BigTextCorrection works.

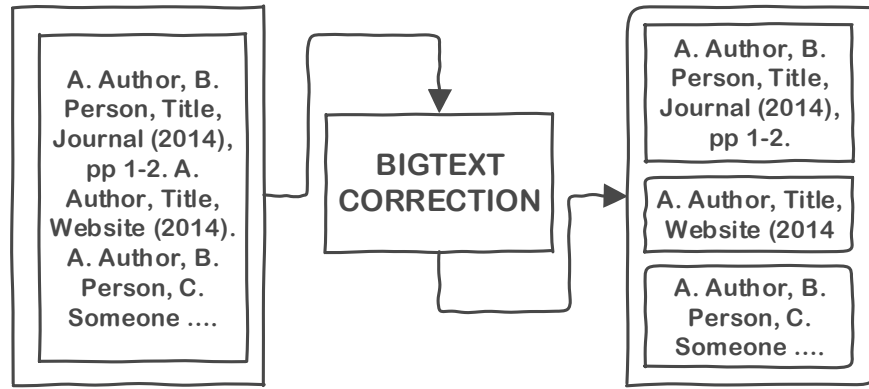


Figure 13 Schematic overview of the BigTextCorrection

The BigTextCorrection works on the basic assumption that each citation contains a year, which consists of four digits possibly encapsulated inside brackets. It scans the text for digits that are likely to represent a year and splits the text on this token. It attempts to determine if the author of the citation comes before or after the split, and intelligently merges it back into the newly created list. Although this approach is prone to errors (e.g. author of previous citation ending up at the next citation), it does result in a list of citations which is consistently formatted. This consistent formatting helps the identification of journals allowing to match known citation patterns.

The threshold value of this correction determines how many citations must have 1,000 characters or more for this correction to apply. The desired behavior is however for this correction to apply if and only if there is at least one citation with 1,000 more characters. Otherwise it might mix up citations and confuse the process later on. Table 9 shows the effect of various threshold values for completeness.

Thresh.	<i>CiteDataSet (250 papers)</i>		<i>ElsevierSet (50 papers)</i>		
	Applies	Alters	Prec.	Rec.	F-Score
0.0	76.0%	27.2%	0.573	0.570	0.571
0.05	12.0%	7.6%	0.575	0.570	0.573
0.07	10.8%	7.2%	0.575	0.570	0.573
0.1	9.6%	6.4%	0.575	0.570	0.573
0.2	8.0%	6.0%	0.575	0.570	0.573
0.5	3.2%	3.2%	0.575	0.570	0.573
0.7	2.4%	2.4%	0.575	0.570	0.573
1.0	0.0%	0.0%	0.575	0.570	0.573

Table 9 Impact of BigTextCorrection on CiteDataSet and ElsevierSet

From Table 9 we observe two distinct cases. The first case makes sure the BigTextCorrection is applied every time, even if no big text is present. This worsens the performance of journal identification on the ElsevierSet because perfectly well organized citation lists are mixed up. The second case is when there is actually at least one citation with 1,000 characters. In this case, human readability greatly improves. The f-score stays the same meaning that the improvement in citation formatting did not enable the detection of new journals. We have set the threshold value to 0.05 in our CiteRep architecture for this correction based on this notion.

4.2 Citation Accuracy

The algorithms used by CiteRep for citation extraction provide us with an answer to the first research question, *how accurately can a list of citations for a publication be extracted from the reference section when given a PDF document as input?* After careful fine-tuning of the various correction thresholds we are able to evaluate the overall performance of the correction chain having enabled all corrections at the same time.

Most corrections do not directly influence the f-score and only improve human readability compensating for mistakes made during the PDF to text conversion process. The impact of the full correction chain on the precision, recall and f-score of the ElsevierSet is shown in Table 10.

<i>CiteDataSet (250 papers)</i>		<i>ElsevierSet (50 papers)</i>		
Applies	Alters	Prec.	Rec.	F-Score
76.0%	56.4%	0.565	0.580	0.573

Table 10 Measurements when all corrections are enabled

As can be observed in Table 10, 56.4% of all papers are altered by the series of corrections. The f-score of the overall citation chain for the ElsevierSet is 0.573, equal to the baseline f-score mentioned in Table 3. Comparing the results to Table 7, the main benefit to increasing the precision and recall is proven to be the logic of the NumberedListCorrection. No further improvement is observed when comparing the baseline measurements to the journals that are found after letting the citations pass through the correction chain. We conclude that the corrections mainly help in making the extracted citation sections easier to read.

Because CiteRep is only concerned with journals that are referenced in a document set, we could have chosen to leave out the correction chain entirely without compromising for overall accuracy of our system. However, the procedure does improve readability because CiteRep stores the citation lists in a clean format. A clean citation list can be of benefit in the future, if the CiteRep software framework is possibly extended with new functionality. The correction chain primarily makes for clean reference lists in our web interface as each intermediate step is made visible to the end-user.

CiteRep’s approach for finding journals in text is proven to be very robust and even works on citation lists that are badly formatted. The impact of the correction chain on overall journal identification accuracy is limited. The next chapter provides insight in how CiteRep identifies the journals within the extracted citation sections.

Chapter 5

JOURNAL IDENTIFICATION

CiteRep identifies journals inside bibliography sections extracted from PDF documents. In this chapter we explain how the identification procedure of the CiteRep architecture works. We start with a schematic overview of the interaction between the journal identification and journal normalization procedures, outlining how they are intertwined in our architecture. The journal normalization procedure, concerned with rewriting journals into one uniform notation, is further detailed in chapter 6.

5.1 Overview

The journal identification and journal normalization steps are intertwined in the CiteRep architecture. The accuracy of the identification phase is improved by taking the normalization procedure into account at an early stage. This enables CiteRep to more reliably convert a piece of text with abbreviations to an actual journal. Figure 14 outlines the overall architecture for journal identification and normalization.

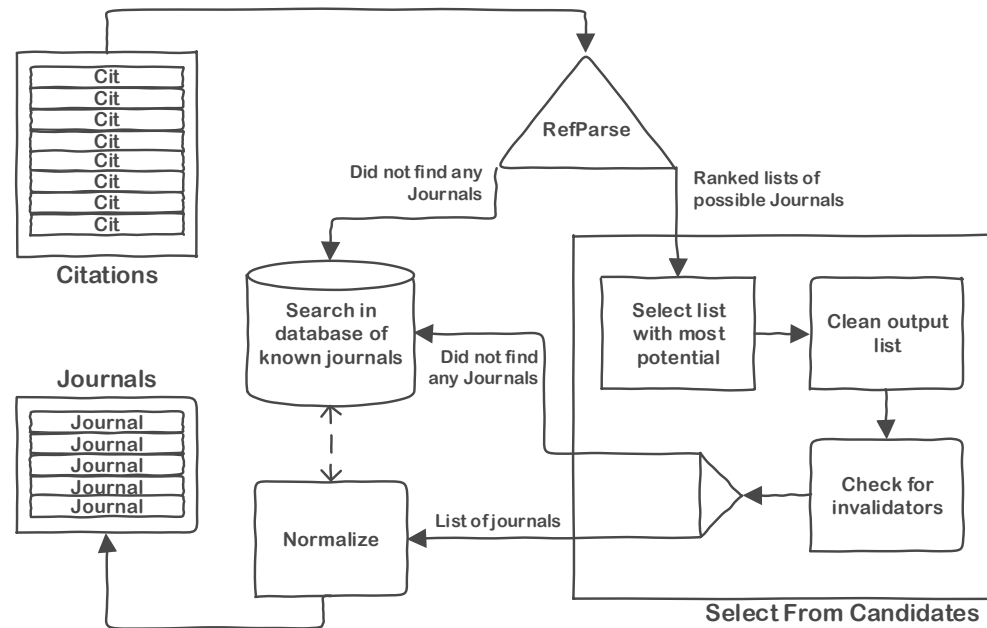


Figure 14 Schematic overview of journal identification and normalization processes in CiteRep’s architecture.

CiteRep incorporates the RefParse [20] library for a first indication as to which parts of a citation actually denote a journal. CiteRep has additional mechanisms to find the journal in a citation if RefParse is unable to find it. The RefParse library and CiteRep’s own mechanisms are detailed in the next sections of this chapter. The journal normalization procedure is detailed in chapter 6.

5.2 RefParse Library

RefParse is an external library used by CiteRep to help find journal references inside citations. Using regular expressions and learned knowledge it splits a citation into parts and annotates each part. RefParse can both be run in a supervised and a standalone mode. In the supervised or interactive mode users can provide feedback on the output which is then used to improve the algorithm within RefParse. Since a requirement of our architecture is the automatic extraction of journals inside citations, CiteRep runs RefParse in the standalone automatic mode. Hence, RefParse is unable to improve itself and will not learn from mistakes. Further effort could be put into allowing user feedback into CiteRep and RefParse but is out of scope for the current implementation. A detailed description of the inner workings of RefParse can be found in the paper by G. Sautter and K. Boehm [20]. For completeness we provide a short summary of how the RefParse library works.

RefParse is a generic library capable of annotating a citation string with entities such as author, title, journal, volume, website and more. RefParse works in several iterative steps based on the core assumption that all bibliographic references in a citation section are formatted the same way. RefParse starts with finding easy to identify sections of a citation which are numbers denoting pagination, edition and year. If multiple four digit numbers are found, RefParse looks at the structure of the other citations to determine which is most probably the year, and which is another number belonging to another entity. RefParse finds the authors by looking at common author characteristics. Together with numeric elements these are called the base elements. The author names are concatenated into the list of authors, and using a majority vote the author name style is determined. Having the author name style, it becomes possible to determine the position of the author list. Together with the list of numeric elements in a citation the most likely reference style is inferred. Now having the citation style, elements such as publisher, proceedings titles and periodical names are extracted. Some portions of the citation are still not processed; the largest still unassigned part of the reference string is labeled to be the title. The title was not determined earlier in the process since it shows the most variation.

In summary, RefParse identifies elements of a given citation by first determining the reference style and then using the reference style to annotate all the pieces in a citation. We have discovered that our input is in many cases not clean enough for RefParse to correctly be processed. Also RefParse is very good at finding well defined elements such as the year of publication, the list of authors and the title. However, since journals are often written in abbreviated format, RefParse is sometimes confused. CiteRep has implemented additional measures to compensate for the cases in which RefParse cannot reliably identify the journal.

5.3 CiteRep Algorithm

The CiteRep algorithm for finding journals in citations comes to aid when the RefParse library cannot reliably identify the journal. The RefParse library produces for each element it has identified (author, title, year, journal, etc.) a list which contains the value for each processed citation. Sometimes the journal ends up at the wrong place. Because the citation list always has a constant format, the error is also consistent. If RefParse mistakes the journal for the publisher, the journal will always be marked to be the publisher for that paper. Our CiteRep algorithm has a *select from candidates* method. This method, as can be observed in Figure 14, takes the output of RefParse and based on characteristics selects the most likely list of elements which contain the journals. If RefParse could not find any journal at all in the citation section, it is send to a database of known papers and a text search is performed. We will explain the working of both cases in the following subsections.

5.3.1 Select RefParse Candidates

Running RefParse on a list of citations yields a list in which recognized entities are annotated. The result is split into separate lists, one for each entity. RefParse has many entities it can detect. The journal entity should end up in the *journal* list if all goes well. Sometimes RefParse makes a mistake, and hence the journal ends up in the wrong list (but does so consistently). The following lists, in decreasing order of likelihood, can hold the journal after RefParse extraction.

journal

journalOrPublisher

proceedingsVolumeTitle

volumeReference

volumeTitle

publisher

title

author

There are even cases in which the journal is marked as being the author. This does happen if the citation section contains many short journals in abbreviated form. In such case the journal *J. Catal* for instance is seen as a paper author. We determine the most likely list to hold the journals by iterating over each list and counting how many journal identifiers are present. For instance, in the example above, the “J. “-part is a clear journal identifier. Each abbreviation that is common in a journal is seen as an identifier. A full list of journal identifiers was derived from our database of known journals as explained in section 5.3.2.

After CiteRep has identified the most likely list to hold the journals (which often is just the journal list itself), the list could still contain flaws. For instance, additional non-journal text might be appended to some journals or the other way round the journal is incomplete and cut off. Even in some cases the entire list might be selected wrong if for instance the authors in a citation contain many abbreviations which are common for journals.

Most journals consist of multiple words and are often prefixed with *J.* or *Proc.* If the list we have selected contains of more than 50% single-word items, we discard it and return empty. In this case we deem that RefParse was not able to accurately find the journals in the list of citations. Following the logic of Figure 14, CiteRep might still be able to find the journals by performing a lookup in the database of known journals.

The list that holds the journals is cleaned by trimming off excess symbol characters and is then send through a check which checks for invalidators. An invalidator is a piece of text that never occurs in a common journal title. The complete list of journal invalidators used by CiteRep is given below in Table 11.

Van de	University of	Last modified	Retrieved
Patent	We have	Bounded by	This is because
Web site	Website	Was seen	Seen in
Which is	Isbn	University	Et al

Table 11 Journal invalidators

The journal is removed from the list of journals if an invalidator was found. Also if the journal is most likely to be a formula it is removed. CiteRep checks if a journal is a formula by counting the occurrence of special characters that are common in formulas. Regular expressions are used to further clean up the results by removing common prefixes such as “paper at the ...”, “reprinted from ...” or “in ...”.

It could still be the case that arbitrary text, which looks like a journal, is appended before the actual journal. CiteRep performs a check to compensate for this. If a journal in the list of journals has more characters in it than the average journal length of that list, we check if we can truncate all characters before the first known journal identifier inside that specific item. Journal identifiers are the known abbreviations and prefixes obtained by analysing the database of known journals as explained in section 5.3.2.

If the entire procedure as described above fails, and hence RefParse did not reliably produce a list of journals, the original list of citations is fed into CiteRep’s search algorithm. The search algorithm uses a database of known journals to look for the journal in a citation as explained in the next section.

5.3.2 Database of Known Journals

In order to improve the accuracy of the identification process, CiteRep makes use of a comprehensive database roughly containing 130,000 unique journals and conference proceedings from all areas of research and from many different countries. The database contains common abbreviations and short notations used by publishers. The database was specially created for CiteRep and as far as we know there is no other journal list publicly available which is free to use and contains this many known journals. The list was compiled from crawling over 20 different sources such as online journal indexes, publisher price lists, university libraries and official lists maintained by Thomson Reuters and Elsevier. The compiled journal list is open-source and freely available for download from GitHub [32].

The database is stored in a single, column separated, text file. The first column contains the original journal title, consecutive columns contain known abbreviations, alternative names and short codes if known. An example entry from our database is shown in Figure 15.

```
aids research and human retroviruses
•  aids res hum retrov
•  aids res hum retrovir
•  aids res hum retrovirus
•  aids res hum retroviruses
```

Figure 15 Example journal database entry

The constructed database of known journals allowed us to compile a list of common journal abbreviations. As can be observed from Figure 15, ‘research’ is abbreviated with ‘res’, ‘human’ is abbreviated with ‘hum’ and retroviruses has several abbreviations. We wrote a simple one-time learning algorithm that takes numerous list of journals as input, removes common words such as ‘in’, ‘and’, ‘of’ and then checks what the most likely mapping is between the full notation and the abbreviated notation. When CiteRep found 100 journals in the database having the same confirmed abbreviated mapping, the abbreviation was added to a new database of known abbreviations. This database is used throughout our architecture and also made freely available for anyone to use on GitHub [33]. Once an abbreviation was learned, the journal list was simplified and related abbreviations left out as the abbreviations could now be automatically generated using the abbreviation database.

The journal list and the known abbreviations are loaded in the system main memory upon the start of CiteRep. In memory search enables quick lookups of journals and abbreviations. In addition to known alternate spellings, additional alternate spellings are also generated for each journal by using the abbreviation list as reference. By doing so, CiteRep increases the chance to find a journal if a new paper uses a slightly different way of abbreviating than was previously observed.

The database of known journals is used to search for journals inside a list of citations if the RefParse algorithm did not succeed in finding the journals. Please note that it could be the case that the database is never used if RefParse correctly identifies the journals. In that case CiteRep directly branches to the normalization step as can be seen in Figure 14. For the cases in which RefParse fails to produce a complete list of journals, or if CiteRep observes that RefParse did not find all journals, the journal database is searched. Also if RefParse could not find journals in 30% or more of the citations, a database lookup is always performed and the largest result set is used.

Please note that the database lookup will only return results that are in the database. The RefParse library and selection procedure however allow for unknown journals to be found. CiteRep is not restricted to the database of known journals. New journals and conferences can still be identified correctly. A piece of text starting with ‘proc ...’ will for instance always be marked as a proceeding, even if the conference proceedings is not known in the compiled database.

CiteRep’s algorithm to search the database of known journals first splits the citation into parts on the common delimiters ‘,’ ‘.’ And ‘:’. All known abbreviations which are followed by a dot are replaced by the same abbreviation without the dot to prevent journal notations from being split. An example of the procedure of splitting a citation is given in Figure 16.

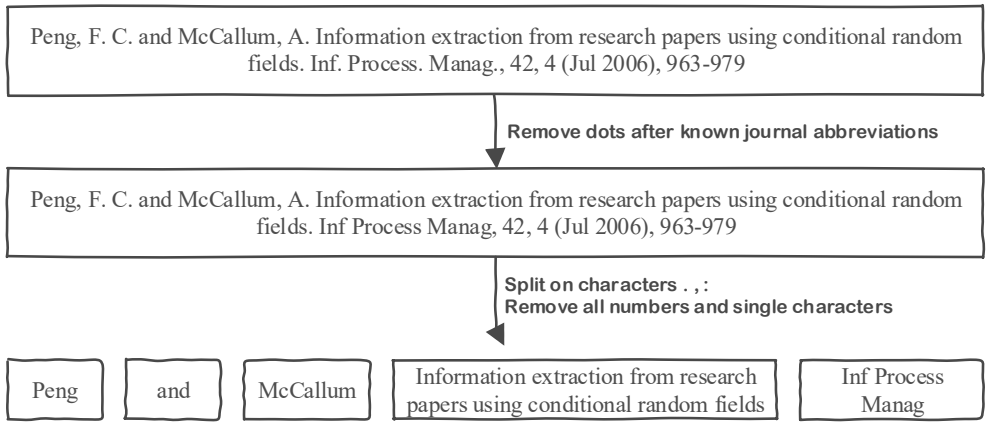


Figure 16 Example of splitting a citation into searchable parts

CiteRep performs a lookup of each part in the database, using several normalizations tactics allowing for spelling differences (see chapter 6). If the journal is found, its original title is looked up in the main memory database and counted for journal citation report statistics. In the case of Figure 16, the journal that is returned by the database is *Information Processing and Management*.

We have found the CiteRep procedure for identification be very powerful in practice. Almost any arbitrary text can be given as input, and the journal will be found if it follows commonly known abbreviations and exists in the database. We have enabled this procedure to also find multiple journals in a single piece of text. A true citation never has references to multiple journals, but a parse error earlier on in the CiteRep

architecture might have for instance concatenated several citations into a single citation. CiteRep is very robust and corrects mistakes made in the citation phase at the identification phase.

Please note that it is not feasible to feed the entire plain text of the PDF document through this method since it is very CPU intensive and database lookups for spelling variations are time expensive. We conclude this chapter by making claims about the accuracy of CiteRep’s journal identification procedure.

5.4 Identification Accuracy

CiteRep’s algorithm for journal identification as outlined in this chapter is capable of accurately finding the journal in a citation, answering the second research question *How accurately can the journal be identified inside a citation?*

CiteRep’s journal identification procedure is very robust and can handle input errors and correct for mistakes that are made by either the author of the document or the PDF parser. The manually created journal and abbreviation databases aid CiteRep with identifying a journal in almost any piece of text.

For evaluation of the accuracy of CiteRep’s journal identification phase we used the CoraDataSet. The Cora test set is standardized and created by Andrew McCallum. The Cora set was previously used in other research to assess the performance of related citation tools, allowing to compare CiteRep to other implementations.

Table 16 from Chapter 7 shows that CiteRep achieves a journal identification accuracy of 73.7 percent, clearly outperforming other citation parsers such as ParsCit (49.7%), FreeCite (52.1%) and RefParse (53.4%). CiteRep performs significantly better than existing technologies when it comes to journal identification.

Chapter 6

JOURNAL NORMALIZATION

Normalization is the process in which CiteRep rewrites differences in journal notation into a single normal form. The CiteRep journal normalizer was already briefly introduced in the overall architecture shown in Figure 14. Normalizing the journals enables CiteRep to generate overall statistics of commonly used journal. If CiteRep would not normalize, differences in spelling would count as entirely new journals in citation reports. Hence CiteRep would not be able to say which journal is for instance most popular, but only which specific notation of a journal is most popular. This chapter presents the inner workings CiteRep’s normalizer.

6.1 Overview

The normalization process consists of two main methods. The first method, called SimpleJournal, is used to remove common pieces of text from a journal which do not uniquely identify that journal. The SimpleJournal method is also able to abbreviate words provided the abbreviation is known. The normalizer will always attempt to return the most abbreviated notation as explained in section 6.1.1.

The second method, called NormalizeJournal, is concerned with looking up journal spelling variations in the database of known journals. It performs a series of ten modifications on the input journal title, and after each modification checks if it shows up in the database. If found, the resulting journal is included in the journal citation report. This procedure is further explained in section 6.1.2

6.1.1 SimpleJournal

The SimpleJournal method removes a list of common words from the journal notation in an attempt to make the result more generic, yet not ambiguous. For instance, the input “*journal of the academy of management, volume 6 (2012)*” will become “*academy management*”. SimpleJournal starts text processing by first removing all text between brackets. All non a-z characters are removed among words that are very common to a journal but do not uniquely identify that journal. Removing these words from journal A and journal B enables us to perform string comparison to see if these journals are actually different spellings of the same journal. Table 12 lists the words that are common in journal notations. These words are removed by the CiteRep SimpleJournal method.

Of the	Of	No	Nd	Th	Volumes	journal
Ed	And	Edition	Vols	volume	J	Proceedings
		Parts	part	Proc	Vol	

Table 12 Common words in a journal notation

After removing common words from a journal notation, CiteRep proceeds with what could be seen as minifying the text. If there exists a known abbreviation in our database, the abbreviation is applied to the journal at hand. For example, the journal “academy management” is minified to “acad manag”. The word *management* shows up in our abbreviation list and hence it is shortened to *manag*. The word *academy* is also minified to *acad* accordingly.

CiteRep uses the SimpleJournal method to determine if different spellings actually represent the same journal in reality. Figure 17 shows an example of several differences in spelling which are all mapped to the same output result.

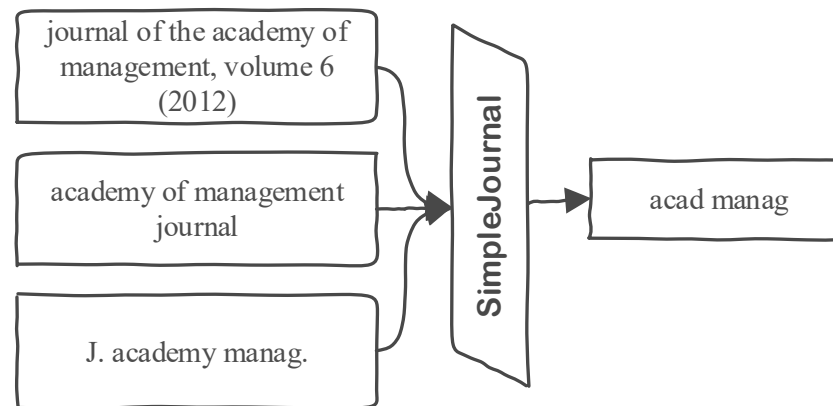


Figure 17 Example of the SimpleJournal method

The pitfall of this method is that one can go too far with simplifying journal notations, possibly leading to journals which are actually different from each other to have the same simplified notation. When the list of abbreviations for CiteRep was created, we attempted to only include abbreviations which do not cause distinct journals to be mapped to the same normal form.

The list of abbreviations was manually validated, removing abbreviations that would result in many distinct words being mapped to the same generic representation. Still the normalization process can be flawed. For instance, *acad* might refer to *academy* or *academic*, hence if there exists a journal which is called *journal of Academic Management* it would also have the same minified representation *acad manag* from Figure 17 albeit being a different journal in reality. CiteRep stores each original representation with the normalized outcome to make what happened transparent to the end-user.

The SimpleJournal method is a trade-off between enabling journals to be compared to one another, and losing the ability to keep distinct journals apart due to shared abbreviations. Running a test on our list of 130,000 known journals yield 3,939 journals which are distinct but have the same SimpleJournal notation if we process abbreviations. There are 3,348 journals with the same base representation if only the common words from Table 12 are replaced and CiteRep does not normalize for known abbreviations.

Further effort could be put into reducing these false positives for the SimpleJournal method. Because of time constraints we have decided that an error margin of 3% is acceptable at this time for the CiteRep normalization procedure. The SimpleJournal method allows to search the database in the NormalizeJournal procedure, taking differences in journal notation into account.

6.1.2 NormalizeJournal

The NormalizeJournal method is a chain of ten actions performed on a journal title, until a match in the database of known journals is found. NormalizeJournal is used by the CiteRep architecture to normalize a journal using a database search. The NormalizeJournal method itself uses the SimpleJournal method as part of its process. Ten actions are performed in order and the chain is broken if the journal is found in the database after applying the action prescribed by the chain. Each action of the chain takes the original journal as input, performs some modification and then looks up the journal in the database. Checks performed are shown in Figure 18.

1. Keep the original input
2. Process journal using SimpleJournal, but do not parse abbreviations
3. Process journal using SimpleJournal, parsing abbreviations
4. Find part in text that contains "... in proc (a-z)"
5. Correct for PDF parsing errors by replacing i-l-1 characters
6. Find part in text that contains "... journal of (a-z)"
7. Same as 6, but also parse abbreviations
8. Remove number representations such as "twelfth edition"
9. Same as 8, but also parse abbreviations
10. If we have not found the journal in the database, but the input text starts with "jour" or "proc", consider it valid as well.

Figure 18 Action chain for the NormalizeJournal procedure

For actions 1-9 the outcome of the prescribed action is used to lookup the journal in the database. If all actions fail to produce a result from the database, CiteRep performs a last check (action 10). This allows to normalize a journal which is not known by the database if the text itself contains a clear identifier that it is indeed a journal or a conference proceeding.

We conclude this chapter by benchmarking the performance of the SimpleJournal and NormalizeJournal procedures. The combination of these procedures allows CiteRep to accurately normalize for differences in journal notation.

6.2 Normalization Accuracy

We have developed the SimpleJournal and NormalizeJournal procedures in answer to the third research question, *how accurately can journal titles be normalized to compensate for abbreviations and spelling differences?* To prove these methods significantly improve the accuracy of finding journals in citation sections we have temporarily disabled the SimpleJournal and NormalizeJournal procedures and measured the influence of doing so on CiteRep’s overall performance. The effect is shown in Table 13.

<i>Normalization</i>		<i>ElsevierSet (50 papers)</i>		
SimpleJ	NormalizeJ	Prec.	Rec.	F-Score
Off	Off	0.0	0.0	0.0
Off	On	0.0	0.0	0.0
On	Off	0.107	0.406	0.170
On	On	0.577	0.570	0.573

Table 13 Effect of disabling SimpleJournal and NormalizeJournal

Turning off the NormalizeJournal procedure greatly decreases the precision of CiteRep. Disabling the database of known journals means it could not be used to assist with identifying journals in a text. Disabling the SimpleJournal method results in a f-score of zero, meaning that for the journals that still could be extracted, there is not a single journal extracted by CiteRep which is normalized to exactly the same notation as used by Elsevier in the validation set.

The Elsevier journals in the Elsevier database are normalized using some undisclosed procedure by Elsevier. To compensate for this behavior, the Elsevier journal notations are also normalized using the SimpleJournal procedure in all comparisons made throughout this paper. Since errors in normalization of the Elsevier set are consistent over time, they do not influence the relative change in f-score when running benchmarks using the ElsevierSet.

Mistakes made during the normalization of the Elsevier journals do negatively count towards the absolute f-score of CiteRep. The only way to overcome this problem was to manually craft a golden standard following the exact journal notation from the papers. Table 15 from Chapter 7 shows that CiteRep achieves an f-score of 66.2% on this dataset, performing significantly better when compared to the ElsevierSet.

Chapter 7

EVALUATION

The citation, identification and normalization phases of CiteRep architecture provide a complete set of algorithms and procedures to obtain a list of journals from any given document repository. CiteRep is our answer to the main research question:

How can existing document repositories be used to provide insight in the usage of journals in publications by university students and staff?

We provide the reader with an overview of the most referenced journals at the University of Twente. The web interface that comes with CiteRep can be used to generate detailed citation reports tailored to end-user requirements on the fly. The results provided in this section 7.2 are indicative of the capabilities of CiteRep. The overall accuracy of CiteRep is assessed using test sets which were not used before.

7.1 CiteRep Overall Accuracy

Up till this point claims in this paper are made based on the CiteDataSet and the ElsevierSet. These validation sets were extensively used to benchmark individual components and to fine-tune threshold values. Because we have iterated many times on these validation sets, CiteRep might be biased towards its input. To make claims about the overall accuracy of the CiteRep tool we used two test sets containing documents our system has not seen before. Table 14 shows the overall performance of CiteRep on a set of 80 papers annotated by Elsevier. Table 15 displays the performance of CiteRep on a set of 40 documents for which the journals were manually annotated. Details about the test sets and the procedure of calculating precision, recall and f-score are discussed in detail in Section 3.3.

<i>ElsevierStandard (80 papers)</i>		
Prec.	Rec.	F-Score
0.544	0.528	0.536

Table 14 CiteRep accuracy using the Elsevier test set

<i>TestSet (40 papers)</i>		
Prec.	Rec.	F-Score
0.748	0.594	0.662

Table 15 CiteRep accuracy using our manually created test set

Elsevier uses undisclosed normalizations in their dataset, negatively influencing the overall performance of CiteRep compared to our own test set. CiteRep is shown to achieve a precision of 74.8% and a recall of 59.4% on a fresh set of documents with annotated journal references. Our TestSet used for benchmarking is made open-source available and can be downloaded from GitHub [34].

There are several other citation parsers which can identify journals when provided with a citation list. We conclude by comparing CiteRep to these existing citation parsers. For comparison we use the annotated Cora Corpus [29]. This dataset was also used by other well-known citation parsers to benchmark their algorithms. This enables us to compare CiteRep to other parsers. For comparison we use the field accuracy metric as defined in [35]. The field accuracy provides insight in how many of the data elements of a bibliographic reference string are identified correctly as explained by [36]. Please note that CiteRep is able to both identify journals and conference proceedings without distinguishing between them. Other citation parsers distinguish between *journal* and *volume title* whereby conference proceedings are identified as being a volume title entry. Other citation parsers also identify entries such as books and documents whereas CiteRep is only concerned with journals. When comparing CiteRep to the performance of these parsers we took the average field accuracy of the journal and volume title fields to allow fair comparison. Table 16 shows the accuracy of CiteRep compared to standalone RefParse [20], ParsCit [21] and FreeCite [37]. The measurements for comparison were copied from [20].

<i>Algorithm</i>	<i>Cora Dataset (500 papers)</i>
	Field Accuracy
ParsCit	49.7%
FreeCite	52.1%
RefParse	53.4%
CiteRep	73.7%

Table 16 Comparison of CiteRep journal identification performance to ParsCit, FreeCite and Refparse

CiteRep outperforms existing citation parsers when it comes to identifying journals and proceedings in a bibliography. CiteRep is the preferred method to use when interested in automatically extracting journals and proceedings from citation sections. Note that the other software solutions used for comparison are able to identify all elements in a citation section, whereas CiteRep is limited to journals and conference proceedings. As shown in Figure 14, CiteRep incorporates RefParse at its core to provide pointers as to where the journal resides in the citation. Table 16 shows CiteRep succeeded in improving the accuracy of RefParse for the specific purpose of finding journals in citations.

7.2 Journal Usage at the University of Twente

CiteRep was used to indexed all papers published in document repositories at the University of Twente. The repositories contain 60,721 papers published between Jan 1965 and May 2016. For some entries of the database the source PDF is missing or the PDF is an image file instead of readable text. In total 51,379 readable documents were processed by CiteRep. The distribution of all of the documents processed by CiteRep is shown in Figure 19 .

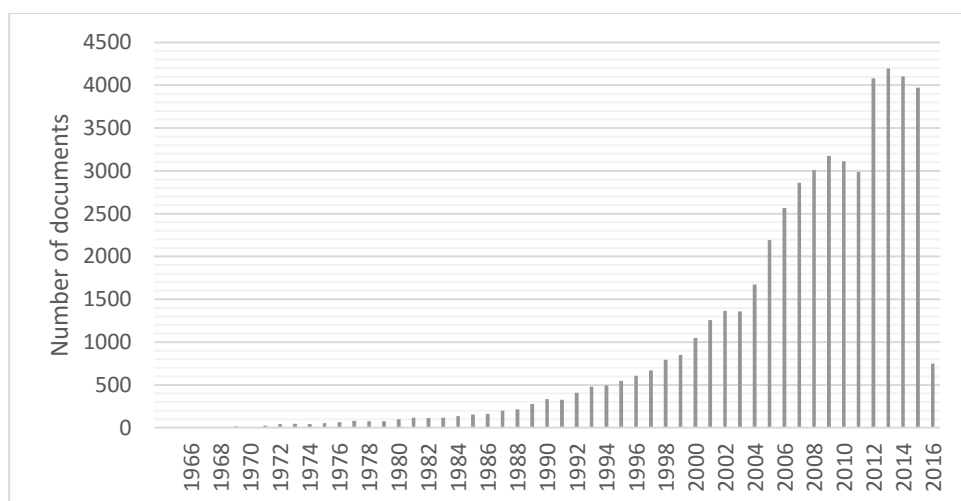


Figure 19 Distribution of documents in UT repositories per year

A total of 845,938 journal references were found in the documents processed from the university repositories. Figure 20 displays the top 20 overall journals of the University of Twente based on the absolute count of journal references from all citations that CiteRep was able to process.

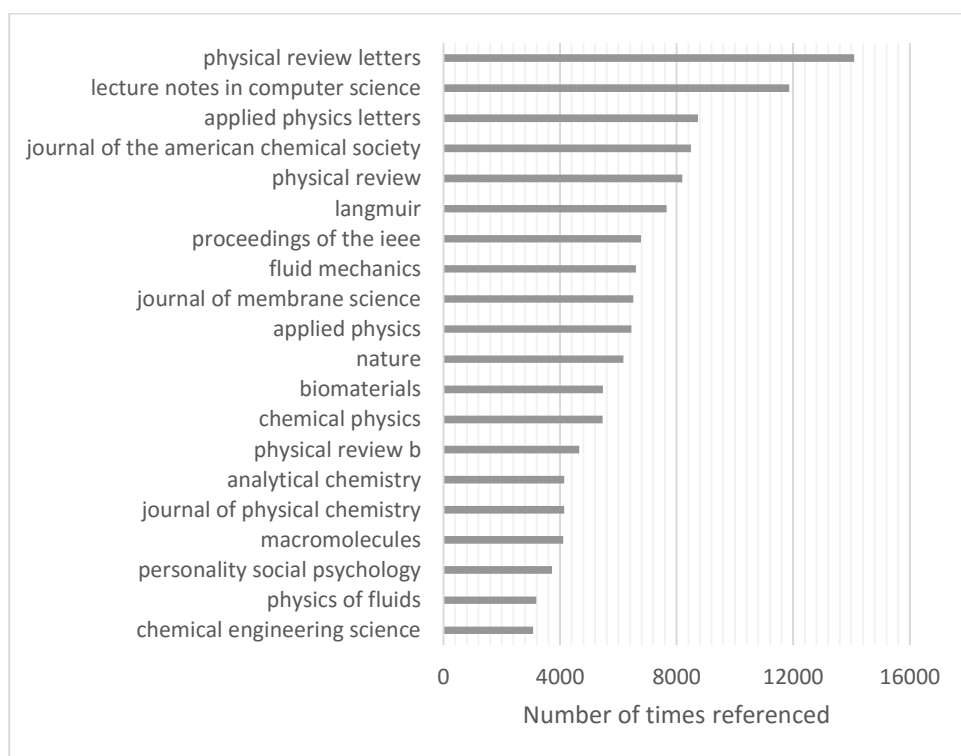


Figure 20 Overall top 20 journals at the University of Twente

For most of the source documents metadata is available denoting which faculty has published a specific document in the repository. Figure 21 shows the distribution of documents per faculty. For each faculty the total number of entries in the repository, the amount of documents that could be processed, and the documents from which journals could be identified is shown.

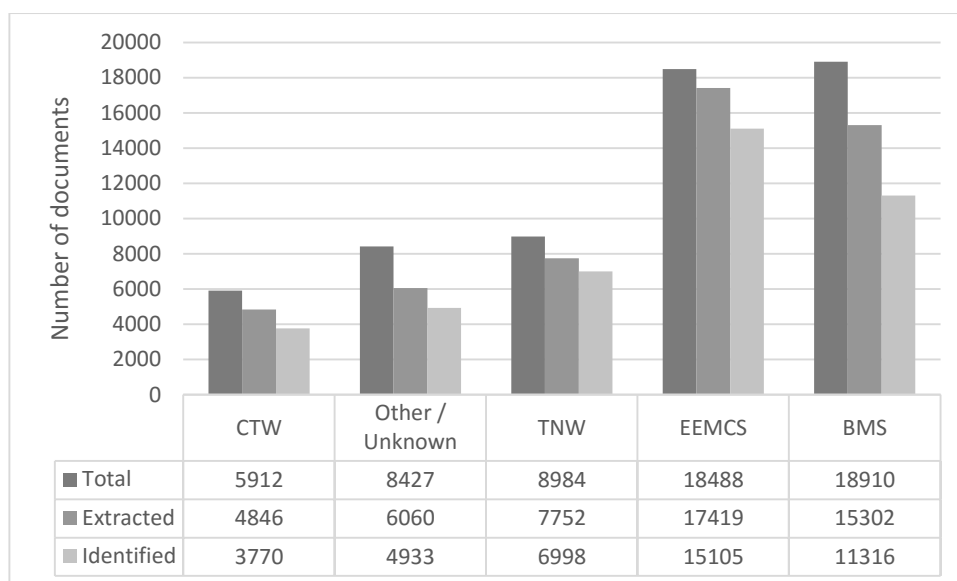


Figure 21 Distribution of documents per faculty

CiteRep is not able to process all documents from the university repositories. Some document entries simply have no associated PDF file or access to the PDF file is restricted. Sometimes the PDF to text conversion fails because the PDF document is an image file. Text could be extracted for 51,379 documents of the total of 60,721 entries in the repository. Journal identification did not succeed for all documents which were extracted to machine readable text. CiteRep was able to find journals in citations from 42,122 out of 51,379 documents as can be observed from Figure 21.

We conclude this chapter providing an overview of the top 10 journals referenced by students and staff for the four largest faculties at the University of Twente between the five-year period of January 2000 to December 2015. Additional journal citation reports can be generated on demand using the CiteRep web dashboard. Figure 22 - Figure 25 are indicative of how CiteRep can be used for journal analytics.

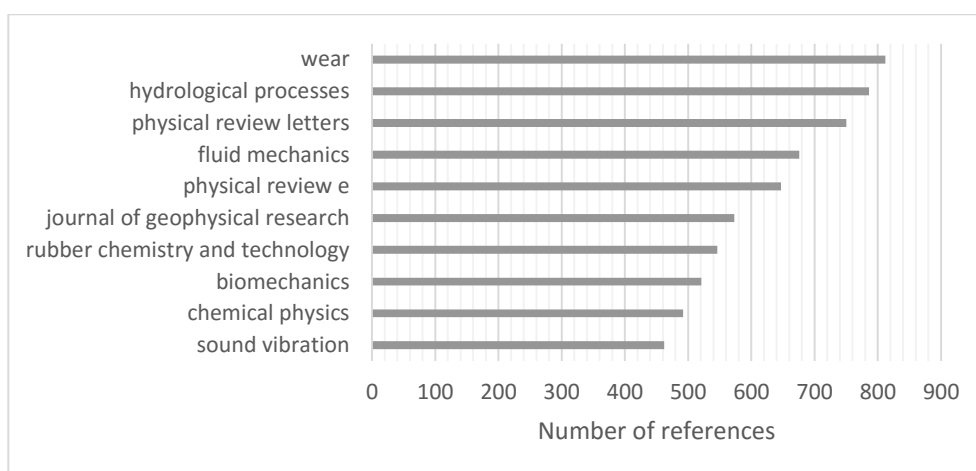


Figure 22 Top 10 journals for the CTW faculty between 2000-2015

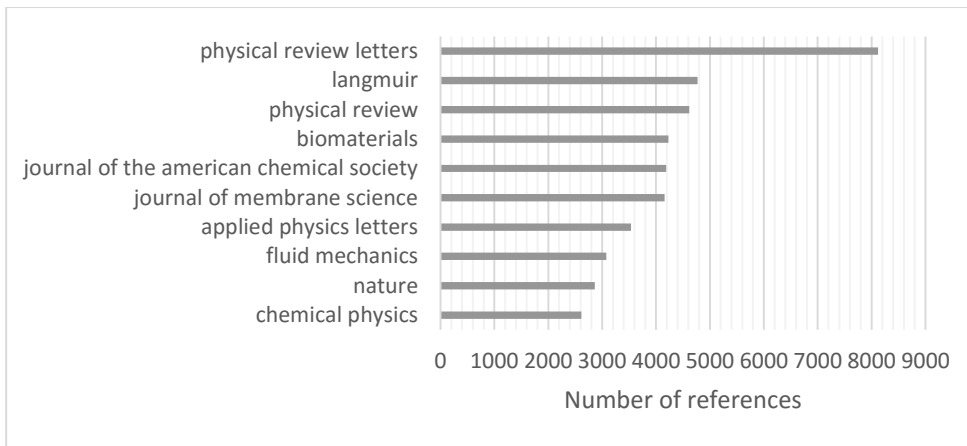


Figure 23 Top 10 journals for the TNW faculty between 2000-2015

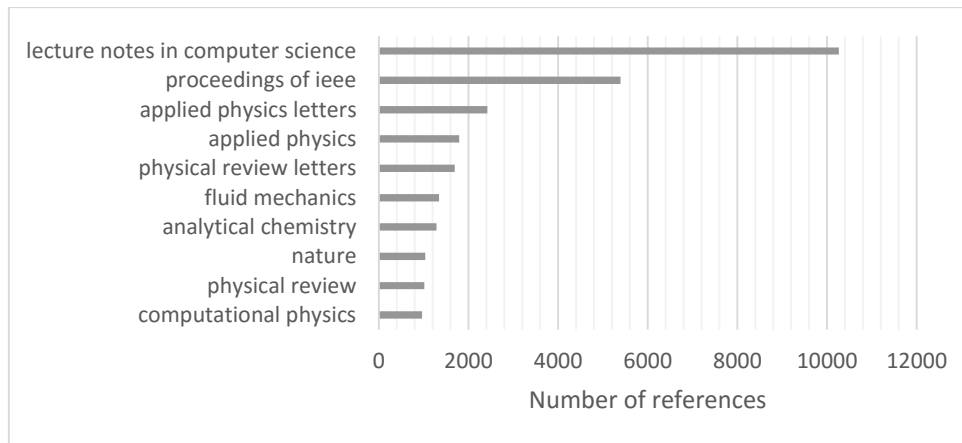


Figure 24 Top 10 journals for the EEMCS faculty between 2000-2015

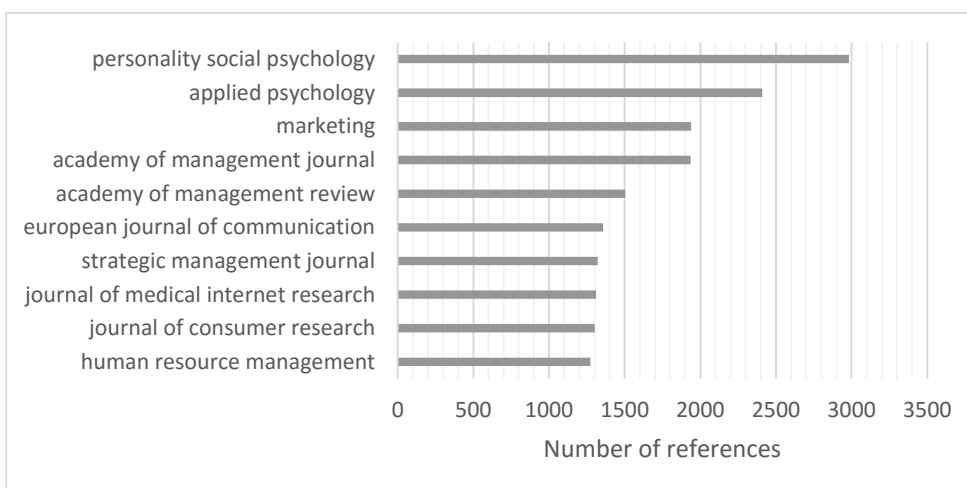


Figure 25 Top 10 journals for the BMS faculty between 2000-2015

Chapter 8

CONCLUSION

This research originated from the business faculty at the University of Twente asking for a way to count journal references in citations from library document repositories. CiteRep provides the University of Twente with a structured procedure for counting journal references in documents written by university students and staff. CiteRep generates journal citation reports using a three phase process of document citation extraction, journal identification and journal normalization. CiteRep outperforms the two most popular open-source citation tools ParsCit and FreeCite with respect to identifying journal references in citations.

Documents are automatically acquired by CiteRep from repositories using the Open Archives Protocol for communication. PDF documents are converted using Apache PDFBox into machine readable text. The citation extraction procedure of CiteRep is able to find the bibliography section inside the extracted text, automatically correcting for text formatting mistakes made during the PDF to text conversion. CiteRep uses RefParse, an open-source generic citation processor, to identify journal references inside citations. Additional mechanisms are included in CiteRep to aid with journal identification when RefParse does not succeed. CiteRep is shown to outperform all other citation libraries with regard to identifying journals within a document bibliography. CiteRep incorporates a unique normalization algorithm to normalize different journal notations into a single uniform notation. The normalization procedure enables CiteRep to count unique journals, providing the end-user with detailed citation reports of journal references in the document set.

CiteRep can aid universities with journal subscription decision making processes, providing insight in which journals are actually used by students and staff. The online dashboard interface of CiteRep provides means to filter journal citation reports based on time intervals, study program and faculty. The web interface can easily be actualized when a new document is published at the University of Twente, allowing for real-time analysis. CiteRep can be easily installed on any machine, running any operating system.

Installation instructions and source-code of CiteRep are freely provided to the reader under the permissive MIT license at <https://github.com/sverkuil/CiteRep>

Chapter 9

FUTURE WORK

In this thesis CiteRep is shown to be able to find and extract citation sections from PDF documents. From the citation section CiteRep is able to extract the journals and using the dashboard web interface journal citation counts are displayed in a user-friendly environment. CiteRep can be used to discover trends over time and aid with decision making concerning journal subscription packages.

The overall accuracy of CiteRep can still be improved, specifically by reducing the number of false positives in the NumberedListCorrection. Sometimes numbered lists that exist in a PDF document are wrongly marked as a citation list, increasing the chance that non-journals from these pieces of text are wrongly marked as journals. Threshold values and regular expressions used for specific sub routines within the NumberedListCorrection should be thoroughly evaluated and optimized.

Further research could look into implementing a learning algorithm for CiteRep. RefParse, which is incorporated in CiteRep, already allows for some basic learning techniques which are currently not enabled in CiteRep because of the lack of a feedback loop. CiteRep could learn from its mistakes by allowing the end-user to provide feedback. It is expected that this approach increases the precision and reduces the number of false positives. Because our architecture is distributed in nature there can be multiple workers running on separate machines. One would have to think of a mechanism to distribute what is learned to all the workers. Also the system would not run entirely autonomous anymore as end-users are required to provide feedback to improve the accuracy of the system.

CiteRep uses a manually compiled database of known journals and common abbreviations to aid with journal identification. There are however constantly new journals published which the system should learn. Although CiteRep can process unknown journals using other techniques, its accuracy increases if the journal is known. A crawler which frequently indexes online journal lists and journal modification logs could be build and integrated with CiteRep. By doing so newly added journals and changes in journal titles are reflected within CiteRep.

We have observed that the list of known abbreviations integrated into CiteRep greatly helps with normalizing for different journal notations. However, this list is not complete. We discovered abbreviations that were unknown by the system, resulting in CiteRep not being able to identify all journals in a bibliography. A helper algorithm could be written which accurately compiles a more complete list of abbreviations using the main journal database as source.

We think that it would be helpful to create an automatic procedure which scans the internet and searches for the journal impact factor, adding this knowledge to CiteRep. Doing so yields two benefits. The first benefit would be for the end-user to correlate the most popular journals at the University of Twente with the journal impact factor. The second benefit would be for the CiteRep algorithm itself. If CiteRep were to consequently find a journal in many citations, but that journal is actually not that popular in practice or does not exist at all, CiteRep might for instance wrongly identify a common author name as being a journal. Misidentifying journals could come from wrong information in our journal database or errors made during the simplification and normalization steps.

There are presumably many other approaches that could be taken at improving the accuracy of CiteRep. We invite other researchers to download and run our tool and we look forward to see what they come up with!

APPENDIX A – COMMUNICATION PROTOCOL

The CiteRep worker and dashboard communicate with each other using a simple challenge response communication protocol. The worker sends regular heartbeat messages at most every 5 seconds to poll the dashboard, much like in a client server polling scheme. If there tasks available to perform, the server responds with a task to perform. This task can be either one of *extract*, in which a PDF document needs to be processed into plain text, or of type *identify* indicating to extract the citation section and identify journals within citations. The identify task will also trigger the normalization procedure in which spelling differences are converted into one uniform notation.

All requests from the worker to the dashboard are HTTP POST requests [38]. The responses are in the standardized JavaScript Object Notation format [31]. Tasks created in the dashboard environment are processed as a first in first out message queue. Tasks assigned to remote workers have at most one hour to complete. If a task fails to complete within the specified time frame it is released and rescheduled. CiteRep ensures that tasks cannot rest idle forever if a remote worker is forcefully quit or a networking error occurs. Most tasks take a few seconds of computing time to complete. If a heartbeat request to the dashboard results in a tasks that should be performed, and that task is completed before the next heartbeat message is send in the 5 second timeframe, the heartbeat message is send early to quickly obtain a new task to perform. Each task spawns a local CPU thread at the machine it runs on, enabling a single worker to work on several tasks concurrently.

A schematic overview of a typical sequence of messages between the worker and the online dashboard using regular heartbeat messages is shown in Figure 26.

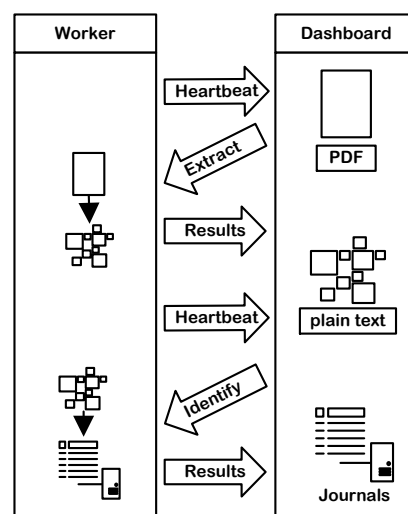


Figure 26 Schematic overview of the worker-dashboard communication protocol

The communication protocol example shown in Figure 26 represents the translation from a PDF document to a list of journals via extract and identify tasks. The extract task sends the location of a PDF file to the worker. The worker, which is typically run inside the university network, downloads the PDF file in main memory for processing. The PDF file is converted into plain text using Apache PDFBox and the citation section is extracted. The result is stored in the database at the dashboard machine. The identify operation sends the citation section back to the worker and the worker processes it to find all journals that are being referenced. A list of journals is send back to the dashboard, completing the processing for that specific document.

The server side PHP dashboard and client side Java worker are designed in such a way that new operations could easily be added in the future. All communication, socket handling and possible networking errors are handled gracefully by the CiteRep architecture. Future additions and modifications can be performed easily without having to learn the specifics of the underlying networking protocol.

The networking protocol allows for basic password based authentication. The password can be found after login in the dashboard web interface. CiteRep ensures that only trusted workers can connect to the web dashboard. Before a worker is connected to the dashboard a simple handshake takes place. This handshake procedure validates if the worker is running an up to date version of the software and if the password matches that of the server. Workers that are successfully connection show up in the dashboard and start processing available tasks.

The extract and identify tasks are at a lower level a simple sequence of HTTP requests carrying request parameters as POST variables and return variables as a JSON body. Figure 27 shows a basic message exchange for an extract task.

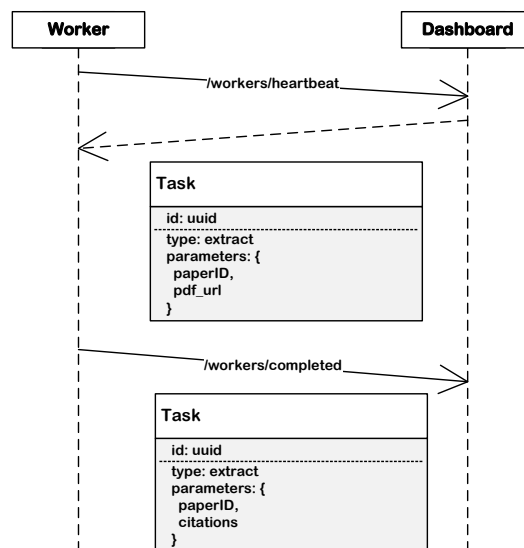


Figure 27 Example worker-dashboard network requests

The CiteRep architecture is secure and modular by design. It is designed with future development in mind and is easily extended to incorporate new functionality. CiteRep follows current standards such as JSON and REST making it future proof and easy to understand and maintain. All used third-party libraries and frameworks within CiteRep are open-source and well maintained by a large community. Even if someone is unfamiliar with the technology used by CiteRep, ample documentation and support exists to extend the CiteRep software.

BIBLIOGRAPHY

- [1] HEIMERL, F., HAN, Q., KOCH, S., and ERTL, T., 2016. CiteRivers: Visual Analytics of Citation Patterns. *IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS* 22, 1 (Jan), 190-199. DOI= <http://dx.doi.org/10.1109/Tvcg.2015.2467621>.
- [2] COLLINS, T., 2012. The current budget environment and its impact on libraries, publishers and vendors. *Journal of library administration* 52, 1, 18-35.
- [3] LEMLEY, T. and LI, J., 2015. "Big Deal" Journal Subscription Packages: Are They Worth the Cost? *Journal of Electronic Resources in Medical Libraries* 12, 1, 1-10.
- [4] BJORK, B.C. and SOLOMON, D., 2012. Open access versus subscription journals: a comparison of scientific impact. *BMC Med* 10, 10.1, 73. DOI= <http://dx.doi.org/10.1186/1741-7015-10-73>.
- [5] BOTERO, C., CARRICO, S., and TENNANT, M.R., 2011. Using comparative online journal usage studies to assess the Big Deal. *Library Resources & Technical Services* 52, 2, 61-68.
- [6] KURMIS, A.P., 2003. Understanding the limitations of the journal impact factor. *J Bone Joint Surg Am* 85, 12, 2449-2454.
- [7] LAGOZE, C., SOMPEL, H.V.D., NELSON, M., and WARNER, S., 2015. The Open Archives Initiative Protocol for Metadata Harvesting <https://www.openarchives.org/OAI/openarchivesprotocol.html>.
- [8] YANG, K. and MEHO, L.I., 2006. Citation analysis: a comparison of Google Scholar, Scopus, and Web of Science. *Proceedings of the American Society for Information Science and Technology* 43, 1, 1-15.
- [9] JACSO, P., 2005. As we may search-Comparison of major features of the Web of Science, Scopus, and Google Scholar citation-based and citation-enhanced databases. *CURRENT SCIENCE-BANGALORE*- 89, 9, 1537.
- [10] C. LEE GILES, K.D.B., STEVE LAWRENCE, 1998. CiteSeer: An Automatic Citation Indexing System. *Proceedings of the third ACM conference on Digital libraries*.
- [11] LEY, M., 2012. The DBLP Computer Science Bibliography: Evolution, Research Issues, Perspectives. *String Processing and Information Retrieval*, 2476, 1-10.
- [12] JIE TANG, J.Z., LIMIN YAO, JUANZI LI, LI ZHANG, ZHONG SU, 2008. ArnetMiner: Extraction and Mining. *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 990-998.
- [13] MIN-YUH DAY, T.-H.T., CHENG-LUNG SUNG, CHENG-WEI LEE, SHIH-HUNG WU, CHORNG-SHYONG ONG, WEN-LIAN HSU, 2005. A Knowledge-based Approach to Citation Extraction. *IEEE International Conference on Information Reuse and Integration*, 50-55.

- [14] SHIH-HUNG WU, M.-Y.D., WEN-LIAN HSU, 2002. FAQ-centered organizational memory. *Knowledge Management and Organizational Memories*, 103-112.
- [15] PENG, F.C. and MCCALLUM, A., 2006. Information extraction from research papers using conditional random fields. *Information processing & management* 42, 4 (Jul), 963-979. DOI=<http://dx.doi.org/10.1016/j.ipm.2005.09.002>.
- [16] SUTTON, C. and MCCALLUM, A., 2006. *An introduction to conditional random fields for relational learning*. Introduction to statistical relational learning. MIT Press.
- [17] KRISTIE SEYMORE, A.M., RONALD ROSENFELD, 1999. Learning hidden Markov model structure for information extraction. *AAAI-99 Workshop on Machine Learning for Information Extraction*.
- [18] CHIEN-CHIH CHEN, K.-H.Y., HUNG-YU KAO, JAN-MING HO, 2008. BibPro: A Citation Parser Based on Sequence Alignment Techniques. In *22nd International Conference on Advanced Information Networking and Applications*, 1175-1180.
- [19] I-ANE HUANG, J.-M.H., HUNG-YU KAO, WEN-CHANG LIN, 2004. Extracting Citation Metadata from Online Publication Lists Using BLAST. In *Advances in Knowledge Discovery and Data Mining*, 539-548.
- [20] SAUTTER, G. and BÖHM, K., 2014. Improved bibliographic reference parsing based on repeated patterns. *International Journal on Digital Libraries* 14, 1-2, 59-80.
- [21] ISAAC G. COUNCILL, C.L.G., MIN-YEN KAN, 2003. ParsCit: An open-source CRF reference string parsing package. In *Joint Conference on Digital Libraries*, 37 - 48.
- [22] LAGOZE, C. and VAN DE SOMPEL, H., 2001. The Open Archives Initiative: Building a low-barrier interoperability framework. In *Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries* ACM, 54-62.
- [23] CLARK, J. and DEROSE, S., 1999. XML path language (XPath) version 1.0.
- [24] LAGOZE, C. and VAN DE SOMPEL, H., 2003. The making of the open archives initiative protocol for metadata harvesting. *Library hi tech* 21, 2, 118-128.
- [25] LIPSON, C., 2011. *Cite right: a quick guide to citation styles--MLA, APA, Chicago, the sciences, professions, and more*. University of Chicago Press.
- [26] LEYDESDORFF, L., 1994. The Generation of Aggregated Journal-Journal Citation Maps on the Basis of the Cd-Rom Version of the Science-Citation-Index. *Scientometrics* 31, 1 (Sep), 59-84. DOI=<http://dx.doi.org/Doi 10.1007/Bf02018102>.
- [27] REUTERS, T., 2009. Journal Title Abbreviations http://images.webofknowledge.com/WOK46/help/WOS/A_abrvjt.html.
- [28] BAEZA-YATES, R. and RIBEIRO-NETO, B., 1999. *Modern information retrieval*. ACM press New York.

- [29] MCCALLUM, A., 2016. Cora Information Extraction
<https://people.cs.umass.edu/~mccallum/data.html>.
- [30] ARNOLD, K., GOSLING, J., HOLMES, D., and HOLMES, D., 2000. *The Java programming language*. Addison-wesley Reading.
- [31] BRAY, T., 2014. The javascript object notation (json) data interchange format.
- [32] VERKUIL, S., 2016. CiteRep Journal List
<https://github.com/SVerkuil/CiteRep/tree/master/client/CiteRepData/journals.txt>.
- [33] VERKUIL, S., 2016. CiteRep Abbreviation List
<https://github.com/SVerkuil/CiteRep/tree/master/client/CiteRepData/abbreviations.txt>.
- [34] VERKUIL, S., 2016. CiteRep Test Set
<https://github.com/SVerkuil/CiteRep/tree/master/client/TestSet/>.
- [35] KRÄMER, M., KAPRYKOWSKY, H., KEYSERS, D., and BREUEL, T., 2007. Bibliographic meta-data extraction using probabilistic finite state transducers. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on IEEE*, 609-613.
- [36] DAY, M.Y., TSAI, R.T.H., SUNG, C.L., HSIEH, C.C., LEE, C.W., WU, S.H., WU, K.P., ONG, C.S., and HSU, W.L., 2007. Reference metadata extraction using a hierarchical knowledge representation framework. *Decision Support Systems* 43, 1 (Feb), 152-167. DOI=
<http://dx.doi.org/10.1016/j.dss.2006.08.006>.
- [37] GOLDBERG, M., 2016. FreeCite Citation Parser
<http://freecite.library.brown.edu/>.
- [38] FIELDING, R., GETTYS, J., MOGUL, J., FRYSTYK, H., MASINTER, L., LEACH, P., and BERNERS-LEE, T., 1999. *Hypertext transfer protocol--HTTP/1.1*.