Evidence-Based Information Extraction for High Accuracy Citation and Author Name Identification

Brett Powley and Robert Dale
Centre for Language Technology
Macquarie University, NSW 2109, Australia
{bpowley,rdale}@ics.mq.edu.au

Abstract

Citations play an essential role in navigating academic literature and following chains of evidence in research. With the growing availability of large digital archives of scientific papers, the automated extraction and analysis of citations is becoming increasingly relevant. However, existing approaches to citation extraction still fall short of the high accuracy required to build more sophisticated and reliable tools for citation analysis and corpus navigation. In this paper, we present techniques for high accuracy extraction of citations and references from academic papers. By collecting multiple sources of evidence about entities from documents, and integrating citation extraction, reference segmentation, and citation–reference matching, we are able to significantly improve performance in subtasks including citation identification, author named entity recognition, and citation–reference matching. Applying our algorithm to previously-unseen documents, we demonstrate high F-measure performance of 0.980 for citation extraction, 0.983 for author named entity recognition, and 0.948 for citation–reference matching.

1 Introduction

Citations are a defining feature of academic literature. Writers cite other researchers' works which are related in some way to their their own work or to their discussion. Since Garfield (1965) first proposed the automatic production of citation indexes, the extraction and analysis of citations from collections of academic papers have been of interest to researchers. Two relatively recent developments make the accurate extraction of citations of particular relevance. Firstly, the increasing availability of digital archives of academic papers provides an accessible source of data for citation analysis, and also has led to increasing demand for better tools for navigating these archives (as the popularity of prototype services such as CiteSeer and Google Scholar shows). As a citation by definition implies a meaningful link between two documents, citations provide a natural starting point for the navigation of these resources. The ability to determine, between two documents, not only the existence of a link but also the individual citations and their context raises the possibility of intelligent navigation based on the reason for the citations. Secondly, research impact is increasingly used as an indicator of academic performance, and, correspondingly, citation counts are increasingly used as a proxy measure of research impact. However, to have confidence in any such measure, we need to know that the algorithm used for extraction of the citations on which such counts are based performs with high accuracy. Our focus in this paper is on development of such an algorithm.

For clarity, we use the terms *citation* and *reference* more precisely than in some other literature. A *reference* appears in a list of works at the end of a document, and provides full

bibliographic information about a cited work. A *citation* is a mention of a work in the body of the text, and includes enough information (typically, an author–year pair or an alphanumeric key) to uniquely identify the work in the list of references.

1.1 Problem scope

Our aim in this work is to develop techniques to extract from a given academic paper a list of citations and, for each constituent citation, the corresponding reference in the reference list; and we want to do this with high accuracy. More precisely, we want to (a) find each instance of a citation in the body of the paper; (b) parse this into a set of author names and years; and (c) find the segment of text from the references which contains the corresponding reference.

Because the references section alone provides sufficient information to link a document with those which it cites, a number of existing works focus on parsing the references section alone (see Besagni et al. (2003); Ding et al. (1999); Wellner et al. (2004) for examples), ignoring the citations in the body text. However, our finding is that integrating extraction of citations with reference parsing provides evidence that allows us to parse references, and in particular to extract author names from references, with much higher accuracy. Additionally, the extracted citations and their context provide useful information which can be used to display and analyse the citation function (see Teufel et al. (2006) for a recent example of such an application).

1.2 Citation styles

We introduce some terminology to describe the variety of citation styles that are encountered in academic literature. *Formal* citations are constructions used to explicitly identify a citation. Formal citations may be subdivided into *textual* citations, which use an author–year pair to uniquely identify an entry in the reference list; and *indexed citations*, which use numbers or other abbreviations to refer to an entry in the reference list. *Informal* citations are sentence constructions which refer to a citation without using a formal construction.

Textual citations may be subdivided into two general styles. A *syntactic* citation forms a syntactic part of the sentence which contains it; removing the citation from the sentence will damage the syntax of the sentence. More precisely, the author name or names form a syntactic part of the sentence; and the year is typically enclosed in parentheses. Some examples of textual citations (in **bold face**) are:

(1) **Levin (1993)** provides a classification of over 3000 verbs according to their participation in alternations involving NP and PP constituents.

(2) **Grosz and Sidner (1986)**, in their tripartite model of discourse structure, classify cue phrases based on the changes they signal to the attentional and intentional states.

(3) A more mixed view on the matter is taken by **Marcus et al. (1993)**.

(4) It is important to point out that **Levin**'s **(1993)** classification is not intended as an exhaustive description of English verbs, their meanings, and their likelihood.

(5) Allowing for a definition of 'realizes' that makes the CB behave more like **Sidner**'s Discourse Focus **(1979)** leads to a very significant reduction in the number of violations of Constraint 1.

(6) Our NLP component is implemented using a CDG parser (Harper and Helzerman, 1995; **Maruyama, 1990a**; **Maruyama, 1990b**) because of its power and flexibility.

Syntactic citations may contain one or more author names, as in examples (1) and (2), and may use a construction such as *et al* in place of a full author list, as in example (3). There may be tokens between the author name and year: genitives (example (4)) are very common, but more extended series of tokens (example (5)) occur too. There also exist cases where there are multiple works with the same author and year, in which an additional letter is used to uniquely identify the reference, as in example (6).
A *parenthetical* citation places the author name and year inside parentheses (or brackets or some other delimiters), and can be removed from the sentence without damaging the syntax of the sentence. An example of a sentence containing parenthetical citations is as follows:

(7) Two current approaches to English verb classifications are WordNet **(Miller et al., 1990)** and Levin classes **(Levin, 1993)**.

Note that the distinction between syntactic and parenthetical citations is based on the syntax of the sentence, rather than the presence or absence of parentheses: we sometimes find cases where syntactic citations are placed entirely within parentheses, as in the following example:

(8) The theoretical foundation has been established in extensive work on semantic verb classes such as (**Levin, 1993**) for English and (**Vázquez et al., 2000**) for Spanish.

These citations are syntactic, not parenthetical, since removing them would damage the syntax of the containing sentence.

An *indexed* citation uses a unique key to refer to a similarly-keyed reference in the reference list. The key is typically a number or an alphanumeric code, often derived from the author name and year. Some examples of indexed citations are:

(9) With the discovery of the quantum Hawking radiation **[2]**, it became clear that the analogy is in fact an identity.

(10) The framework of attribute grammars has been utilized for the development of compiler construction [**ASU86**, **Far84**, **RM89**, **KHZ82**], editing environments [**Rep84**, **HT85**] and program transformation[**Joh87**, **CDPR99**].

(11) Hawking [**11**] has argued that under certain conditions in this set K when extended in the past direction.

(12) Attribute grammars were introduced by Knuth [**Knu68**, **Knu71**] as a way of describing semantics of context-free languages.

(13) The bitonic sorting network is discussed in the algorithms collection of [**Knu73**] and several textbooks.

Indexed citations often have the same property as parenthetical citations, that they can be removed from a sentence without damaging the sentence's syntax, as in examples (9) and (10). In some cases (examples (11) and (12)) they are combined with author names in a similar fashion to that found in textual syntactic citations (although where a textual syntactic citation requires both the author and the year to uniquely identify the reference, an indexed citation requires only the key). Occasionally, they are used as syntactic citations, as in example (13).

An *informal* citation refers to another work, but without all the information required to uniquely identify the reference; generally, the additional required information is implied by an earlier citation to the same work. Examples of informal citations are as follows:

(14) **Levin** groups verbs based on an analysis of their syntactic properties, especially their ability to be expressed in diathesis alternations.

(15) **Her** approach reflects the assumption that the syntactic behavior of a verb is determined in large part by its meaning.

Informal citations use either the author name alone, as in example (14), or a pronoun, as in example (15), to refer to an earlier citation; they may appear in documents using either textual or indexed citations.

For the present work, we are interested in textual citations only: these form the dominant type of citation in our corpus; they are more relevant to our related citation analysis work; and in many ways they present the more difficult case, so the work involved in solving the problem of extracting textual citations is a superset of that required to work on indexed citations.

2 Related work

The core problems in information extraction for citation analysis are (a) the extraction and segmentation of reference data; (b) the extraction and resolution of citations; and (c) the extraction of document metadata. There have been several approaches to these problems, although citation extraction has not been addressed as widely as the others: it is often assumed that the reference list will provide equivalent information to the list of citations.

2.1 Citation extraction

Bergmark et al. (2001) report on heuristics for extracting citations (which they call 'contexts' and 'reference anchors') from ACM papers, reporting precision of 0.53, based on randomly selected papers. These papers exclusively use numbered citation keys (e.g. *[1]*) rather than the textual keys which we aim to extract. Bergmark (2000) reports in more detail on extracting information from digital library papers, including citations in a variety of formats. She does not report results for individual extraction tasks, but reports 86.1% 'average accuracy', for the number of 'elements' correctly extracted from each document; 'elements' include the title, author, year of publication, references, and citations. The widely-used CiteSeer system developed by Giles et al. (1998) attempts to extract bibliographic information from references as well as the citation context (i.e. words surrounding the citation). They report being able to extract authors from references 82.1% of the time. Sarawagi et al. (2003) use regular-expression-based heuristics to extract citations and document metadata from a collection of LaTeX documents, by simply parsing the source; they do not report detailed results, but the broader applicability of this work is doubtful since in most collections the LaTeX source for papers is rarely available; extracting information from natural language text is considerably more challenging than extracting explicitly tagged text from LaTeX source code. More recently, Councill et al. (2005) report on using machine learning to extract named entities from the acknowledgements section of scientific papers; they report precision of 0.7845 and recall of 0.8955 for name extraction.

2.2 Reference parsing

Reference parsing is the problem of extracting the individual bibliographic fields – author, title, year, publication, and so on – from the references in a document. There have been various approaches to this problem, one sub-task of which (author name extraction) we address in this work. Besagni et al. (2003) use part-of-speech tagging of words in references (from a corpus of

pharmacology journal papers) to segment them, and report 90.2% accuracy in extracting author names.

Wellner et al. (2004) use conditional random fields for reference segmentation and coreference resolution on a collection of references from CiteSeer, reporting segmentation accuracy across all fields of 94.9%. Takasu (2003) employs hidden Markov models and support vector machines for reference segmentation, reporting high accuracy results, but pointing out that their test corpus, comprising papers from a single journal, had extremely consistent formatting. Ding et al. (1999) use a rule-based ('template mining') system for reference segmentation, reporting 95% accuracy in extracting author names.

2.3 Metadata extraction

Berkowitz and Elkhadiri (2004) report on extracting authors and titles from documents; they report recall of 25.96% for exact extraction of author names; for 24.99% of papers they manage to extract either part of the author name(s), or the name(s) plus extra text. Giuffrida et al. (2000) use a knowledge-based system to extract metadata from computer science journal papers, reporting 87% accuracy in extracting author names. Seymore et al. (1999) use hidden Markov models for the same task, reporting 93.2% accuracy for author name extraction from a narrow corpus of computer science research papers.

3 The corpus

The Association for Computational Linguistics Anthology[1] is a digital archive of approximately 10,000 conference and journal papers in computational linguistics. The ACL Anthology was chosen as the primary corpus for this work. The style of papers in this corpus varies, covering conference papers, workshop papers, technical reports, and full journal papers, for a variety of conferences and publications.

Although the reference lists in many of the papers in the Anthology are produced using LaTeX for conferences where LaTeX style files are provided, this is by no means universal; many others are produced using other software (such as Endnote), and a significant number appear to be produced manually. The dominant style of citation is textual, but there are many formatting variants, even when automated software has been used. Therefore, while one might expect fairly consistent formatting across the corpus, there is in fact remarkable variation. This allows us to develop and test more general heuristics for citation extraction than would be required of a corpus with consistent style.

To produce a text corpus for processing, we used an open-source tool[2] to extract text from the PDF sources, and retained only those documents for which the text extraction process succeeded, yielding approximately 6,000 documents. The main reason for failed text extraction for the other 4,000 is a feature of some PDF files: custom font encodings. Some PDF files include an embedded subset of a font used in the document, which comprises a table of codes and the glyphs to render for each code. However, these codes do not necessarily correspond to any standard encoding such as ASCII or UTF-16; when this occurs, it is non-trivial to recover the original text from the file (indeed, none of the PDF text extraction tools which we evaluated were able to do so). The 6,000 successfully processed documents, however, do provide a sufficiently large and varied corpus for our work.

[1] ACL Anthology, available at `http://acl.ldc.upenn.edu/`.

[2] PDFBox, available at `http://www.pdfbox.org/`.

There are several other issues in the extraction of text from PDF files which affect strategies for information extraction. Firstly, the output of the extraction process is an unformatted text stream: the only intact formatting cues are line breaks. Font changes, blank lines, and all other formatting are absent. A second issue arises from the fact that spaces often do not occur in the text stream in PDF files, but rather their presence must be inferred from the positions of surrounding characters. Although the text extraction tool we used generally does a good job of this, there are cases where either spurious spaces are introduced or interword spacing is omitted, making accurate tokenisation of the text more difficult.

The training data set used for development of our citation extraction algorithm included papers randomly selected from across the Anthology, but with a bias towards those from 2000 onwards, since these were 'born digital' rather than scanned, making it possible to isolate OCR errors when evaluating our algorithm's performance. For the experiments in this paper, a subcorpus of documents not involved in the development process was selected, so that we could avoid 'testing on the training data'. This subcorpus comprised 5 papers randomly selected from 3 years of each of 4 collections (Association for Computational Linguistics (ACL) conference proceedings; International Conference on Computational Linguistics (COLING) conference proceedings; ACL workshops; and *Computational Linguistics* journal papers), totalling 60 papers. Book reviews, discussion articles, and other documents without citations were excluded, as were documents where font encoding issues caused text extraction errors.

Individual documents are segmented into header, body, references, and appendix sections using textual cues. The body section of the document is then dehyphenated and segmented into sentences; additional heuristics are used to distinguish section headings common in academic papers from sentences. The data to be processed for each document then comprises a list of sentences from the body, and a segment of text containing the references section.

4 Techniques

4.1 An integrated approach

Our approach to reference segmentation, citation extraction, and citation–reference matching is an integrated one, based on the idea that each of these tasks can be improved by evidence derived from the others.

Beginning with the first of these tasks, there are several problems which make extracting information from references non-trivial. Punctuation, which would normally be considered as the main field separator, is highly semantically overloaded: for example, full stops are used variously after author initials, after abbreviations, and to separate fields in the reference. Formatting cues (such as the use of italics for the title) are absent from the text stream delivered from the PDF file, as are line breaks separating references. Attempting to parse references into component fields in isolation is therefore difficult, particularly if we want to do so in a general fashion which makes no assumptions, or limited assumptions, about the order or formatting of fields. However, if we consider references in their context in the document, there is additional evidence which can assist us in tagging words in the reference. In particular, each reference ought to have at least one corresponding citation in the body text; this citation will contain the author names. While the citations and references contain instances of the same entities (author names), they do so in different textual contexts; in the citations they are part of sentences, while in the references they are part of bibliographic records. The distinct instances of these entities can be employed as mutual constraints when recognising either citations or references. By locating corresponding citations for a reference, we have compelling evidence for the named

⟨citation-instance⟩ ::= ⟨author-list⟩ ⟨year-list⟩
⟨author-list⟩ ::= { ⟨author-surname⟩ ⟨author-separator⟩* }+ [**et al**] [**'s**]
⟨author-separator⟩ ::= **,** | **;** | **and**
⟨year-list⟩ ::= [**(**] { ⟨year⟩ ⟨year-separator⟩* }+ [**)**]
⟨year-separator⟩ ::= **,** | **;**
⟨year⟩ ::= { **1900** | **1901** | **1902** | . . . } [**a** | **b** | **c** | . . .]

Figure 1: Simplified grammar for a textual citation instance

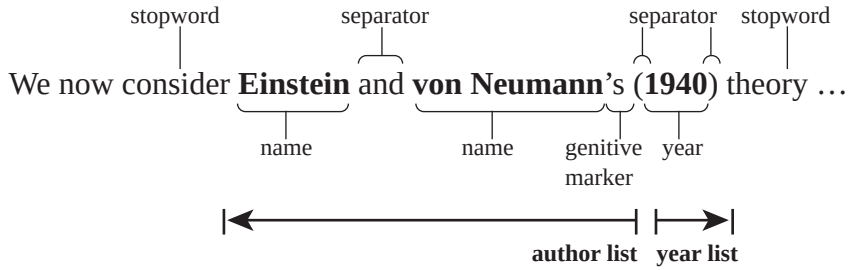

Figure 2: Extraction of citation information from a sentence

entity recognition task as applied to references. Similarly, extracting textual citations requires the ability to recognise surnames in the body of the text; evidence from the references section about what words represent surnames can be used to perform this task more accurately.

4.2 Citation extraction

The citation extraction algorithm works at the sentence level to isolate and tag citations. We begin with the observation that textual citations are anchored around years. In an earlier experiment, we found that we could reliably identify candidate sentences containing citations by looking for years alone: on a random selection of papers from 2000–2005 covering 294 citation instances, this simple heuristic gave a recall of 0.99.

Our first step is therefore to search each sentence for a candidate year token (a 'year' for this purpose being a 4-digit number between 1900 and the current year, potentially with a single character appended to it). If we find such a token, our task is then to determine whether it forms part of a citation, and if it does, to extract the author names that accompany it. A simplified version of the grammar for a citation on which our algorithm is based is shown in Figure 1. In general, we may say that a textual citation comprises one or more authors followed by one or more years; in practice, the variety of constructions which a writer might use to format a citation means that it is somewhat more complicated.

Writers often use a list of years as shorthand for citing multiple papers by the same author: for example *Smith (1999; 2000)* (which in fact represents citations of two separate works). Given the candidate year, we therefore first search backwards and forwards to isolate a list of years. We then search backwards from the year to find the list of authors, skipping over punctuation and separators, and stopping when we encounter a non-surname word or a stopword; an illustration of this process is shown in Figure 2. If no author names are found, we conclude

that the candidate year was a number unrelated to a citation. We also treat a small number of temporal prepositions which commonly appear before years as stopwords, concluding that the candidate year is not a citation if preceded by one of these (*in*, *since*, *during*, *until*, and *before*). Otherwise, having found a list of authors, we normalise the citation instance into a list of citations each consisting of a list of authors and a single year. We also record whether the citation contains an *et al* string (indicating that the list of authors is not comprehensive) or whether it ends with a genitive (e.g. *Powley's (2006) citation extraction algorithm*). The key problem in citation extraction is accurate identification of author surnames, the algorithm for which is described in the following section.

Test data set	
Number of documents	60
Candidate sentences (containing years)	1743
Citing sentences (containing citations)	1620
Citation instances	2406
Citation extraction	
Precision	0.9992
Recall	0.9612
F-measure	0.9798

Table 1: Evidence-based citation extraction

To evaluate the performance of the citation extraction algorithm, the citation extractor was run on the test corpus, and output produced including all candidate sentences (i.e. those with years), identified citation instances, and citation instances parsed into individual author names and years. Results were hand annotated to identify (a) token sequences wrongly tagged as citations; (b) correctly identified but incorrectly segmented citations; and (c) missed citations. The results are shown in Table 1. For the precision and recall scores, a citation was counted as successfully extracted only if it was correctly located and the list of author names and years correctly extracted.

The high performance of the citation extraction algorithm is largely due to the high performance of the named entity recognition algorithm on which it relies, which we describe in the following section. Error analysis of the missed citations shows that 27% were missed due to OCR errors in a single document; the remainder were due to text extraction errors (missing or extra spaces in the text stream) or deficiencies in the document itself (for example, works cited but missing from the references list).

4.3 Named entity recognition

The citation extraction algorithm relies on the ability to identify author surnames. In particular, we require the ability to determine whether a word preceding a candidate year is a surname (and therefore that the year forms part of a citation); and the ability to distinguish surnames from other words to determine where the list of author names stops (or, more precisely, begins).

4.3.1 An evidence-based algorithm

Our named entity recognition algorithm is based on the observation that any author name in the body of the document ought also to appear in the references section. A candidate surname in a

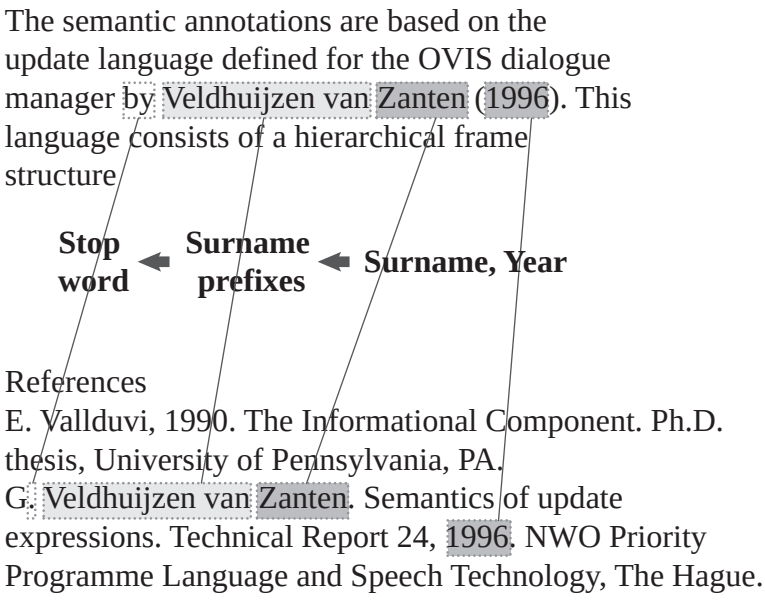

Figure 3: Compound named entity recognition

citation is a capitalised token preceding a year (or another surname); the simplest approach is to search the references section for the same token, and if it appears, assume that the candidate token is a surname. However, surnames are not the only capitalised words that appear in the references section, so treating the entire references section simply as an author name gazetteer in this fashion will generate false positives. To be more certain that the token that we have found in the references section is an author name, we search for both the candidate surname and the year from the reference appearing within a 5-line window of each other. (We observed that the distance between the author names (at the beginning of the reference) and the year (somewhere later in the reference, depending on the reference style) rarely exceeds 5 lines.)

An additional problem that we want our named entity recogniser to be able to handle is that of compound surnames, or surnames which consist of more than a single capitalised word; our experiments suggest that approximately 2% of author names in the Anthology are of this type. Commonly, these comprise a surname and a number of prefixes (often prepositions) from a relatively fixed set: for example, *von Neumann*, *van den Bosch*, *Uit den Boogaart*, *Della Pietra*, and *Al Shalabi*. Our initial approach to this problem was therefore to build a list of surname prefixes, and tag items from this list preceding a capitalised surname as surname prefixes. However, compound surnames which consist of elements from a non-closed set are not uncommon: for example, *Gaustad van Zaanen*, *Villemonte de la Clergerie*, *Tjong Kim Sang*, and *Schulte im Walde*.

Our strategy for detecting the bounds of compound surnames is then to make no assumptions about the set of words which can comprise a surname and its prefixes. Rather, we use evidence from the two (or more) distinct instances of a surname which we have: one in the body of the document as part of the citation, and one in the references section as part of a reference. An example of compound named entity recognition is shown in Figure 3. We start at the capitalised surname word in the body text, and its counterpart in the references text. Moving backwards in the body and references, we compare words, continuing until a non-matching word is found. Matching words are then tagged as part of the surname. In order to test the performance of

Test data set	
Number of documents	60
Author name instances	3531
Multiword author names	73
Named entity recognition	
Precision	0.9988
Recall	0.9678
F-measure	0.9831

Table 2: Alignment-based author named entity recognition

the surname recogniser, we ran our citation extractor on our test corpus to output citations (including the author surname and year), and manually annotated the results; a positive result meant that the algorithm correctly identified all tokens in the author name. Performance for the named entity recognition task is shown in Table 2. For the precision and recall scores, a surname is counted as successfully recognised only if the name and all prefixes (and no additional tokens) are correctly extracted.

The alignment-based algorithm gives extremely good results: the high precision (0.9988) shows that we rarely misidentify a token as an author name, or miss author name prefixes. The high recall indicates that we rarely miss author names; error analysis shows that the main cause of missed names in our test data set was malformed author names, due either to misspelling or errors in the text extraction process.

4.3.2 Baseline algorithm

Given the good performance for the alignment-based algorithm, we decided to compare its performance with a simpler algorithm using only orthographic cues (i.e. capitalisation) and a fixed set of prefixes, and to test our intuition that the more sophisticated algorithm was necessary. We

A	Ai	Ap	Aux	Á	Ak	Ar	Az	A’	Al	As	Af	Am	Au	Ag	An	Aus	Bel	
D	De’	Dell’	Des	D’	De la	Della	Det	Da	De las	Dellas	Di	Dal	De lo	Delle				
Dia	Dalla	De los	Delli	Die	Dallas	Degli	Dello	Do	Dalle	Dei	Dellos	Dos	Das					
Del	Den	Du	De	Delah	Der	E	Ei	Eit	En	Een	Ein	El	Et	Eene	Eine	Els	Ett	
Gl’	Gli	Ha	Hen	Hinar	Hoi	Hai	Het	Hinir	He	Hi	Hinn	Heis	Hin	Hn				
I	Í	Il	Im	Isa	Ka	Ke	L	Le	Li	Lou	L’	Les	Lis	Lu	La	Lh’	Lo	Las	Lhi	Los
Mia	Na	Ni	Ní	Nje	Ny	O	Ó	O’	’O	Oi	Op	Op de						
’S	Si	T	Ta	Ten	To	’T	Te	Ter	Um ā	Um	Un	Une	Uns	Una	Uno	Us		
Van	Van der	Vom	Von zu	Van de	Vel	Von	Van den	Ver	Von der									
Y	Ye	Yn	Yr	Z	Zu	Zum	Zur											

Table 3: MARC 21 List of surname prefixes (Library of Congress, 2002)

implemented a baseline algorithm for author named entity recognition based on the hypothesis that an author name could be identified as a capitalised word preceded by one or more ‘surname prefixes’. The source for the list of prefixes was the Library of Congress cataloguing guidelines for personal names (Library of Congress, 2002); the list is shown in Table 3. This algorithm was tested on a corpus of 1454 conference papers from the ACL Anthology. Since the precision

Test data set	
Number of documents	1454
Author name instances	38998
Multiword author names	771
Author name identification	
Precision	0.92
Recall	1.0
Prefix identification	
Precision	0.36
Recall	0.26

Table 4: Baseline author named entity recognition

of our evidence-based algorithm was extremely high, we decided that we could use the output of the evidence-based algorithm as the gold standard, and compared the results of the baseline algorithm to it. Results are shown in Table 4 for author name identification (correctly tagging a capitalised word as part of a surname) and prefix identification (correctly tagging all prefix words that comprise part of the surname). The perfect recall score is a reflection of the fact that the gold standard data set is produced using capitalisation as a starting point, the same heuristic used by the baseline algorithm. The lower precision score reflects the failure of the baseline algorithm to distinguish capitalised words which are not author names. The poor performance for prefix identification confirms that our intuition was correct: a fixed list of prefixes does not provide satisfactory coverage for the variety of compound author names encountered in our corpus.

4.4 Citation–reference matching

Citation–reference matching involves two tasks: finding the reference in the references section corresponding to each citation; and segmenting the references section into individual references. Recall that the text stream extracted from the original PDF document is unformatted: cues such as font changes and blank lines are absent, so we can rely only on textual cues for the segmentation task. The main textual cue we use for this is the location of author names in the references section. Rather than isolating these two tasks, we integrate the citation–reference matching and reference segmentation, using evidence from citation–reference matching to determine where individual references begin and end.

As we collect citations from the body of the document, we tag the corresponding author name–year pairs in the references section, creating a list of author names and their positions. This first pass through the references section tags author names explicitly corresponding to citations from the body text. There may still, however, be author names which did not occur in the body text because they were the *alii* authors in an *et al* citation, or because they occurred in a reference which wasn't cited. To partially address this problem, we make another pass through the references section, this time tagging all names which appear on the author name list we built on the first pass. This means that all instances of an author's name are tagged, including those not explicitly appearing in a corresponding citation. (This still does not tag names for which there is only one instance in the document; we consider the reasons for this and strategies for

Test data set	
Number of documents	60
Citation instances	2503
Unique citations	1449
References	1595
Citation - reference matching	
Precision	0.9954
Recall	0.9058
F-measure	0.9485
Reference segmentation	
Precision	0.9903
Recall	0.9658
F-measure	0.9779

Table 5: Citation-reference matching and segmentation

dealing with it in the following section.)

When first testing this algorithm, we discovered that author surnames also frequently appear in two other places: as editors of collections, and occasionally, for authors whose names have become synonymous with a technique, in titles of papers (for example, *Collins parser*, *Brill's tagger*). To deal with these cases, we added an additional test based on a count of words after the last candidate author, and also on detection of stop words such as *in* which typically separate the name of the collection from the article details; these provide a reasonable heuristic for determining when we are past the end of the author list in a reference. Names appearing after this point are assumed not to be author names.

We also use the author name list for segmenting the references section into individual references. Since author names invariably occur at the beginning of a reference, the line position of each author name is also a candidate for the beginning of a reference segment, although only an author name from the first line will truly indicate this. A candidate reference segment then consists of the location of the author name (potentially the start of the reference) and the year (some way into the reference). To turn the candidate segment list into a list of actual references annotated with author names, we scan the candidate reference segment list, combining overlapping segments and adding author names from the candidate segments into a list of author names for that segment.

Once this process is complete, we have a list of references, each with a key comprising a list of author names and a year, and a list of extracted citations, which is similarly keyed. Our task is then to match each citation to the corresponding reference. We first deal with citations which do not include an *et al* entry; these ought to list all authors in the reference, and so we match only those for which the citation author list and the reference author list are identical. On a second pass, we match *et al* citations; this time, we match references which contain all the explicitly-named authors in the citation.

To test citation–reference matching and segmentation performance, the citation extractor was run on the test corpus, and for each document, unique instances of citations along with the identified reference listed for manual annotation. Using unique instances of citations gives a better indication of performance (since the same logic is involved in matching, for example,

Powley 2006 to the correct reference whether it is cited once or multiple times in the same document), and also considerably reduced the manual annotation burden in evaluating results. The results are shown in Table 5. In calculating precision and recall scores for citation–reference matching, we counted those instances where a citation was matched to the beginning of the correct reference. For reference segmentation, we counted those instances where the beginning and end of the reference were correctly identified, and where no incorrect segments were added. High precision is again an indication of the effectiveness of the named entity recognition algorithm, and also of the approach of integrating citation extraction and reference segmentation. Recall is not as good: approximately 9% of citations were not matched to a reference. The main cause of citation–reference matching failure was deficiencies in the text, due to either writer error (inconsistent spelling of names or missing references) or text extraction issues (OCR errors or PDF extraction errors).

5 Discussion and future work

The evidence-based approach to named entity recognition, on which our citation extraction and citation–reference matching algorithm is based, performs extremely well. In some respects, this is consistent with our intuition about recognising names: we recognise a word as a name partly because we have seen it elsewhere in a context which tells us that the word represents someone's name; using additional instances of a named entity as evidence provides our algorithm with knowledge roughly analogous to this. The ability to deal with compound surnames is another clear advantage of this approach. While these represent only about 2% of the author names in our test corpus, we could not claim that our citation extraction algorithm performed with high reliability without explicitly handling them. For applications such as citation analysis for measurement of research impact, it would be unacceptable to miss important works merely because the form of an author's name was inconvenient.

For this work, we used only evidence internal to a document for recognising names in that document. There were, however, two cases where this approach did not identify all names. The first case is where there was only one instance of an author name in the references section, and no citation explicitly containing that name, because the corresponding citation contained *et al* rather than an exhaustive name list, and no other papers with the same author were cited. The second case is where a work appears in the reference list but is not cited (perhaps because the writer thought it was relevant reading even though it was not directly discussed in the paper, or because the writer had removed the citation as the paper was edited, but not the corresponding reference). In future work, we plan to collect evidence across the corpus of documents, thereby automatically creating a gazetteer of author names which can then be more broadly employed in the named entity recognition task, and used to identify author names in cases like these. We also see potential for extending the approach of finding multiple instances of an entity to extraction of other information from the document: in particular, the paper title, author names, and affiliations from the header of a paper; and various bibliographic fields in addition to author names from the references list.

We have for the current work endeavoured to use a clean data set so that we could validate our approach without the complication of OCR errors. Nevertheless, our collection still contained a single OCR'ed paper, and still had various textual errors introduced by PDF extraction; for larger-scale corpus analysis, we will need to address this issue in order to maintain high accuracy. While our alignment-based algorithm expects exact matches, we expect that it ought to be easily extended to accommodate approximate matches; indeed, our evidence-based approach ought to make approximate matching more reliable, since we can employ multiple candidate

instances of a name to determine whether an approximate match is likely to be appropriate or not.

Accurate extraction of citations and reference data has some immediate practical applications: two of these are automatically producing bibliographic records (e.g. in BibTeX format) for a document or a corpus, and automatic annotation of documents (e.g. hyperlinking citations to the corresponding reference, and hyperlinking references to a copy of the cited paper). However, we anticipate more sophisticated applications for accurately extracted citation data. In particular, we plan to analyse collections of citing sentences – both from an individual document to cited works, and from citing works to an individual document – in order to determine the semantics of relationships between documents, and provide navigational tools based on those semantics. As an example, while finding all the papers that cite a particular work may be useful, more useful might be knowing more specifically which of those papers use the technique described in that work. We also see potential for using citing sentences and navigating chains of citations to automatically produce summaries (containing facts from a work) and reviews (containing others' use and opinions of a work).

References

Bergmark, D. (2000). Automatic extraction of reference linking information from online documents. Technical Report CSTR2000-1821, Cornell Digital Library Research Group.

Bergmark, D., Phempoonpanich, P., and Zhao, S. (2001). Scraping the ACM digital library. *SIGIR Forum*, 35(2):1–7.

Berkowitz, E. and Elkhadiri, M. R. (2004). Creation of a style independent intelligent autonomous citation indexer to support academic research. In *Proceedings of the the Fifteenth Midwest Artificial Intelligence and Cognitive Science conference MAICS 2004*, pages 68–73.

Besagni, D., Belaid, A., and Benet, N. (2003). A segmentation method for bibliographic references by contextual tagging of fields. In *Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on, Vol., Iss., 3-6 Aug. 2003*, pages 84–88 vol.1.

Councill, I. G., Giles, C. L., Han, H., and Manavoglu, E. (2005). Automatic acknowledgement indexing: Expanding the semantics of contribution in the CiteSeer digital library. In *Proceedings of the Third International Conference on Knowledge Capture*, Banff, Canada.

Ding, Y., Chowdhury, G., and Foo, S. (1999). Template mining for the extraction of citation from digital documents. *Library Trends*, 48(1):181–207.

Garfield, E. (1965). Can citation indexing be automated? In Stevens, M., editor, *Statistical Association Methods for Mechanical Documentation, Symposium Proceedings*, National Bureau of Standards Miscellaneous Publication 269, pages 189–142. National Bureau of Standards.

Giles, C. L., Bollacker, K., and Lawrence, S. (1998). CiteSeer: An automatic citation indexing system. In Witten, I., Akscyn, R., and Shipman III, F. M., editors, *Digital Libraries 98 - The Third ACM Conference on Digital Libraries*, pages 89–98, Pittsburgh, PA. ACM Press.

Giuffrida, G., Shek, E. C., and Yang, J. (2000). Knowledge-based metadata extraction from PostScript files. In *DL '00: Proceedings of the fifth ACM conference on Digital libraries*, pages 77–84. ACM Press.

Library of Congress (2002). LC guidelines for usage - X00's alphabetic list of surname prefixes. In *MARC 21 Format for Authority Data*. Library of Congress, Washington, DC.

Sarawagi, S., Vydiswaran, V. G. V., Srinivasan, S., and Bhudhia, K. (2003). Resolving citations in a paper repository. *SIGKDD Explorations Newsletter*, 5(2):156–157.

Seymore, K., McCallum, A., and Rosenfeld, R. (1999). Learning hidden markov model structure for information extraction. In *Proceedings of the AAAI 99 Workshop on Machine Learning for Information Extraction*, pages 37–42.

Takasu, A. (2003). Bibliographic attribute extraction from erroneous references based on a statistical model. In *Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries*, pages 49–60.

Teufel, S., Siddharthan, A., and Tidhar, D. (2006). Automatic classification of citation function. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 103–110, Sydney, Australia. ACL, Association for Computational Linguistics.

Wellner, B., McCallum, A., Peng, F., and Hay, M. (2004). An integrated, conditional model of information extraction and coreference with application to citation matching. In *UAI 2004: Proceedings of the 20th conference on uncertainty in artificial intelligence*, pages 593–601, Banff, Canada.