What Others Say About This Work? Scalable Extraction of Citation Contexts from Research Papers

Petr Knoth[1(✉)], Phil Gooch[2], and Kris Jack[2]

[1] Knowledge Media Institute, The Open University, Walton Hall, Milton Keynes MK7 6AA, UK
petr.knoth@open.ac.uk

[2] Mendeley Ltd., Elsevier B.V., 14-18 Finsbury Square, London EC2A, UK

Abstract. This work presents a new, scalable solution to the problem of extracting citation contexts: the textual fragments surrounding citation references. These citation contexts can be used to navigate digital libraries of research papers to help users in deciding what to read. We have developed a prototype system which can retrieve, on-demand, citation contexts from the full text of over 15 million research articles in the Mendeley catalog for a given reference research paper. The evaluation results show that our citation extraction system provides additional functionality over existing tools, has two orders of magnitude faster runtime performance, while providing a 9% improvement in F-measure over the current state-of-the-art.

Keywords: Information extraction · Citation extraction · Text-mining · Digital libraries

1 Introduction

There are already over 114 million academic papers on the Web [1]. With over 1 million papers published each year [2] and an estimated 10% year on year increase in the annual number of these outputs [3], researchers need tools to help them decide what to read. While recommendation systems for academic papers, such as those provided by Google Scholar, Mendeley Suggest [4] or CORE [5,6] have been created to address the problem of discovering relevant literature, more can be done to help users to effectively navigate through the network of scientific papers. One traditional yet effective way of discovering new and relevant content is by following the edges of the citation graph in the opposite direction, i.e. from the cited to the citing articles. Unfortunately this activity is, even in the most popular scholarly communication systems, not adequately supported. Although users can discover articles that cite a particular work, Google Scholar and similar services do not enable the user to quickly understand how important and relevant to their interest that citation link is, prior to accessing that document.

Similarly, users of academic digital libraries need to make choices about what to read. When presented with a particular article landing page, they need to decide if investing time in reading the article is worthwhile. Such decision is typically based on (a) the perceived relevance of the article to the current researcher's interest and (b) the importance or trust in the work.

While the former is typically assessed by scanning the abstract and title, the latter is today often evidenced using the paper's citation count, typically displayed on the article details page, the journal impact factor or other similar metric. However, all approaches relying on an aggregate function of citation counts to evidence the importance of an article face problems caused by the variety of situations in which people cite work [7]. As described by Eugene Garfield [8], the motivations for citing prior work include: paying homage to pioneers, giving credit for related work (homage to peers), identifying methodology, equipment, and the like, providing background reading, correcting one's own work, correcting the work of others, criticising previous work, substantiating claims, alerting researchers to forthcoming work, authenticating data and classes of fact (such as physical constants), identifying original publications in which an idea or concept was discussed, identifying the original publications describing an eponymic concept or terms, arguing against the work or ideas of others and disputing the claims of others to have been first with their work.

As a result, we believe researchers can benefit from leveraging citations in a qualitative rather than just quantitative way. Citation contexts, i.e. the text surrounding a citation, explain how the cited paper is used in this particular work. By extracting all these mentions from the full text of articles citing a document of interest, we can help researchers to quickly explore the ways in which a given paper was useful in other peoples' work, hence we can help them decide whether the work might be useful in their own work. Our assumption is that by enabling researchers to quickly interrogate the contexts in which a given paper is used, we can assist them in making a more informed choice about whether or not to read it.

Consequently, we address the problem of automatically retrieving and extracting citation contexts for a given research paper. The presented work brings the following contributions:

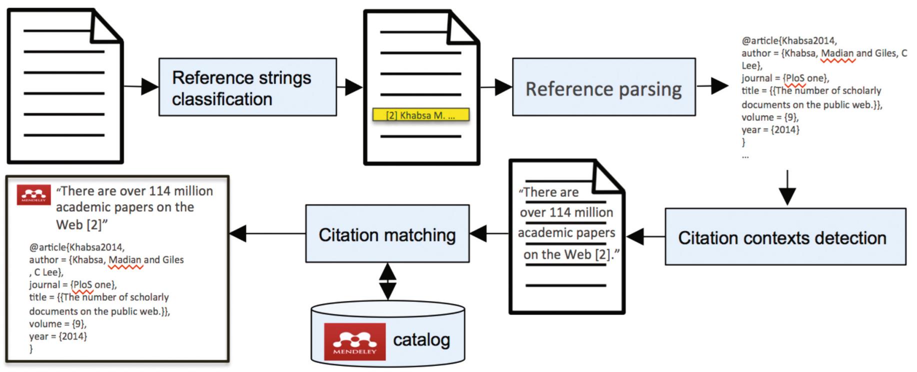
– We present a new, scalable tool which uses machine learning techniques to recognise and parse references from unstructured text and extracts the textual content surrounding their mentions.
– We report on the results of an end-to-end evaluation of this tool and discuss its advantages over existing solutions.

In addition to the above mentioned use cases, we believe this work could also be applied in other situations, in particular, (a) to improve browsing in digital libraries by enabling more focused navigation across resources (e.g. it would be possible to (hyper-)link to a particular fragment rather than just to a document), (b) as a tool to assist researchers/funders in understanding which claims from a work have been discussed and/or built upon in further work and

(c) as a supporting tool to understand the contribution of a researcher, a research group, organisation, etc., at a finer granularity.

2 Citation Contexts Extraction Method

The citation contexts extraction process consists of four stages depicted in Fig. 1. We first clean and pre-process the input text. We then process the text line by line classifying each as either a reference line or not (Sect. 2.1). The lines classified as a reference are then passed to a probabilistic parser based on Conditional Random Fields (CRFs) [9] that splits each reference into its constituent fields (Sect. 2.2), such as authors, title, year and venue. We subsequently use a set of regular expressions to link each reference to all its citations in the processed document extracting all the citation contexts (Sect. 2.3). Finally, we try to link each of the citation reference strings to a unique ID of the cited document (Sect. 2.4).


Reference strings classification
[2] Khabsa M. ...
Reference parsing
Citation contexts detection
"There are over 114 million academic papers on the Web [2]."
Citation matching
catalog
"There are over 114 million academic papers on the Web [2]"

```
@article{Khabsa2014,
author = {Khabsa, Madian and Giles, C Lee},
journal = {PloS one},
title = {{The number of scholarly documents on the public web.}},
volume = {9},
year = {2014}
}
...
```

Fig. 1. The four stages of citation context extraction.

2.1 Reference Classification

The following features were used to train the classifier to distinguish the text of references from non-references at the line level:

- **(F1) Line length** (float): Character line length as a ratio to the mean line length in the document.
- **(F2) Is within reference or numbered block** (Bool) Follows a heading that signifies the start of a references block, or within a block of at least n consecutively numbered lines.
- **(F3) Date presence** (Bool)
- **(F4) Contains URL or DOI** (Bool)

- **(F5) Contains cue words** (Bool): Contains words that likely appear in a reference, such as “ibid”, “et al.” or “in press”
- **(F6) Contains publication word stem** (Bool): Contains a word from a list (“journal, adv, bull, proc, stud, biochem, etc”)
- **(F7) Contains a page range** (Bool)
- **(F8) Contains volume information** (Bool) Contains words, such as “vol”, or common “vol/issue/pages” stereotype pat- terns.
- **(F9) Contains editor information** (Bool): Words, such as (eds, ed. etc.)
- **(F10) Punctuation ratio** (float): Ratio of punctuation characters to the total line characters.
- **(F11) Capital letters ratio** (float): Ratio of capital letters to total line characters.
- **(F12) Camel case bigrams ratio** (float): Ratio of camel case bigrams to the line bigrams.
- **(F13) Starts with a numeric label** (Bool)
- **(F14) Surname/initials pair** (Bool): Contains a surname or initials pairs.

Using the above features, we have trained SVM, Random Forest, Logistic Regression, Decision Tree and CRF models. The first four models were created using the WEKA [10] software workbench. We used the CRF++ toolkit [11] implementation for the CRF model. This model uses, in addition to the above mentioned features, sequence information consisting of feature values of each line and the preceding and following 3 lines.

2.2 Reference Parsing

Given the plain text of a reference in any bibliographic format, the goal of reference parsing is to fill in a template consisting of fields, such as author, title, journal and year. To solve this problem we apply the AnyStyle parser[1], which is an open-source tool that uses CRFs to split a reference string into its constituent fields and output the result as BibTeX. The parser ships with a default model and training data that consist of 657 records of annotated data. To increase the accuracy of the parsing, we have retrained the CRF model using additional training data from Mendeley (see Sect. 4.2).

2.3 Context Extraction

Citation context extraction addresses the problem of locating all the links in the body of a paper to each reference and extracting the text surrounding them. There are three main approaches of connecting a citation to its reference we support:

- The reference is preceded with a number which is used as citation marker in the body of the document.

[1] http://anystyle.io/.

- The reference is preceded with an abbreviation created, for example, as a name & year, which is used as a citation marker in the body of the document.
- The citation is linked to a reference using a footnote.

We approach the problem of linking the citation to a reference by using a set of regular expressions. These regular expressions were manually curated and fine-tuned on a test set. Using a naive baseline method, the citation context snippet is then formed by a context window of 300 characters to the left and right of the position of the citation.

2.4 Citation Matching

The final step is to link each of the references cited in a given research paper to a unique document ID of the cited document. We use the catalog search functionality of the Mendeley API for this purpose. For each parsed reference, we compose a query using the following logic:

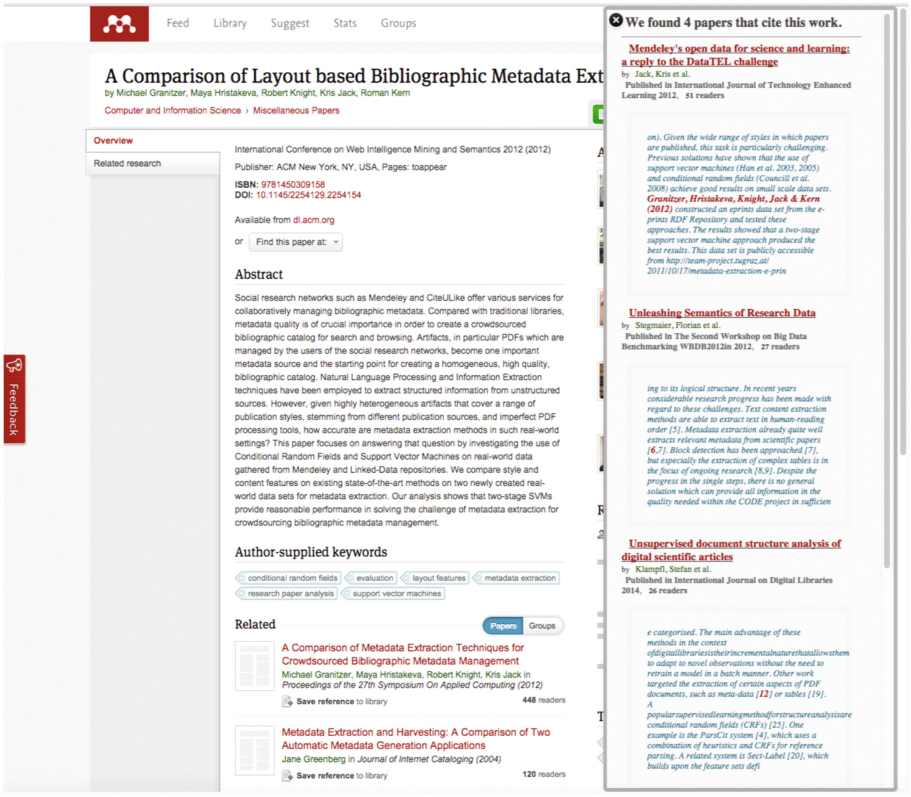
- If the reference string contains an identifier, such as a DOI, we use this identifier to look up the record.
- If we manage to extract the title from the reference string, the query contains the title plus year and author information, provided this is available.
- If the reference does not contain (or we don't manage to extract) an identifier nor a title, such as in "J. A. Maruhn and W. Greiner, Z. Phys. 251, 431 (1972).", we fall-back to a fuzzy lookup on author, source and year.

If the look up is successful for a reference, we record the document ID and normalise the parsed metadata based on the information in the Mendeley catalog.

3 Collecting Information About What Others Say About This Work

As our goal is to retrieve all the citation contexts for a given reference article, we need a fast way of determining the set of articles citing the reference article. We have considered two approaches of addressing this problem. In the first approach, we would apply the citation extraction tool described in Sect. 2 to create a catalogue of research articles and their citations. We would start by deduplicating research articles and adding all of them into the catalogue with their metadata. We would then process the full texts of all these papers. For each paper, we would extract and parse its references and would try to match each reference to this catalogue using a learnt similarity threshold for a metric, such as Jaccard coefficient. We would then generate a pair of citing catalogue ID and cited catalogue ID for each successfully matched tuple.

As this process contains many non-trivial steps where errors can occur, we have decided to opt for an alternative approach which relies on already



A Comparison of Layout based Bibliographic Metadata Ext

by Michael Granitzer, Maya Hristakeva, Robert Knight, Kris Jack, Roman Kern

Computer and Information Science › Miscellaneous Papers

International Conference on Web Intelligence Mining and Semantics 2012 (2012)

Publisher: ACM New York, NY, USA, Pages: toappear

ISBN: 9781450309158
DOI: 10.1145/2254129.2254154

Available from dl.acm.org

Abstract

Social research networks such as Mendeley and CiteULike offer various services for collaboratively managing bibliographic metadata. Compared with traditional libraries, metadata quality is of crucial importance in order to create a crowdsourced bibliographic catalog for search and browsing. Artifacts, in particular PDFs which are managed by the users of the social research networks, become one important metadata source and the starting point for creating a homogeneous, high quality, bibliographic catalog. Natural Language Processing and Information Extraction techniques have been employed to extract structured information from unstructured sources. However, given highly heterogeneous artifacts that cover a range of publication styles, stemming from different publication sources, and imperfect PDF processing tools, how accurate are metadata extraction methods in such real-world settings? This paper focuses on answering that question by investigating the use of Conditional Random Fields and Support Vector Machines on real-world data gathered from Mendeley and Linked-Data repositories. We compare style and content features on existing state-of-the-art methods on two newly created real-world data sets for metadata extraction. Our analysis shows that two-stage SVMs provide reasonable performance in solving the challenge of metadata extraction for crowdsourcing bibliographic metadata management.

Author-supplied keywords

conditional random fields, evaluation, layout features, metadata extraction, research paper analysis, support vector machines

Related

A Comparison of Metadata Extraction Techniques for Crowdsourced Bibliographic Metadata Management
Michael Granitzer, Maya Hristakeva, Robert Knight, Kris Jack in *Proceedings of the 27th Symposium On Applied Computing (2012)*
448 readers

Metadata Extraction and Harvesting: A Comparison of Two Automatic Metadata Generation Applications
Jane Greenberg in *Journal of Internet Cataloging (2004)*
120 readers

We found 4 papers that cite this work.

Mendeley's open data for science and learning: a reply to the DataTEL challenge
by Jack, Kris et al.
Published in International Journal of Technology Enhanced Learning 2012, 51 readers

*on). Given the wide range of styles in which papers are published, this task is particularly challenging. Previous solutions have shown that the use of support vector machines (Han et al. 2003, 2005) and conditional random fields (Councill et al. 2008) achieve good results on small scale data sets. **Granitzer, Hristakeva, Knight, Jack & Kern (2012)** constructed an eprints data set from the e-prints RDF Repository and tested these approaches. The results showed that a two-stage support vector machine approach produced the best results. This data set is publicly accessible from http://team-project.tugraz.at/2011/10/17/metadata-extraction-e-prin*

Unleashing Semantics of Research Data
by Stegmaier, Florian et al.
Published in The Second Workshop on Big Data Benchmarking WBDB2012in 2012, 27 readers

*ing to its logical structure. In recent years considerable research progress has been made with regard to these challenges. Text content extraction methods are able to extract text in human-reading order [5]. Metadata extraction already quite well extracts relevant metadata from scientific papers [**6**,7]. Block detection has been approached [7], but especially the extraction of complex tables is in the focus of ongoing research [8,9]. Despite the progress in the single steps, there is no general solution which can provide all information in the quality needed within the CODE project in sufficien*

Unsupervised document structure analysis of digital scientific articles
by Klampfl, Stefan et al.
Published in International Journal on Digital Libraries 2014, 26 readers

*e categorised. The main advantage of these methods in the context ofdigitallibrariesistheirincrementalnaturethatallowsthem to adapt to novel observations without the need to retrain a model in a batch manner. Other work targeted the extraction of certain aspects of PDF documents, such as meta-data [**12**] or tables [19]. A popularsupervisedlearningmethodforstructureanalysisare conditional random fields (CRFs) [25]. One example is the ParsCit system [4], which uses a combination of heuristics and CRFs for reference parsing. A related system is Sect-Label [20], which builds upon the feature sets defi*

Fig. 2. The "What cites this work" browser bookmarklet allows the user to see how a given research paper is cited in other peoples work. Instead of having to search for the mentions of the paper, the citation context is automatically extracted and displayed to the user in the form of a snippet.

existing databases. More specifically, we use the Mendeley Catalogue as an already deduplicated database of research papers. The Mendeley Catalogue contains over 70 million unique research articles crowd-sourced from Mendeley users. As we host the full text documents uploaded by users on our servers, we can process these articles using the citation context extraction tool. In order to identify the articles citing a given document from the Mendeley Catalogue, we rely on information from Scopus. Scopus is one of the largest citation databases of peer-review literature. As the citation information in Scopus is automatically extracted and then manually curated, it is of high quality. The dump of the Scopus citation dataset we used in our experiments consisted of over 934 million citation pairs.

To enable a fast retrieval of document IDs citing a given paper, we first match Scopus citing-cited article pairs to the Mendeley Catalogue. We then group these pairs by the cited document aggregating all citing document IDs on one line. Finally, we index this dataset using Elasticsearch.

To retrieve the citation contexts for a given research paper, all we need to do now is to:

1. Query the citations index to retrieve Mendeley IDs of all documents citing a given reference document.
2. Download the full texts of the citing documents.
3. Process the documents using the citation context extraction tool retrieving only the citation contexts referring to the reference document.
4. Retrieve canonical metadata for each of the citing references from the Mendeley Catalogue to accompany the citation context with information about the source it comes from.
5. Rank the citations according to some criterion and reorder.

We have built a demonstrator in the form of a "What cites this work" browser bookmarklet that implements this method for articles in the Mendeley Catalogue. As the user launches the bookmarklet, the title and the DOI of the currently visited article is retrieved from the HTML page of the Mendeley Catalogue. The system then follows steps 1–5, where steps 2–3 are done in parallel. Our demonstrator ranks the retrieved citations according to the popularity of the citing article, i.e. based on the number of Mendeley readers who have that article in their library. An example of the result is shown in Fig. 2. While it has recently become the de facto standard in scientific databases and search engines to provide a link to a list of articles citing a given article, these systems typically do not show the context in which the article is cited. For example, the ACM Digital Library displays only the list of article titles that cite a given document and Google Scholar displays the list of titles with their abstracts but not with the citation contexts. We believe that using citation contexts as snippets is more informative and useful to the user.

4 Evaluation

There is a number of components that form our pipeline. In order to get a good understanding of the system's performance, we have to evaluate all of them.

4.1 Reference Classification

To train the reference classification models described in Sect. 2.1, we have created a training set of 1,000 randomly selected PDFs from the Mendeley Catalogue for which we have canonical citation data from Scopus. These PDFs were converted to a single text file consisting of about 300 k lines and all actual citations were labeled 1 and non-citations 0. The validation set consisted of 1,000 PDFs (365 k lines) randomly selected from PubMed labeled in the same way as the training data.

The results (Table 1) show that SVM, CRF and J48 were the top performers. However, the observed run-time performance of the CRF was much faster than the other two methods, which is why we decided to use this model within our tool.

Table 1. Evaluation of line-level citation classifiers.

Model	Precision	Recall	F-measure
SVM	0.998	0.998	0.998
CRF	0.996	0.997	0.997
J48	0.997	0.997	0.997
Random Forest	0.993	0.993	0.993
Log Regression	0.990	0.990	0.990

Table 2. Parsing error rate for evaluation data set of 26,000 citation records.

Model	Parse error rate
Baseline	22%
Retrained model	5%

4.2 Reference Parsing

One of the difficulties with parsing a citation string is dealing with noisy data that might be extracted from, for example, PDF files. There may be inconsistencies in text encoding, punctuation and use of white space. In addition, many different citation referencing styles need to be handled. We created training data representative of a wide range of citation styles and included noisy data exactly as extracted from the source document. The training set consisted of 600 manually structured citations from open access papers in the Mendeley catalogue in addition to the 657 training records supplied with the Anystyle parser[2].

In order to see whether the CRF model trained on the additional data performs better than the default model shipped with AnyStyle, we created an evaluation set of 26,000 structured citation records randomly selected from PubMed research papers. The size of the data sample gives us at 99.9% significance level a confidence interval of just below 0.1. The evaluation references were then compared with the system generated references to calculate a raw error rate based on character-level differences.

A comparison of the error rate between the system-generated references and those from PubMed are shown in Table 2. By retraining this parser on a more representative data sample that included an additional 600 records, we have reduced the error rate of the baseline citation parser from 22% to 5% - more than a four-fold error reduction. Our intuition behind the significant error reduction is that the default AnyStyle model was trained on too little and too clean data.

4.3 End-to-end Citation Extraction

We ran two end-to-end evaluations against 26,000 research article PDFs randomly selected from the Mendeley Catalogue and for which canonical citation data were available from the manually curated Scopus database. The gold data comprises the raw string value and structured citation (author, title, year, source), and the expected catalogue identifier for each reference cited.

This evaluation faced significant challenges, such as that the extracted citations might be in a different citation format than in the canonical record as well as that the PDF may be locked by the creator or may be a scan.

[2] http://anystyle.io/.

For the first evaluation (Fig. 3), each extracted citation was parsed with the CRF model described in Sect. 2.2 and the authors, title, year, and source and DOI fields extracted were used in a query to the Mendeley Catalogue lookup API. Results of this evaluation are shown in Table 3.

For the second evaluation, we attempted to match extracted citation strings against the canonical citation strings for each article. We also ran the same end-to-end evaluation with the state-of-the-art CERMINE [12] software on the same hardware to compare performance both in terms of accuracy and processing speed. Results of this evaluation are shown in Table 4.

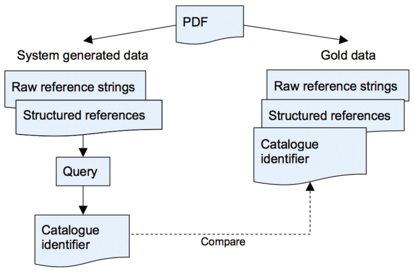


Fig. 3. Evaluation pipeline comparing retrieved identifiers with expected identifiers in the gold data set.

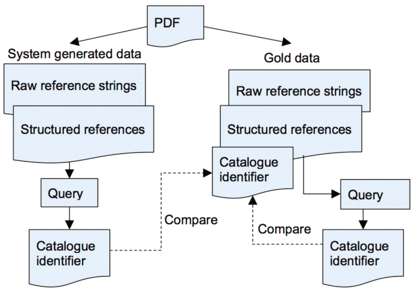


Fig. 4. Evaluation pipeline with optimal baseline data that uses the gold data to generate queries.

Table 3. End-to-end citation extraction and linking via parsing and querying.

Canonical citations	Extracted references	Matched references	False matches	Precision	Recall	F-measure
731,119	731,119	453,750	127,469	0.776	0.621	0.690

Table 4. End-to-end citation extraction and matching via hashing, P = precision, R = recall, F = F-measure, t = time in seconds.

Canonical citations	Extracted references	Matched references	False matches	Precision	Recall	F-measure	time (seconds)
Our system	887,191	657,277	238,696	0.734	**0.741**	**0.737**	$5.5x10^3$
CERMINE	887,191	524,248	187,331	**0.789**	0.591	0.676	$3.46x10^5$

Table 5. Optimal catalogue matching performance given a query string from the gold set.

Measure	Score
Precision	0.78
Recall	0.72
F-measure	0.76

Table 6. System catalogue matching performance given an automatically generated query string.

Measure	Score
Precision	0.78
Recall	0.63
F-measure	0.69

While the individual results for citation classification and citation parsing show high performance, the end-to-end results for citation network indicates that there is space for improvement. This may be because the matching of extracted citations against their canonical form needs to be more sophisticated or that the catalogue lookup is too imprecise. However, the end-to-end results still represent an overall 9% F-measure improvement on the previous state-of-the-art using the same evaluation set and using the same metrics, and our approach also runs two orders of magnitude faster (10^3 vs 10^5 seconds to complete) on the same hardware.

In order to distinguish catalogue lookup errors from errors in the citation extraction pipeline, we decided to compare the system performance against an optimal baseline. This optimal baseline helps us to answer the question of what would be the maximum achievable performance for locating the correct catalogue entry if perfect, structured citations could be extracted automatically from each evaluation article. To do this, for each structured citation in the gold set we generated a catalogue query (in the same way as was done for the system generated structured citation) and compared the identifier of the returned result with the expected identifier in the gold set (Fig. 4).

A comparison of the end-to-end system evaluation with the optimal baseline is shown in Tables 5 and 6. Table 5 shows the results of the optimal baseline evaluation when comparing the retrieved catalogue identifier given a "perfect" query string generated from the gold data, with the expected identifier in the gold set, for each document in the evaluation set. The idea here is that a successful lookup is not guaranteed even when the query is generated from the gold dataset. We want to see how this optimal baseline performs to be able to compare it with the citation extraction system performance.

Table 6 shows the end-to-end system results evaluated by comparing the retrieved catalogue identifier given a query string generated from automated extraction and parsing, with the expected identifier in the gold set, for each document in the evaluation set.

The results show that the system performs at $\frac{0.69}{0.76} = 91\%$ of the optimum that could be expected if a perfect query string could be generated for each document in the evaluation set.

5 Related Work and Discussion

Previous approaches such as ParsCit [13] use heuristics to identify the block of references at the end of the article and regular expressions to split these into individual references strings. In this work, we use CRFs to identify individual reference strings anywhere within the document, which allows references in, for example, footnotes to be extracted. Similar to ParsCit, our approach requires only plain text as input.

Other approaches, such as SectLabel [14] and pdfextract use rich-text features such as font size, position, and indentation to identify reference sections in order to improve extraction performance. Although our approach leads on our evaluation set excellent results without requiring such features, it may well be improved with the addition of them. This assumption is consistent with the findings of Kern & Kampfl [15] who enriched ParsCit with features, such as font information, reporting a slight improvement in parsing performance. While we have not yet performed a direct comparison with these approaches, one needs to consider the trade-off between our lightweight approach that allows reference extraction, parsing, context extraction and linking to be performed in real time, and more complex approaches that may not allow such real-time processing.

Our work addresses the following limitations of some existing tools:
Poor runtime performance (CERMINE [14], CrossRef pdfextract [1]). We ran CERMINE and pdfextract on our evaluation data set and hardware. CERMINE took 4 days while pdfextract failed to complete. The evaluation shows our tool runs two orders of magnitude faster than the state-of-the-art CERMINE tool.
Unrealistic reference formatting requirements. They require as input exactly one reference string per line (e.g. ParsCit [4]). Extracting candidate citations with exactly one citation per line is challenging, as many tools that extract text from formats such as PDF either preserve hard line breaks, or attempt to wrap the text, which works well for running paragraphs, but tends to glom multiple citations together (e.g. pdftotext). In contrast, our tool can deal with situations where a reference string is split across a number of consecutive lines which are prior to reference parsing reconnected.
Reference position requirements. They require citations to appear in a block towards the end of the document under a heading such as "References" or "Bibliography" and/or require the references to be formatted with hanging indents (e.g. ParsCit, pdfextract). In contrast, our tool assumes that references can appear anywhere in the document body, such as in footnotes. The lower reliance on document structure makes the tool also applicable to non-academic documents.

6 Future Work

There is a number of ways in which we can improve and apply the tool in the future. First we would like to implement more sophisticated logic for determining the citation context boundaries. At the moment this is only based on a fixed size character window. One option is to detect semantically coherent segments by applying the Text Tiling algorithm [16], using only the segment in which the citation occurs as the citation context. However, as this might significantly increase the runtime, we might want to opt for a more lightweight solution.

The second area of interest is the automatic classification of reasons for citation, for example, to the categories specified by Garfield [8] as listed at the beginning of the paper or Teufel [17]. Such work would also be closely related to the identification of influential citations [18]. This has the potential to improve the browsing capabilities of digital libraries and to be used as a feature in the development of new research evaluation metrics/scientometrics. Another strand of work constitutes the application of the citation context extraction tool to effectively construct a sentence/paragraph level co-citation matrix. As demonstrated in [19], such co-citation information could be used as a valuable feature in recommender systems.

7 Conclusions

We have successfully applied CRFs to address two problems: real-time extraction of bibliographic reference strings, from anywhere within the text of an article, and splitting those strings into structured queries to a large digital library of research papers. This approach is article-format and domain agnostic and can potentially be modified for any digital library. We have applied our method to an existing citation network to extract citation contexts and links, so that researchers can read more easily what others say about a given article.

References

1. Khabsa, M., Giles, C.L.: The number of scholarly documents on the public web. PLOS ONE **9**(5), 1–6 (2014)
2. Björk, B., Roos, A., Lauri, M.: Scientific journal publishing: yearly volume and open access availability. Inf. Res. **14**(1) (2009)
3. Bornmann, L., Mutz, R.: Growth rates of modern science: A bibliometric analysis. CoRR abs/1402.4578 (2014)
4. Hristakeva, M., Kershaw, D., Rossetti, M., Knoth, P., Pettit, B., Vargas, S., Jack, K.: Building recommender systems for scholarly information. In: 1st International Workshop on Scholarly Web Mining (SWM), International Conference on Web Search and Data Mining (2017)
5. Knoth, P., Zdráhal, Z.: CORE: three access levels to underpin open access. D-Lib Mag. **18**(11/12) (2012)
6. Knoth, P., Anastasiou, L., Charalampous, A., Cancellieri, M., Pearce, S., Pontika, N., Bayer, V.: Towards effective research recommender systems for repositories. In: Proceedings of Open Repositories 2017, Open Repositories (2017)
7. Pride, D., Knoth, P.: Incidental or influential? - challenges in automatically detecting citation importance using publication full texts. In: 21st International Conference on Theory and Practice of Digital Libraries (TPDL) (2017)
8. Garfield, E., et al.: Citation analysis as a tool in journal evaluation. American Association for the Advancement of Science (1972)

9. Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the 18th International Conference on Machine Learning, pp. 282–289. Morgan Kaufmann, San Francisco (2001)
10. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. SIGKDD Explor. Newsl. **11**(1), 10–18 (2009)
11. Kudo, T.: CRF++: Yet another CRF toolkit. Software (2005), https://taku910.github.io/crfpp/
12. Tkaczyk, D., Szostek, P., Dendek, P.J., Fedoryszak, M., Bolikowski, L.: CERMINE - automatic extraction of metadata and references from scientific literature. In: 11th IAPR International Workshop on Document Analysis Systems, DAS 2014, Tours, France, 7–10 April 2014, pp. 217–221 (2014)
13. Councill, I.G., Giles, C.L., Kan, M.: Parscit: an open-source CRF reference string parsing package. In: Proceedings of the International Conference on Language Resources and Evaluation, LREC 2008, 26 May–1 June. European Language Resources Association, Marrakech, June 2008
14. Luong, M.T., Nguyen, T.D., Kan, M.Y.: Logical structure recovery in scholarly articles with rich document features. IJDLS **1**(4), 1–23 (2010)
15. Kern, R., Klampfl, S.: Extraction of references using layout and formatting information from scientific articles. D-Lib Mag. **19**(9/10) (2013)
16. Hearst, M.A.: Texttiling: segmenting text into multi-paragraph subtopic passages. Computat. Linguist. **23**(1), 33–64 (1997)
17. Teufel, S., Siddharthan, A., Tidhar, D.: Automatic classification of citation function. In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, EMNLP 2006, pp. 103–110. Association for Computational Linguistics, Stroudsburg (2006)
18. Valenzuela, M., Ha, V., Etzioni, O.: Identifying meaningful citations. In: AAAI Workshops (2015)
19. Gipp, B., Beel, J.: Citation Proximity Analysis (CPA) - a new approach for identifying related work based on co-citation analysis. In: Larsen, B., Leta, J. (eds.) Proceedings of the 12th International Conference on Scientometrics and Informetrics (ISSI 2009), vol. 2. International Society for Scientometrics and Informetrics, Rio de Janeiro, July 2009. ISSN 2175–1935