Expert Systems with Applications 42 (2015) 6391–6407

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

LEXA: Building knowledge bases for automatic legal citation classification

Filippo Galgani [*], Paul Compton, Achim Hoffmann

School of Computer Science and Engineering, University of New South Wales, Australia

ARTICLE INFO

Article history:
Available online 28 April 2015

Keywords:
Knowledge acquisition
Natural language processing
Citation analysis
Legal documents

ABSTRACT

This paper presents a new approach to building legal citation classification systems. Our approach is based on Ripple-down Rules (RDR), an efficient knowledge acquisition methodology. The main contributions of the paper (over existing expert-systems approaches) are extensions to the traditional RDR approach introducing new automatic methods to assist in the creation of rules: using the available dataset to provide performance estimates and relevant examples, automatically suggesting and validating synonyms, re-using exceptions in different portions of the knowledge base. We compare our system LEXA with baseline machine learning techniques. LEXA obtains better results both in clean and noisy subsets of our corpus. Compared to machine learning approaches, LEXA also has other advantages such as supporting continuous extension of the rule base, and the opportunity to proceed without an annotated data set and to validate class labels while building rules.

1. Introduction

Legal citations are crucial for the operation of the legal system where case law is used. The citations made in court decisions indicate how other cases are related to the current case by the respective judges. The treatment received by cited cases forms an important component in understanding the law and interpreting it appropriately for new cases. Given the very large number of court decisions being made every day, legal professionals need assistance in finding relevant court decisions and relevant citations. One vital task is identifying how the court in subsequent cases has considered the primary case judicially. This paper describes our new approach to build legal citation classification systems. We focus in particular on identifying citations which were "**Distinguished**" from the current case. The class Distinguished is the most common negative history marker found, and was the focus of the work of Conrad and Dabney (2001), the only previous attempt at legal citation classification, who claim that "*from a legal research standpoint it is more important to have overturned, weakened or questioned opinions revealed rather than have reaffirmed or respected decisions featured*". Identifying instances of a minority class is an important task in several other text processing applications such as in the finance domain (e.g. report and news monitoring) or in the intelligence sector.

Citation classification is often tackled with supervised machine learning algorithms. The alternative is the use of expert systems, usually composed of manually created rules. In these kinds of system human intuition guides the learning, with domain experts encoding domain knowledge directly in rules. One reason for the persistent difficulty with building more sophisticated NLP systems is the extraordinary variety in which certain content may be presented in natural language. Capturing the variety of possible natural language patterns appears to require substantial task-specific knowledge, and it is still not clear how to best capture this domain knowledge. While manual approaches can be tedious, automatic methods require large amounts of data and the results may still be inferior.

We propose to use Ripple-down Rules, an incremental knowledge acquisition (KA) methodology, to automatically identify Distinguished citations. The main advantage of RDR over traditional expert systems is that it allows the continuous extension of the knowledge base without increasing the effort or difficulty for the humans involved.

The first contribution of this paper is to show that an RDR-based approach can successfully tackle a citation classification task in a complex domain such as law. To do so, our approach extends RDR to use the available annotated dataset to provide additional support during rule acquisition. This is done providing performance estimates and selecting relevant examples to better tune rules, suggesting validated synonyms and other generalizations, and supporting the re-use of exception rules.

[*] Corresponding author.
E-mail addresses: galganif@cse.unsw.edu.au (F. Galgani), compton@cse.unsw.edu.au (P. Compton), achim@cse.unsw.edu.au (A. Hoffmann).

http://dx.doi.org/10.1016/j.eswa.2015.04.022
0957-4174/Crown Copyright © 2015 Published by Elsevier Ltd. All rights reserved.

The second contribution of this paper is to compare the performance of our KA approach with machine learning approaches. We show that our system LEXA outperforms machine learning, and its performance degrades more gracefully when dealing with noisy data. Other advantages of our approach are that it does not require pre-classified citations and it includes the possibility of easily extending the knowledge base, and of resolving class ambiguities at the same time as building the system, as the expert looks at individual cases.

The rest of the paper is organized as follows: Section 2 introduces legal citations. In Section 3 we discuss existing approaches to citation classification, while in Section 4 we introduce Ripple-down Rules, the knowledge acquisition methodology which we used as basis for our system. Section 5 describes our corpus of legal citations. Section 6 describes our KA framework LEXA (Legal tEXt Analyzer), and Section 7 presents the results on citation classification, and a comparison with machine learning approaches. We then introduce two possible modifications to the framework: automatic synonym expansion (Section 8) and automatic exception suggestion (Section 9), which can assist in the automatic creation of rules. Finally Section 10 draws conclusions and presents directions for future work.

2. Legal citations

Information overload is a significant problem in legal research due to the large amount of legal material existing in textual form. Natural language processing applications are needed to cope with constantly increasing information. In common law countries (such as Australia, the UK and the USA) the legal principles underpinning the law are often distributed across multiple legislative and case report documents. In addition legal principles evolve over time and therefore the temporal aspects of the documents must be considered, such as what judgements have been overturned and what new decisions have been made in relation to the legal principles (Murphy, Steele, & Shen, 2008). Judges cite cases that they rely on when making their decisions, as past decisions have a binding effect on any following decision in a process that is known as *stare decisis* (Moens, 2007). Court decisions or cases can be instructive as they introduce a new principle or rule, modify or interpret an existing principle or rule, or settle a question upon which the law is doubtful.

In case-law systems citations are a conspicuous feature of most judicial opinions: examining citations tells us how the law we are relying on has been interpreted. Even though a case may be good law, its holding may be affected by a subsequent interpretation by the same court or different courts, which can affect its precedential value. For this reason it is vital for law professionals to know whether the decision has received positive, negative or neutral treatment in subsequent judgements. It is therefore a common practice for curated databases (for example WestLaw KeyCite or LexisNexis CaseBase), to have experts attaching a label to each citation specifying how the case was considered, for example whether it has been "applied","not followed", "distinguished", etc.

An example of in-text citation is:

> *Eden refers to Hughes Aircraft Systems International v Airservices Australia (1997) 76 FCR 151 (Hughes) and Cubic Transportation Systems Inc v State of New South Wales [2002] NSWSC 656 (Cubic Transportation). The facts of Hughes were vastly different from those of the present case. In Hughes, there were two competing tenderers, the applicant in the proceeding being the unsuccessful one. Finn J held that in the circumstances of the case, there was an implied contract that might be called a "process contract", according to which Airservices Australia incurred implied contractual obligations in favor of the tenderers, including an implied obligation to conduct its evaluation fairly and in a manner that would ensure equal opportunity as between them. Cubic Transportation also concerned competing tenderers.*

Table 1
Distribution of citation classes. To build our system we used only the classes in bold.

Cited	9346	Affirmed	91	Disapproved	8
Referred to	3017	Quoted	87	Noted	7
Applied	**1803**	Approved	61	Relied on	4
Followed	**1759**	Not Followed	57	Doubted	3
Considered	1339	Reversed	20	Compared	2
Discussed	706	Ref to	15	Adopted	2
Distinguished	**460**	Explained	10	Referred	2
Related	94	Questioned	9	Overruled	2

In this case *Hughes* is classified as "Distinguished", as differences with the current case are emphasized, while *Cubic Transportation* is classified simply as "Referred to".

The aim of our system is generating these citation labels automatically. Negative treatment is less frequent than positive or neutral, and it is considered more important from a legal research point of view: "*researchers analyzing case law opinions regard such negative indirect history citations as indispensable.*" (Conrad & Dabney, 2001). Among negative markers, the Distinguished label is the most common (see Table 1). Given the importance of identifying negative treatments we present a system that automatically finds Distinguished citations, with the aim of performing this task automatically with a performance similar to human experts. In this paper we report results for the Distinguished class, but our system could easily be modified to handle other classes. The automatic analysis of citations, besides its intrinsic utility, could also support other NLP tasks in this domain, such as automatic summarization (Galgani, Compton, & Hoffmann, 2015).

3. Related works

The problem of information overload in the legal domain has stimulated the development of several natural language processing applications for legal text, for example automatic summarization (Hachey & Grover, 2006; Galgani, Compton, & Hoffmann, 2012; Yousfi-Monod, Farzindar, & Lapalme, 2010), retrieval (Moens, 2001), information extraction (Moens & Angheluta, 2003; Palau & Moens, 2009) and automatic translation (Farzindar & Lapalme, 2009; Gotti, Farzindar, Lapalme, & Macklovitch, 2008).

However in the automatic analysis of legal citations very little has been done. The only work on automatic classification of legal citations that we are aware of is (Conrad & Dabney, 2001), on identifying citations of the type "Distinguished", in order to support professional editors who manually identify these citations, the system ranks the candidate distinguished citations which are then manually inspected. The system is composed of 20 hand coded high level rules, which recognize different kinds of patterns, formed by lexical clues supplied both by domain experts and by inspection of multiple cases from a training corpus. These patterns allow different forms of words and the presence of synonyms and gaps, but a formal rule language is not described. When testing the rule base on an unseen test set, the authors showed a precision of 9.15% and a recall of 59.09%, which illustrates the challenge of the creation of rules with high precision.

Zhang and Koppaka (2007) differentiate citations according to the legal principle discussed (citations to the same document will focus on different issues of the case) in order to build a network of legal citations. Citations are organized in sub-networks, each focusing on one specific legal issue, thus reducing the number of cases that a researcher (usually interested in a particular issue) has to go through. The problem of automatically finding referents in the text (citation contexts) is usually targeted using regular expressions and heuristic rules (Mowbray, Chung, & Greenleaf, 2009). Another related line of research is that examining citations to legislation. Several parsers have been developed to automatically recognize, normalize and deduplicate citation in and between laws and legislation, for Dutch (de Maat, Winkels, & van Engers, 2006; van Opijnen, 2010), Italian (Palmirani, Brighi, & Massini, 2003) and Spanish (Martínez-González, de la Fuente, & Vicente, 2005) law. In our corpus most the references were already identified and marked with html links.

The literature on automatic classification of citations in other genres is more extensive, particularly for citations in scientific papers. In the system of Nanba and Okumura (1999) cue phrases are used to classify citations from scientific articles into three categories: *Basis*, *Comparison or Contrast*, and *Other*. In their work, a list of short cue phrases is selected manually and encoded in 160 rules, which achieve an average accuracy of 83% on a test set. Pham and Hoffmann (2003) aim at reducing the time spent on manually listing cue phrases, using an efficient knowledge acquisition framework, based on Ripple-down Rules. The rules are expressed as simple patterns consisting of an arbitrary number of words and gaps between them. The system is able to discern between four citation types (*Basis*, *Support*, *Limitation*, *Comparison*) and the accuracy outperforms the system of Namba and Okomura.

Another system that automatically performs citation classification is described in Teufel, Siddharthan, and Tidhar (2006). To recognize meta-discourse features the authors propose patterns based on two mechanisms: (i) a finite grammar over strings with Part-of-Speech (POS) place holders and classes of equivalence for similar words, (ii) POS-based recognition of agents and actions. These features are used in conjunction with location, verb tense and voice to train a statistical classifier, which is able to recognize four classes of citation and obtains an accuracy of 79%. More recently Xu, Martin, and Mahidadia (2013) presented an approach to classify citations into three classes: functional, perfunctory or ambiguous. Their work focuses on experimenting with different types of features, such as local textual (cue patterns) and extra-textual (stylistic and positional) features and network-based structural features, which attempt to capture relationships between authors and between papers using different centrality measures.

Related to citation classification is work on sentiment analysis and opinion mining (Pang & Lee, 2008), which has been applied to citation analysis, for example to determine whether an author is citing a piece of work as supporting evidence or as research that he or she dismisses (Piao, Ananiadou, Tsuruoka, Sasaki, & McNaught, 2007). The analysis of the sentiment of a citation indicates the polarity of the citation as related to agreement or disagreement. Athar (2011) addressed the problem of identifying positive and negative sentiment polarity in citations to scientific papers (with the additional category of *objective*), exploring a range of features such as different length n-grams, POS tags, dependency relations and two different polarity lexicons. Athar and Teufel (2012) tried different fixed context sizes to classify scientific sentiment towards a target paper. Ding et al. (2014) introduce content-based citation analysis, to address a citation's value by interpreting each one based on its context at both the syntactic and semantic levels.

Other systems have analysed citation context in relation to different tasks. In 2004 Nakov, Schwartz, and Hearst (2004) pointed out the possibility of using citation contexts directly for text summarization, as they provide information on the important facts contained in the paper. An application of the idea can be found in Qazvinian and Radev (2008) and Qazvinian, Radev, and Ozgur (2010), where a summary is created extracting a subset of the sentences that constitute the citation context. Mohammad et al. (2009) apply this approach to multi-document summarization, building up on the claim by Elkiss et al. (2008) about the difference in information given by the abstract and the citation summary of a paper. Mei and Zhai (2008) use citation data to summarize the impact of a research paper, Zhu, Turney, Lemire, and Vellino (2015) to measure academic influence and Xu, Martin, and Mahidadia (2014) to identify and characterize scientific specialisations, Citation contexts have been used also to improve information retrieval from scientific papers (Ritchie, Teufel, & Robertson, 2006; Ritchie, Robertson, & Teufel, 2008). We think that, in a similar way, citation analysis in the legal domain can support tasks such as summarization and retrieval.

4. Basis of our approach

Our approach is based on incremental knowledge acquisition (KA), in particular the Ripple-down Rules (RDR) methodology (Compton & Jansen, 1990; Edwards, Compton, Malor, Srinivasan, & Lazarus, 1993). While this methodology has been presented in depth in other publications, we outline it here for those readers unfamiliar with it.

In RDR, rules are created by domain experts without a knowledge engineer, the knowledge base (KB) is built with incremental refinements from scratch, while the system is running: the domain expert monitors the system and whenever it performs incorrectly he signals the error and provides as a correction a rule based on the case which generated the error, which is added to the knowledge base. RDR is essentially an error-driven KA approach; the incremental refinement of the KB is achieved by patching the errors it makes, in the form of an exception rule structure.

A single classification RDR (SCRDR) is a binary tree (see Fig. 2) and associated with each node is a *rule* (a condition and a conclusion). *Cases* (domain objects to be classified) are evaluated as they are passed from node to node, starting from the root. At any node in the tree, if the case matches the condition of the node, the node is said to *fire*. If a node fires, the example is passed to the next node following the *except* edge. Otherwise, the case is passed down the *if not* edge, if there is any. This determines a path through a SCRDR tree for a case. The final conclusion is the conclusion of the node that fired last, i.e. that is deepest in the tree (but is often not a leaf node). To ensure that at least one node fires, usually the root contains a trivial condition to associate the example to the default class. RDR is a binary decision tree, but differs from standard decision trees in that compound clauses are used to determine branching, and these clauses need not exhaustively cover all cases so that it is possible for a decision to be reached at an interior node. Also, in practice an SCRDR tree tends to be very unbalanced with most edges being *if not* edges.

When an instance is misclassified, a new node (a new rule) is added as a child of the node that incorrectly matched the instance (called the *cornerstone* node), the new node is attached to the last node along the evaluation path (which is a leaf of the tree). If the node has no *except* link, then the new node is attached using an *except* link, otherwise, the new node is attached using an *if not* link. To determine the rule for the new node, the domain expert formulates a rule which is satisfied by the new example that generated the error, but not by the case(s) associated to the cornerstone node (the last node which fired in the evaluation path). This rule represents an explanation for why the conclusion for the case at hand should be different.

The strength of RDR is easy maintenance: the point of failure is automatically identified, the expert patches the knowledge only locally, considering the case at hand, and new rules are placed by the system in the correct position and checked for consistency with cases previously correctly classified, so that unwanted indirect effects of rule interactions are avoided. The manual creation of rules, in contrast with machine learning, requires a smaller quantity of annotated data, and the human in the loop provides important guidance for building a knowledge base which learning techniques can usually only achieve with the use of much more data.

We should also note that RDR was introduced to allow a domain expert to directly build knowledge bases, without the need to involve a knowledge engineer. In our case we designed a system which we believe could be used directly by a legal expert with no engineering skills. However, we did not have the opportunity to involve a legal expert and thus the rules were made by one of the authors, a computer engineer with no formal legal training.

RDR have been applied to different problems and applications; for an overview see (Richards, 2009). RDR have been extended to tackle natural language processing tasks with the system KAFTIE (Pham & Hoffmann, 2004, 2005). KAFTIE introduces a new rule language, with patterns over annotations as rule conditions, and an incremental knowledge acquisition framework to support the rapid prototyping of new NLP systems for various tasks. It has been applied to several tasks, including summarization (Hoffmann & Pham, 2003), extracting positive attributions, extracting temporal relations and citation classification (Pham & Hoffmann, 2003). Other application of RDR to text processing include extracting knowledge from medical text (Ruiz-Sánchez, Valencia-Garcıa, Fernández-Breis, Martınez-Béjar, & Compton, 2003; Valencia-Garcıa, Ruiz-Sánchez, Vivancos-Vicente, Fernández-Breis, & Martınez-Béjar, 2004), email classification (Ho, Wobcke, & Compton, 2003; Krzywicki & Wobcke, 2009), document retrieval (Kim & Compton, 2004, 2006), part of speech tagging (Xu & Hoffmann, 2010) and open information extraction for the web (Kim, Compton, & Kim, 2011).

This paper builds to a significant extent on previous work on RDR. It extends this by demonstrating that the Ripple-down Rule approach can be successfully applied to more complex NLP domains. Our system introduces several new features to facilitate knowledge acquisition in situations where an annotated dataset is available, supporting a better tuning of the rules with feedback and suggestions, both computed from the dataset. These mechanisms are described in Section 6.

5. Corpus

Decisions of different courts are often made publicly available, in Australia this service is provided by AustLII[1] (the Australasian Legal Information Institute) (Greenleaf, Mowbray, King, & Van Dijk, 1995). The Australasian Legal Information Institute is an online resource for Australian legal material, providing free Internet access to Australasian public legal information (legislation, treaties and decisions of courts and tribunals – case law, law journals...). AustLII is one of the largest sources of legal materials on the net, with over four million searchable documents, a number which increases every day. This resource is widely accessed with over 900,000 hits daily. A similar project, the World Legal Information Institute[2] (WorldLII), is the extension of this database to other countries, with the aim of providing "*free, independent and non-profit access to worldwide law*". WorldLII (which is maintained by AustLII) includes over 1230 databases of case-law and other materials, from 123 jurisdictions in 20 countries, with the majority of the material coming from Commonwealth countries.

Legal case reports contain citations to other cases: in AustLII all the cases are available as html pages, and some of the citations are automatically marked up with hyperlinks to the corresponding cited cases. Citation data of this kind is largely available; however, annotated data, which is needed to perform experiments on automatic classification, is more scarce. Among the many databases, related to Commonwealth of Australia case law, our attention was caught by the Federal Court of Australia (FCA) cases. This collection of cases is the only one in AustLII where for many cases at the beginning of a case, there is a list of all the citations (analogous to the bibliography section in a scientific paper), and for each of them the type of citation is indicated (such as "applied" or "distinguished"), taken from an annotation scheme of 24 classes, an example is given in Fig. 1. This information is added by the legal authors of the document, but is not present in all cases, only a subset. This classification of citations in FCA cases in AustLII is not the only classification existing, as many commercial providers have similar schemes, but to our knowledge it is the only one from which a research corpus can be readily constructed and made available.

We built a robust parser to analyse the case reports and extract information about citations, in order to build a corpus of annotated legal citations. We downloaded all cases from FCA for the years 2007, 2008 and 2009; however, not all cases have labelled citations. Our parser analyses the html pages of the reports and extracts information about the citations. We downloaded 5705 documents from the three years, and among these 2027 contain labels attached to the citations. In total we identified 18715 labelled citations. We also downloaded all the cases from 2006 (1834 documents, 727 with citations labelled, a total of 6541 citations), which we left to be used later as a test set. As the corpus was built in 2010 more recent cases were not available. It seems there is no other existing corpus of annotated legal citations available, and it could prove very useful for studying the function of citations in the legal domain. We have made a corpus of all these citations available on-line in XML format for other interested researchers.[3]

We recorded for each citation the destination and source cases and the type of citation as indicated by the legal authors; the class distribution is presented in Table 1. We should note that different principles in the primary case may be treated differently in the same citing case, so that combinations such as applied/distinguished are possible (indicating that one principle was applied and another distinguished), so there may be more than one classification label for each citation.

For our classification experiment we focused on identifying **Distinguished** (D) citations, which we believe to be particularly significant from the legal research point of view. The class Distinguished is the most common negative history marker found, and was the focus also of the work of Conrad and Dabney (2001), described in Section 3, who claim that "*from a legal research standpoint it is more important to have overturned, weakened or questioned opinions revealed rather than have reaffirmed or respected decisions featured*".

5.1. Human agreement

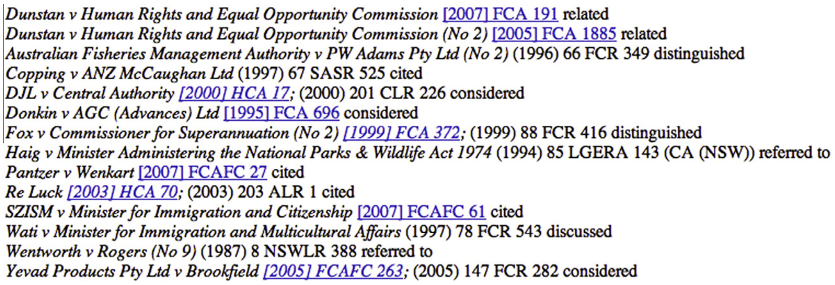
During the initial knowledge acquisition phase, we observed that often the classification of cases is ambiguous and different class labels can be appropriate (Posner (1999) already noted that "*because the cost of inaccurate citing usually is low, there is much careless citing; and so quantitative studies of citations are bound to contain a lot of noise*").

To measure the level of agreement among experts in classifying citations, we compared the classification given by the FCA, as found in AustLII, with the available classifications of the same citations by

[1] http://www.austlii.edu.au.

[2] http://www.worldlii.org.

[3] The corpus can be downloaded from http://www.cse.unsw.edu.au/~galganif/FilippoGalgani/Corpus.html and http://archive.ics.uci.edu/ml/datasets/Legal+Case+Reports



Dunstan v Human Rights and Equal Opportunity Commission [2007] FCA 191 related
Dunstan v Human Rights and Equal Opportunity Commission (No 2) [2005] FCA 1885 related
Australian Fisheries Management Authority v PW Adams Pty Ltd (No 2) (1996) 66 FCR 349 distinguished
Copping v ANZ McCaughan Ltd (1997) 67 SASR 525 cited
DJL v Central Authority *[2000] HCA 17*; (2000) 201 CLR 226 considered
Donkin v AGC (Advances) Ltd [1995] FCA 696 considered
Fox v Commissioner for Superannuation (No 2) *[1999] FCA 372*; (1999) 88 FCR 416 distinguished
Haig v Minister Administering the National Parks & Wildlife Act 1974 (1994) 85 LGERA 143 (CA (NSW)) referred to
Pantzer v Wenkart [2007] FCAFC 27 cited
Re Luck *[2003] HCA 70*; (2003) 203 ALR 1 cited
SZISM v Minister for Immigration and Citizenship [2007] FCAFC 61 cited
Wati v Minister for Immigration and Multicultural Affairs (1997) 78 FCR 543 discussed
Wentworth v Rogers (No 9) (1987) 8 NSWLR 388 referred to
Yevad Products Pty Ltd v Brookfield *[2005] FCAFC 263*; (2005) 147 FCR 282 considered

Fig. 1. At the beginning of the case, a list of citations with the corresponding classes.

Table 2
Agreement between AustLII (rows) and CaseBase (columns). Kappa = 0.43. The citations that form our training set are in bold.

	D	FA	Other
D	**225**	8	161
FA	9	**1041**	1777
Other	65	468	11313

the Lexis Nexis CaseBase Case Citator, a commercial database of case law.[4] The CaseBase classification scheme agrees with AustLII on several classes, but also introduces new ones. Comparing citation labels between the two, we found that human-expert-provided class labels are not necessarily agreed with by other human experts, with 47.1% of the citations having the same label in both databases, and 52.9% having different classifications.

In effect, this results in noise in our data for the purpose of training and testing our classifiers. As a consequence of this considerable discrepancy in human expert opinion, which we think is due to the class boundaries not being very sharp, it appears to be more appropriate to limit our study to those cases where the human-provided class labels from FCA and Lexis Nexis CaseBase agree. For this reason we selected only those citations which were classified as Distinguished by both AustLII and CaseBase.

To further make the class boundaries well defined, we selected as the negative class (citations not belonging to the D class) only cases belonging to either **Followed** or **Applied** (FA) citations, for which the two sources agreed. We thus considered a two-class problem, D versus FA. This was only for training, our intention was that the system should be able to discriminate Distinguished from all other classes. Table 2 shows the agreement between the two sources on this two-class problem, which are rarely confused with each other. While we built our system considering only D and FA citations, we tested it also with citations of other types, to check its performance with the more difficult full datasets (Section 7).

Our initial corpus contained 460 citations of class Distinguished and 3496 Followed or Applied citations. This number is lower than the sum of the Followed and Applied counts as some citations are classified as both Followed and Applied. Of the initial 460 FCA cases marked as Distinguished in AustLII, 225 (48.9%) had the same label in CaseBase, 8 were classified as either Follower or Applied and 161 as another class. Of the 3496 Followed or Applied citations in AustLII, only 1041 (29.8%) were labelled either Followed or Applied in CaseBase, while 9 were labelled Distinguished (see Table 2). The test set contains 72 D cases and 257 FA cases, agreed by both sources.

5.2. Citation context

Locating the portion of text, the citation context, where the citation is actually made by the judge is a more complex task, see for example (Kaplan, Iida, & Tokunaga, 2009; Qazvinian & Radev, 2010; Ritchie et al., 2008) for the analogous problem in scientific papers, which however present more standard forms of in-text citations. In legal cases in-text citations can be presented in different ways, including:

- The full name of the case, e.g. *Yevad Products Pty Ltd v Brookfield [2005] FCAFC 263; (2005) 147 FCR 282*, which is often abbreviated to the name of the parties, e.g. *Yevad Products Pty Ltd v Brookfield*.
- The name of just one of the parties, e.g. *Yevad*.
- An indication of the law report, e.g. *(2005) 147 FCR 282*, or a court identifier and decision number, e.g. *[2005] FCAFC 263*.
- A combination of these components, for example *Brookfield 147 FCR*.
- It is also not unusual that a case is referred to just with the name of the respective judge, e.g. *I understand Brennan J's reference to the prospect of a grant of special leave*. . .. We resolved this type of reference only where a judge's name is only involved in one possible cited case.

Some of the references are already marked up with html links in AustLII (Mowbray et al., 2009). Where html links were not available, we resolved all these types of references automatically using a shallow parser based on regular expressions.

We created a corpus which contains for each citation its class label(s) and the associated paragraph(s) in which the citation occurs (Galgani & Hoffmann, 2010). Often cited cases are mentioned more than once so that we have several paragraphs for each citation; it is also common to have citations to different cases in the same paragraph. This corpus can be directly used for supervised machine learning or for knowledge acquisition as described later. The citation contexts (paragraphs where the citation occurs) for all the 25256 citations are included in the corpus that we have made available.

6. The LEXA knowledge acquisition framework

6.1. Annotations and rule language

Our approach to citation classification is based on building a knowledge base that annotates text at different levels. The knowledge base contains rules that specify certain conditions, if these are satisfied portions of text (whose beginning and end positions are defined in the rule) are annotated with various labels and a list of feature-value pairs. A special kind of annotation, the CLASS annotation, gives the predicted class of a citation. The other annotations are used as support for CLASS annotations.

[4] http://www.lexisnexis.com.au/en-au/products/casebase-case-citator-online.page.

The implementation of the annotations is based on the open source platform GATE[5] (Cunningham, Maynard, Bontcheva, & Tablan, 2002). A first layer of linguistic annotations is posted automatically by the preprocessing tools provided with GATE. We used the Tokenizer, Sentence Splitter, Part of Speech Tagger, Stemmer and Morphological Analyser to generate *Token* annotations and their corresponding features: Token annotations cover every token and have features such as the root, stem and category (part of speech tag) of the word; sentence boundaries are also individuated.

The other annotations are created by the rules of the knowledge base, and were implemented using the Semantic Tagger from GATE (based on JAPE grammars). These user-created rules can be divided in two groups. A first group of rules in a preliminary phase annotates some domain specific information: judge names (i.e. *Brennan J*, possibly linked to the corresponding case), parties involved (e.g. *the plaintiff*, *the appellant*...), courts (e.g. *a full court of the HCA*...), citations of paragraphs (e.g. *case at [145–148]*), etc. The user can create new annotations as needed, or modify the existing ones with new rules.

The second group of rules posts class labels over citations, creating CLASS annotations which specify which class we believe a citation belongs to, based on the previously posted annotations and other text features. The following shows a typical portion of annotated text (not all annotations are displayed):

> *The facts of* [CASE.type=target *Hughes*] [TOKEN.root=be *were*] *vastly different from those of* [CASE.type=this *the present case*].

We use the following notation: [ANNOTATION.feat=val *some text*] means that there is an annotation *ANNOTATION* that spans *some text*, and has a feature *feat* of value *val* (we do not show all the annotations but only the relevant ones). In this example the Token annotation is posted by the linguistic preprocessing module (one feature, *root*, is shown, but there are others), while CASE is posted by the first group of rules: CASE is an annotation which recognizes when a case is mentioned. To classify an instance we post the CLASS annotation over CASE, for example:

> *The facts of* [CLASS.distinguished [CASE.type=target *Hughes*]] [Token.root=be *were*] *vastly different from those of* [CASE.type=this *the present case*].

The user can at any stage create new annotations and use them in creating rules. Every rule is of the form "*if Condition then Conclusion*", where the condition is a pattern: if the pattern matches a span of text, the conclusion is executed. When designing the system we aimed at making the creation of rules as easy as possible. Special attention was given to the rule language, which needs to be powerful in order to identify patterns expressive enough to classify citations.

The condition part of any rule is a regular expression of tokens and other annotations and if the condition applies, the conclusion of the rule specifies the action to be taken. This is usually a new annotation to be posted (specifying name, features and position of the annotation), but it can also involve deletion or modification of existing annotations (or features). The pattern in the condition is a regular expression which can include both linguistic properties (e.g. POS, token, stem) or other annotations (concepts previously identified).

We express rules in the form *Pattern* − > *Conclusion* (where conclusion is the new annotation posted), we use *Distinguished* as a short form for the *CLASS.Distinguished* annotation posted over *CASE*. In the condition part, for brevity we use the following conventions: the pattern [*ANNOTATION*] requires the presence of the annotation; *someword* is a short form for [Token.string=*someword*]; we also use *GAPN* to indicate a gap of maximum length N (short form for [*Token*]{0,N}). An example of a simple rule would be:

> [CASE] [Token.root=be] [GAP4.containsnot='negation'] *different* − > Distinguished
>
> Which would match:
>
> *The facts of* [CASE.type=target ***Hughes***] [Token.root=be ***were***] ***vastly different*** from those of [CASE.type=this *the present case].*
>
> The same rule would not match:
>
> *The facts of* [CASE.type=target *Hughes*] [Token.root=be *were*] [NEGATION *not*] *vastly different from those of* [CASE.type=this *the present case].*

because of the presence of the negation annotation, or

> *Vastly different facts from* [CASE.type=this *the present case*] [Token.root=be *were*] *those of* [CASE.type=target *Hughes].*

due to the different ordering of the components.

We believe that this rule language has sufficient expressiveness to enable the user to specify complex rules, while keeping it reasonably easy to create conditions. The user can also define recurrent concepts and then use them in the rules, in a multi-level fashion. Using regular expressions it is possible to specify complex conditions such as "take the first case up to two sentences after", or "there must be annotation X in the same paragraph (or sentence)" and so on. Specific annotations can be used to create rich lexicons; for example [REFCASE] was created during the development to identify phrases that refer to a previously mentioned case (e.g containing phrases like "*that case*", "*that proceeding*", "*the case before* [JUDGE]"...), with such annotations linked to cases with a set of rules. This mechanism lets the rules recognize text at different abstraction levels (string, root, stem and group created by the user with specific annotations), so that a larger variety of linguistic realization can be captured by the patterns.

In our exploratory study we created rules directly using the JAPE language, as the rules were created by one of the authors. A system used by legal experts would need a suitable interface to facilitate writing rules, providing a translation between simple regular expression patterns and the JAPE language. For example, it would not be difficult to provide an interface that would translate the rule

> [CASE] [Token.root=be] [GAP4.containsnot='negation'] *different* − > Distinguished

into the corresponding JAPE version. Because our experiments are only meant to demonstrate the feasibility of the proposed approach, we did not implement such an interface.

6.2. Rule acquisition

Rule acquisition proceeds according to the Ripple-down Rule methodology as described in Section 4 (an example of a portion of the RDR tree is shown in Fig. 2). The following steps are involved:

1. The user is presented with a citation that is currently misclassified by the KB, which can be a citation which received a wrong label, or which has not received any label yet.
2. The system shows the relevant text where the cited case is mentioned (whose length can vary between few sentences to several paragraphs) together with the list of rules, if any, that give the wrong classification for this case. The correct class, known from the stored annotation is also shown.
3. Based on the content of the text and using the annotations present, the user writes a pattern that matches the current text and associates it with the correct conclusion.

[5] http://www.gate.ac.uk/.

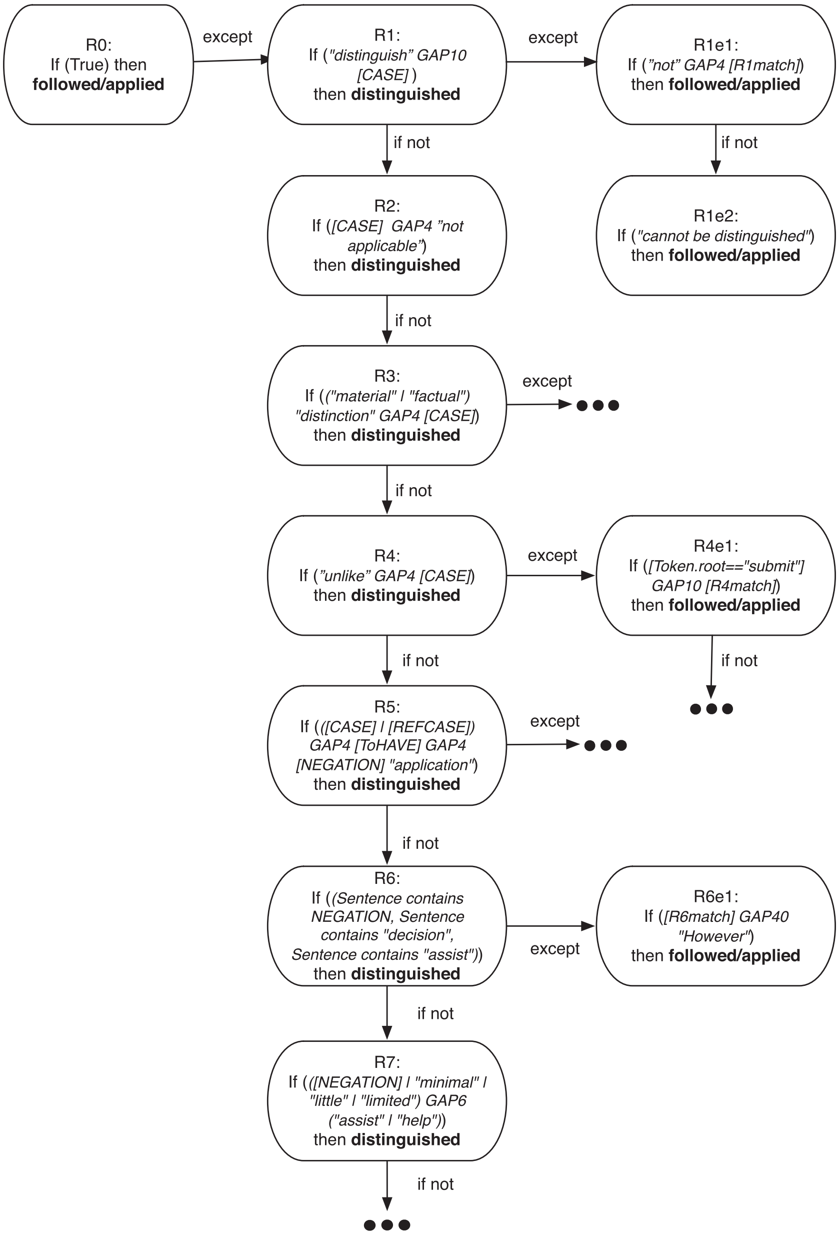

R0:
If (True) then
followed/applied
except
R1:
If ("distinguish" GAP10 [CASE])
then distinguished
except
R1e1:
If ("not" GAP4 [R1match])
then followed/applied
if not
R2:
If ([CASE] GAP4 "not applicable")
then distinguished
if not
R1e2:
If ("cannot be distinguished")
then followed/applied
if not
R3:
If (("material" | "factual") "distinction" GAP4 [CASE])
then distinguished
except
if not
R4:
If ("unlike" GAP4 [CASE])
then distinguished
except
R4e1:
If ([Token.root=="submit"] GAP10 [R4match])
then followed/applied
if not
if not
R5:
If (([CASE] | [REFCASE]) GAP4 [ToHAVE] GAP4 [NEGATION] "application")
then distinguished
except
if not
R6:
If ((Sentence contains NEGATION, Sentence contains "decision", Sentence contains "assist"))
then distinguished
except
R6e1:
If ([R6match] GAP40 "However")
then followed/applied
if not
R7:
If (([NEGATION] | "minimal" | "little" | "limited") GAP6 ("assist" | "help"))
then distinguished
if not

Fig. 2. Portion of RDR tree. The notation is explained in Section 6.1 (in the system the patterns are encoded with the JAPE language).

4. The rule is tested for consistency on all the corpus, showing all the cases whose classification would be changed by this rule. The user can in this way estimate the performance of the rule, getting a measure of precision and recall of the single rule, and decide based on this information how to modify the rule, for example making it more specific or more general.
5. When the user is satisfied the rule is committed to the KB and inserted in the RDR tree by the system.

For example the user is presented with a portion of text, where a case is mentioned, but not yet classified by the knowledge base (step 1). The system displays the following text and indicates that *Hughes* is a Distinguished case (step 2):

> *Eden refers to* [CASE *Hughes Aircraft Systems International v Airservices Australia (1997) 76 FCR 151 (Hughes)*] *and* [CASE *Cubic Transportation Systems Inc v State of New South Wales [2002] NSWSC 656 (Cubic Transportation)*]. *The facts of* [CASE *Hughes*] *were vastly different from those of the present case. In*[CASE *Hughes*] , *there were two competing tenderers, the applicant in the proceeding being the unsuccessful one.* [REFCASE *Finn J*] *held that in the circumstances of the case, there was an implied contract that might be called a "process contract", according to which Airservices Australia incurred implied contractual obligations in favor of the tenderers, including an implied obligation to conduct its evaluation fairly and in a manner that would ensure equal opportunity as between them.*

The user locates a segment of text considered to be the justification for choosing the class Distinguished (step 3). For example:

> *the facts of* [CASE] *were vastly different* $->$ Distinguished

A rule like this is very specific; in fact it would not match any other case in the corpus (and this would be indicated by the system). The user can modify the condition part by building regular expressions (for example some words may be optional) and insert gaps between the components. In this case the system suggests substituting *were* with the annotation *toBE*, a lexicon (previously created by the user) which includes all the forms of the verb “to be” together with synonyms such as “appear”, “is considered”, etc…For example the rule could be modified to:

> [CASE] [GAP8] [toBE] [GAP4] *different* − > Distinguished

The system (step 4) shows other cases matched by this rule, and it suggests other kinds of generalisations, for example the annotation [REFCASE] (introduced earlier) can be used as an alternative to [CASE]. For each modification the user can see the effects of the rule on the corpus, visualizing the number of correct and incorrect classifications and perhaps inspecting the corresponding texts. After looking at the results the user may come up with the following rule:

> ([CASE] | [REFCASE]) ([GAP8] | [Split]) [toBE] [GAP4] *different* − > Distinguished

where [Split] indicate that the first and second part may be in two different (but adjacent) sentences. This expression is matched with all cases of the corresponding node, and the user obtains an indication of how many cases are correctly classified (in this case 12 out of 17 matches are correct) and can inspect the other matches, so that the rule can be refined if necessary. An example of another match for the same rule is:

> *Similarly, in my view the factual position confronting Allsop J in* [CASE ***SZJBA***] [toBE ***was***] ***very different*** *from that which arises in this appeal, and the failure to inquire cannot be seen on its face to subvert the observance of the Tribunal of its obligation to give procedural fairness.*

When the user is satisfied with its quality, the rule is committed to the knowledge base (step 5).

The critical step of the process is the creation of the rule, which ideally covers as many cases as possible while retaining high precision. The system supports the user both providing an indication of the precision and the number of cases covered, estimated on the training set, and selecting and displaying relevant examples from the training set. Some examples of the automatic support given by the system in creating rules are provided in Table 3. This real-time validation of the rules in step 4 is our major extension to the standard RDR methodology: we use the available dataset to provide feedback both showing new examples that are matched by the current rule, as well as providing estimates of rule precision and coverage on the whole dataset. While estimating rules’ accuracy on an available dataset is common practice in data mining and machine learning methods, it has not previously been part of direct feedback to the user during RDR rule creation, probably because RDR has focussed more on dealing with a stream of cases as they occur rather than working through a dataset.

After committing a rule it is possible to inspect any errors resulting from this rule, in order to create exception rules. For example one case for which the previous rule introduces an error is:

> *The actual provisions considered in* [Class.Distinguished [CASE ***Emanuele [1997] HCA 20; 188 CLR 114***] ***are not materially different***] *to those with which I am concerned, and in my view the analysis of the High Court in that decision is equally applicable to ss 236 and 237 of the Act.*

To correct the error, the user creates a rule repeating all the necessary steps as described above. For example a simple rule to correct this case would be:

> [Class.Distinguished.contains=’negation’] − > Followed/Applied

Which would be inserted as an exception rule of the previously created rule, thus being tested for all its matches.

Table 3
An example of the automatic support that our system LEXA provides to guide the creation of the rule discussed above. For each version of the rule the system displays the number of correct and total matches, examples of correct and incorrect matches (the part matched by the rule is in bold), and possible generalizations (underlining terms that belong to a user created lexicon, *toBE* in the example).

facts of [CASE] *were vastly different* − > Distinguished
Matches = 1(correct)/1(total), precision = 1
Examples of correct classification:
The **facts of** [CASE ***Hughes***] ***were vastly different*** *from those of the present case.*
Examples of incorrect classification:
None
Possible generalizations:
CASE belongs to the group REFCASE
were belongs to the group toBE

[CASE] [GAP8] [toBE] [GAP4] *different* − > Distinguished
Matches = 8(correct)/14(total), precision = 0.571
Examples of correct classification:
Similarly, in my view the factual position confronting Allsop J in[CASE ***SZJBA***] [toBE ***was***] ***very different*** *from that which arises in this appeal, and the failure to inquire cannot be seen on its face to subvert the observance of the Tribunal of its obligation to give procedural fairness.* [7 more]
Examples of incorrect classification:
The actual provisions considered in [CASE ***Emanuele [1997] HCA 20; 188 CLR 114***] [toBE ***are***] ***not materially different*** *to those with which I am concerned, and in my view the analysis of the High Court in that decision is equally applicable to ss 236 and 237 of the Act.* [5 more]
Possible generalizations:
CASE belongs to the group REFCASE

([CASE] | [REFCASE]) ([GAP8] | [Split]) [toBE] [GAP4] *different* − > Distinguished
Matches = 27(correct)/44(total), precision = 0.614
Examples of correct classification:
In SZICU v Minister for Immigration and Citizenship [2008] FCAFC 1, the Full Court distinguished SZJBA on the basis that [REFCASE ***that decision***] [toBE ***was***] ***different*** because in that case ’there was a failure to inquire into readily available and centrally relevant information’ (at [29]). Similarly,…[26 more]
Examples of incorrect classification:
The position in [CASE ***Re Addstone***] [toBE ***was***] ***slightly different*** from that which occurs in the present case. However, there are a number of aspects of the Deed of Settlement which seem to me to point in favor of the confidentiality orders at least for the time be…[16 more]
Possible generalizations:
None

When creating exception rules, the user can save default exception rules, which can be tested automatically with any new rule inserted. This mechanism was introduced to avoid creating the same rule in different parts of the tree. For example default exception rules were made which look for a negation inside the match, or for the word “*However*” just after the match, and similar constructs. When the user decides to create exceptions for a specific node, these default rules are tested on the node and the outcome (in terms of number of total/correct matches) is shown to the user who can quickly select which rule to add to the node.

The possibility to save and re-use exception rules, automatically tested when a rule is created, is another improvement we made to the classic RDR framework. We also retain all the advantages of adopting an RDR structure: the process of organization the knowledge is completely managed by the system, rather than by the user: a rule is inserted as a justification for one case, and is valid only for that specific context (i.e. cases that traverse the tree until that node). The rules are automatically inserted in the appropriate position in the tree independent of the user, so that the new rule does not alter the effect of previous rules (except for the node being modified) with unwanted interactions. The user only provides a rule to correct a specific case, and never has to position the rule in the tree. The user is concerned only with the current rule and the cases covered by that rule, as exemplified in Table 3. As we mentioned in Section 4, one of the aims of using RDR is to allow a domain expert to build a knowledge base directly, without the need for a knowledge engineer. Unfortunately we could not test if the rule language and knowledge acquisition procedure are simple enough to be used with ease by a legal expert in order to build complex knowledge bases. Though we had favorable comments from legal experts about the usability of the language, we do not have experience with them actually using it.

The knowledge base we built recognizes Distinguished (D) cases, as opposed to a background class. The examples we used for the background class were Followed or Applied (FA) (see Section 5) but it can be any other class when applying the rules. The classification part of the KB is composed of a default rule (which assigns the majority or background class, FA) and two kinds of rule: to identify D citations and to revert to FA (background class) citations. The user can choose to see a case which is not classified yet (i.e. a D case which is matched only by the default rule), and to make a rule to capture it and similar cases, thus improving the recall of the D class. Otherwise the user can choose to see cases incorrectly classified as D by a specific rule, and create one or more exception rules to fix the mistake. For each existing rule the system records which cases are correctly/incorrectly classified, so that the user can see the performance of every single rule, decide where to intervene and refine the rules by adding exceptions to increase the precision of the system. In this way the system facilitates the inspection of the data, both in the form of non-classified data, and wrongly/correctly classified cases for each rule. The training corpus is used to indicate the performance of created rules, showing statistics on the number of correct and incorrect matches.

6.3. Suggesting synonyms

One of the main challenges in creating good rules is the large variety of different terms and expressions that a judge can use to express the same or similar concepts. To deal with this variety of possible linguistic realizations, the system supports generalization of terms at different levels: the user can build more general rules by abstracting a term to its stem or lemma, or by specifying lists of synonyms, or by using a lexicon to represent a general concept.

In order to increase the generalization capabilities of the rules, we introduced a module to suggest possible synonyms to the user when creating a rule, to try to cover other cases which use different terminology, but the same structure. Our system obtains synonyms from two different sources:

- WordNet 3.0 (Miller, 1995) is a lexical database that encodes the majority of English terms, organized in sets of synonyms known as synsets. To find synonyms of a given a term, we look for it in any of these synsets, and we extract all other lemmas of the synset as synonyms.
- The LOIS (Lexical Ontologies for Legal Information Sharing) database (Peters, Sagri, & Tiscornia, 2007) is a legal knowledge base consisting of legal WordNets in six languages (English, Italian, Dutch, Portuguese, German and Czech), which shows the correspondence of legal concepts and lexical items from different languages. We use only the English part. In the same way as WordNet we can use the synsets to obtain synonyms of a given word, and use semantic links of type *synonymy*, *near_synonymy* and *xpos_near_synonymy* to extract synonyms from similar synsets.

While the LOIS vocabulary is not very big (we extracted 866 groups of synonyms from the English part), it focuses on legal terminology and so it is a valuable addition to WordNet, suggesting words pertinent to the legal domain. On the other hand WordNet provides a larger number of synonyms, some of which are irrelevant, as it often suggests synonyms with different word senses from the intended one (we do not perform any sense disambiguation). When available, other synonym resources can be easily added to improve the suggestion of relevant synonyms.

The synonym component uses these two resources and the training corpus to suggest meaningful synonyms to the user when creating a rule. When the user creates a rule to match the case at hand, he/she inserts some of the terms from the case at hand into the rule. The system gets a list of synonyms for every word (from the Lois and WordNet databases): for each term in this list the system also reports to the user if the term exists in the whole corpus, and then counts the matches for the given pattern (i.e. new matches of the rule if we substitute the original term with each synonym). The user can thus judge the number of correct and incorrect matches for each possible synonym, as well as inspect the individual matches, and based on this information decide which synonyms should be included in the rule. The system also indicates to the user, for each word, which new cases it would cover matching not only the word itself but also its root or its stem. The user can easily select which synonyms to insert in the rule as a disjunction. The user can choose to insert words that have no matches in the corpus if they are considered appropriate, with the aim of generalizing over terms that do not appear in the training corpus. An example for the rule described in the previous section is given in Table 4.

Besides the two databases of synonyms, the system also checks, for each word, if it is present in any other rule of the knowledge base. If a rule is found, it is indicated to the user who can pick up other synonyms from the previous rule. The user can also create new synonym groups comprising of a set of words (to be reused later) and create an annotation to identify the group. The system will suggest if there are groups which contain any of the words used. Dealing with the rich variety of linguistic realizations for the same or similar concepts is a bottleneck for many text analysis

Table 4
An example of system information for the synonym component. For each possible synonym, the system specifies how many matches (correct/total) we would get using the proposed synonym instead of the original term. The first two entries of the list substitute the term with its stem or its root. We can see for example that the word "*distinguishable*" has 40 matches, of which 30 are correct, and so we can include it in the rule. It may be valuable to add also other words even if they have no matches ("*dissimilar*" in the example). Part of the system output is omitted for brevity.

[CASE] [GAP8] [toBE] [GAP4] *different* − > Distinguished

Matches = 8(correct)/14(total), precision = 0.571

Examples of correct classification:

Similarly, in my view the factual position confronting Allsop J in[CASE ***SZJBA***] [toBE ***was***] ***very different*** *from that which arises in this appeal, and the failure to inquire cannot be seen on its face to subvert the observance of the Tribunal of its obligation to give procedural fairness.* [7 more]

Examples of incorrect classification:

. . .[5 more]

Possible synonyms for *different*:

different is already present in rules 7, 14, 22 [view]

stem(*differ*): matches = 9(correct)/22(total) [view examples]

root(*different*): matches = 8(correct)/14(total) [view examples]

dissimilar: matches = 0(correct)/0(total) [no examples]

contrary: matches = 0(correct)/1(total) [view examples]

distinguishable: matches = 30(correct)/40(total) [view examples]

distinct: matches = 0(correct)/0(total) [no examples]

. . .

[CASE] [GAP8] [toBE] [GAP4] (*different*|*dissimilar*|*distinguishable*) − > Distinguished

Matches = 38(correct)/54(total), precision = 0.704

Examples of correct classification:

Both [CASE *Mobileworld Communications [2003] FCA 1404; (2003) 61 IPR 98*] *and* [CASE ***Global Brand Marketing [2008] FCA 605; (2008) 76 IPR 161***] [toBE ***are***] ***distinguishable*** from this case because in both cases the ownership position was clear. [11 more]

Examples of incorrect classification:

. . .[4 more]

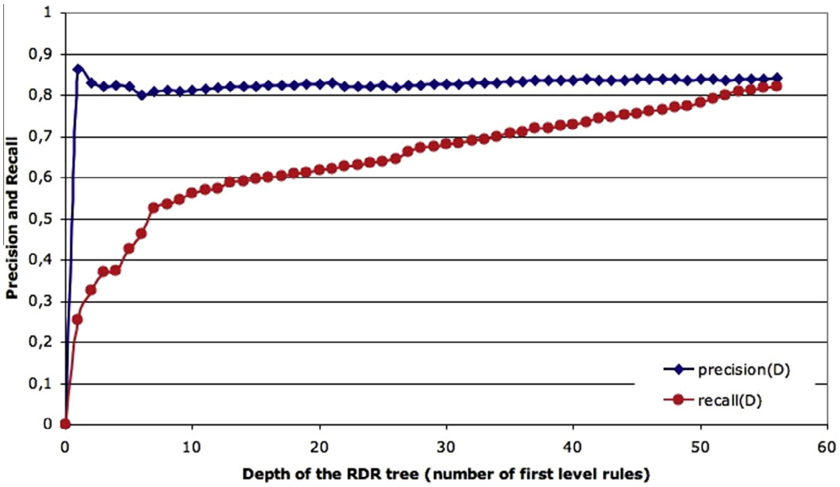


Fig. 3. Recall and precision of the D class, as rules are added to the KB, on the training set (225 D instances).

systems. We believe that this synonym identification support (both from the two databases, from the previously created rules and from the groups created by the user) facilitates the creation of more general rules, thus reducing the knowledge acquisition effort required, without increasing the complexity of building rules. The automatic test of all synonyms on the corpus lets the user quickly identify which synonyms increase the number of correct matches.

7. Experimental results and comparison with machine learning

As described in Section 5, we built a training set, extracting all **Distinguished** (D) and **Followed** or **Applied** (FA) citations, for which the original court classification is the same as the classification found in the Lexis Nexis CaseBase.

We conducted a number of knowledge acquisition sessions to build a knowledge base, using our training set of 225 Distinguished (D) cases and 1041 Followed/Applied (FA) cases. The rules were built by one of the authors (no legal expert was involved). Taking around 40 hours we developed a KB, which contains a total of 78 rules (of which 54 identify Distinguished citations and 24 are exception rules). During the development we found that most of the user's time was spent reading the text fragments surrounding the citations as these can often be lengthy (up to several paragraphs) and difficult to understand; the phase of actually creating and testing rules usually took a shorter time. Some cases were not understood by the user, who could not find a reason why a case was marked as Distinguished and thus no rule was made. The performances of the knowledge base for the D class as rules are added is shown in Fig. 3. The results are presented using the well-known measures of precision, recall and F-measure (computed with $\beta = 1$):

$$Precision(D) = \frac{number\ of\ cases\ correctly\ classified\ as\ D}{total\ number\ of\ cases\ classified\ as\ D\ by\ the\ system}$$

$$Recall(D) = \frac{number\ of\ cases\ correctly\ classified\ as\ D}{total\ number\ of\ cases\ of\ class\ D}$$

$$F(D) = \frac{2 \cdot Precision(D) \cdot Recall(D)}{Precision(D) + Recall(D)}$$

Or equivalently:

$$Precision(D) = \frac{true\ positives}{true\ positives + false\ positives}$$

$$Recall(D) = \frac{true\ positives}{true\ positives + false\ negatives}$$

As a test set, we used unseen data from FCA cases from the year 2006. We built three different test sets, of increasing difficulty:

1. Test set 1: only cases for which AustLII and CaseBase give the same classification, using FA cases as background (non-D) cases. This results in 72 D cases and 257 FA cases. This is the same procedure that we used to build the training set.
2. Test set 2: all citations classified by AustLII as either D or FA, regardless of their CaseBase classification. This corresponds to adding to test set 1 the AustLII citations for which the two sources do not agree. Test set 2 contains 160 D cases and 1276 FA cases.
3. Test set 3: we consider as the negative class (non-D) all citations with any classification other than Distinguished, as found in AustLII. This set contains the same D citations as test set 2, but it uses as non-D citations all other classes (not only FA citations), giving a total of 160 D cases and 6325 non-D cases.

In Table 5 we report for each test set the number of D citations correctly extracted (true positives), and the number of non-D citations (FA or other classes) incorrectly extracted (false positives).

In order to compare our system to machine learning approaches, we trained two types of models, a Naive Bayes (NB) classifier and a Support Vector Machine (SVM) classifier. We used a bag-of-words model, where for each citation the presence or absence of every term in the citation context is encoded in a boolean feature. To give to the machine learning approach the same information used in the rules, we used as a context the whole paragraph or paragraphs that contain the citation. In extracting all the words of the citation context, we experimented with two types of features: the words in their inflected form (for a total of 13575 features), or stemmed and stopword filtered words (total number of features, 9807). We found that stemming improves the performance of the machine learners.

We used the Porter stemmer (Porter, 1980) and the NLTK (Bird, Klein, & Loper, 2009) list of stopwords to pre-process the data, while the classifiers were trained and tested using WEKA, an open source machine learning software (Hall et al., 2009). Due to the fact that the models did not do well on recognizing the minority class Distinguished, we tried to improve their performance by building a balanced data set: we oversampled the D class (random sampling with replacement) obtaining a training set of 2084 cases (1041 FA

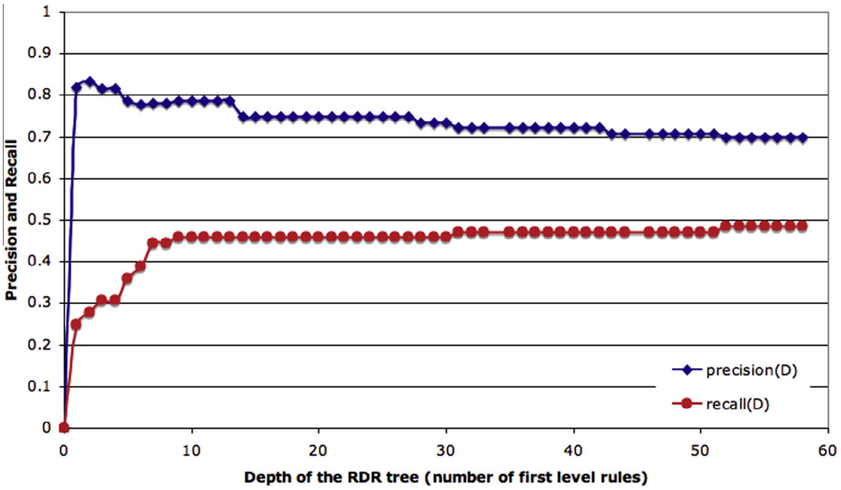


Fig. 4. Recall and precision of the D class, as rules are added to the KB, on test set 1 (72 D instances).

Table 5
Performances on the different test sets. For each test set, we report the number of correctly identified D citations (true positive – TP), and non-D citations incorrectly selected as D (false positive – FP). The performance of human annotations in CaseBase is given as an upper bound. The last row reports results on test set 3, but using the larger training set (not applicable to LEXA and CaseBase).

	LEXA		Best NB		Best SVM		Human-made CaseBase	
	TP	FP	TP	FP	TP	FP	TP	FP
Test set 1 (72 D + 257 FA cases)	35	15	27	22	28	12	72	0
Test set 2 (160 D + 1276 FA cases)	49	54	67	338	44	73	72	1
Test set 3 (160 D + 6325 other cases)	49	251	67	2088	44	530	72	31
Test set 3 Training on all cases	(49)	(251)	56	1281	13	69	(72)	(31)

Table 6
Results on the three test sets. NB_s stands for Naive Bayes using stemming, NBos for Naive Bayes using oversampling of D instances, other columns titled analogously (in bold the best results for each metric).

	LEXA	NB	NB_s	NBos	NBos_s	SVM	SVM_s	SVMos	SVMos_s
(a) Test set 1 (2006): 72 D cases and 257 FA cases. As the sample size is limited, the confidence intervals for the F-score overlap between LEXA and the ML techniques									
Pre(D)	**0.70**	0.57	0.55	0.33	0.32	0.66	**0.70**	0.67	**0.70**
Rec(D)	0.49	0.32	0.37	**0.57**	**0.57**	0.37	0.39	0.39	0.39
F(D)	**0.57**	0.41	0.45	0.42	0.41	0.48	0.50	0.49	0.50
(b) Test set 2 (2006): 160 D cases and 1276 FA cases. The F-score of LEXA is statistically significantly higher than NB, but not than SVM (p-value 0.05)									
Pre(D)	**0.47**	0.17	0.16	0.13	0.13	0.38	0.33	0.38	0.33
Rec(D)	0.31	0.35	0.42	0.67	**0.68**	0.26	0.22	0.27	0.22
F(D)	**0.37**	0.23	0.24	0.22	0.22	0.31	0.27	0.32	0.27
(c) Test set 3 (2006): 160 D cases and 6325 other cases. The F-score of LEXA is statistically significantly higher than NB and SVM (p-value 0.05)									
Pre(D)	**0.16**	0.03	0.03	0.02	0.03	0.07	0.06	0.08	0.06
Rec(D)	0.31	0.35	0.42	0.67	**0.68**	0.26	0.22	0.27	0.22
F(D)	**0.21**	0.06	0.06	0.05	0.05	0.11	0.10	0.12	0.10
(d) Test set 3, machine learners trained using the larger training set (460 D and 18070 non-D cases) – not applicable to LEXA									
Pre(D)	–	0.05	0.04	0.04	0.04	0.16	0.06	**0.13**	0.05
Rec(D)	–	0.14	0.13	0.35	**0.38**	0.08	0.06	0.07	0.05
F(D)	–	0.07	0.07	0.07	0.07	**0.11**	0.06	0.09	0.05

and 1041 D). However this technique did not result in any performance increase.

Table 5 compares the performance of our system to these machine learning approaches: it reports the number of D citations correctly identified (true positives), and the number of non-D citations which are incorrectly classified as D (false positives). Here we selected the best NB and SVM as those with higher F-score for the D class. The complete results for each model are given in Table 6. The two tables show that LEXA outperforms the best machine learners trained: it has a higher number of true positives and a lower number of false positives than SVM in all test sets. While Naive Bayes extracts more D citations, it also incorrectly extracts many more non-D citations. As the test data becomes more noisy, the performance of LEXA degrades more gracefully than those of the machine learning approaches, whose false positives increase at a higher rate. For example the NB precision drops to 3% for test set 3, while LEXA has a precision of 16%.

Initially we trained the machine learning models on the same dataset used to build the knowledge base (72 D and 257 FA citations). We tried to improve their performance providing more training data and built a larger training set taking all D cases as positive examples and all non-D cases as negative examples, without any agreement constraint. This increases the training set size to 460 D citations and 18070 non-D citations. In Tables 5 and 6 we

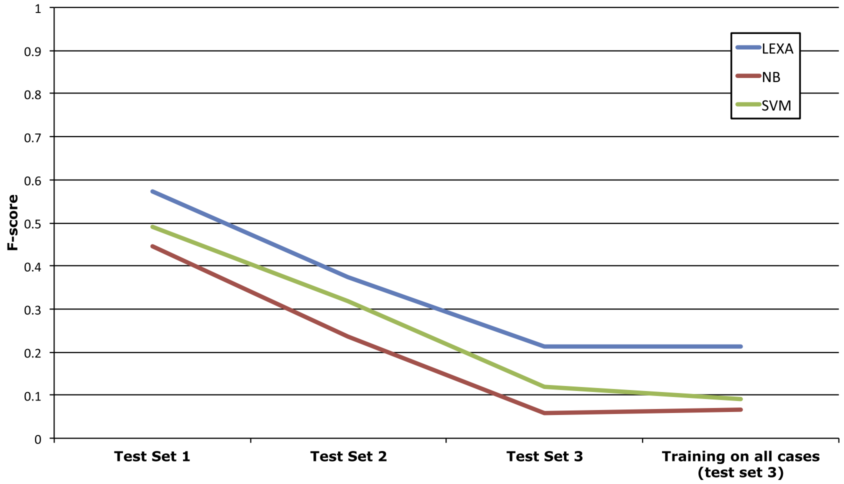


Fig. 5. F-score for the D class for LEXA and the best NB and SVM, for different test sets. LEXA performs better than the two ML approaches.

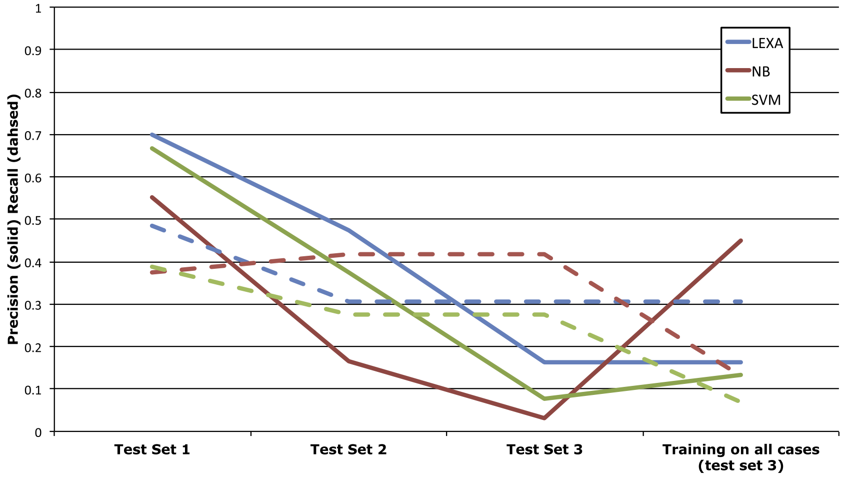


Fig. 6. Recall and precision of the D class for LEXA and the best NB and SVM, for different test sets.

can see that the larger training set slightly improves performance for Naive Bayes, but not for SVM. Both models extract fewer cases with a consequent increase in precision but drop in recall. This result supports our intuition to initially train the system with a restricted set of "cleaner" examples. Initially we had tried to create rules with all available test cases, but we found that for many cases the reason for one class or the other was not clear. This led to our decision to build a cleaner training set using a limited set of classes and agreement data.

The results in Tables 5 and 6 indicate that it is difficult to tune the Machine Learning algorithms to provide good levels of both precision and recall. While in some cases we can improve the number of D instances identified using oversampling (such as in the Naive Bayes case), this technique also increases the number of false positives. Table 5 clearly shows that LEXA consistently selects more D cases and less non-D cases.

The trends of the three approaches are depicted in Figs. 5 and 6. We can see that while SVM has a trend similar to LEXA, with both precision and recall decreasing at the same time, NB maintains the same level of recall for both test set 1 and test set 2 (the recall for test set 3 is the same as test set 2 as no D cases are added to the dataset). The bias towards favoring recall over precision for NB can be explained with the skewed nature of our dataset (there are more examples for FA than D). It is known that in presence of skewed data Naive Bayes' decision boundary weights can be biased (Rennie, Shih, Teevan, & Karger, 2003) (due to the factor that accounts for the prior probability of the class). The different test sets have different proportion of D and FA instances: the learner is trained on a set with a proportion similar to test set 1 (around 1:4), while test set 2 is even more skewed (1:8) and thus in this test set the classifiers is biased towards the D class (due to the prior learnt from the training set not reflecting the actual distribution in the test set), resulting in higher recall and lower precision. This explanation is confirmed by the fact that using a different training set which is more skewed (Table 6(d)) the recall goes down while precision increases. When we train the NB using oversampling, the difference in skewness between training and test set is larger and thus the bias towards recall increases (Table 6). As seen in Table 5 NB is thus classifying a large number of cases as D, with a high rate of false positives. SVM does not suffer from the same bias towards recall, as it manages better the unbalanced nature of the dataset, but it select less true positives and more false positives than LEXA. As seen in Figs. 5 and 6 the two models have very similar trends, but LEXA is consistently superior to SVM for any test set.

Table 5 reports also the number of citations for which our second source of human labeling (CaseBase) agrees with the ground truth (AustLII). As test set 1 considers only cases agreed by both, CaseBase has trivially a perfect performance. However, as we introduce the other citations in the test sets, the CaseBase human classifiers start missing D citations and incorrectly extracting non-D citations. CaseBase performance can be considered as an upper bound on the performance of the systems. It suggests that part of the errors made by the three methods comes from the intrinsic ambiguity of the class labels. In relation to this, we suggest that a by-product of the knowledge acquisition approach could be a better definition of class boundaries: during rule creation the expert views cases and selects reasons for them belonging to one class or the other, and in the process perhaps more carefully considers the correctness of the label. The fact that experts using RDR are more consistent in labeling was suggested by previous RDR studies (Wang, Boland, Graco, & He, 1996).

However, Table 5 also suggests that labeling inconsistency is not the only reason for the performance ceiling reached by the three systems, as there is a wide gap between the number of D citations correctly identified by the CaseBase experts and those identified by the automatic systems, both for true and false positives. The performance ceiling probably comes from the fact that the features used do not well reflect the actual reasons for the classifications: the presence or absence of a specific word in the paragraph in question will only indicate in some percentage of cases what the correct class is. Other cases have more complex reasons and in particular ML techniques using bag of words simply pick additional words which may have poor generalization qualities. It seems likely that to fully understand the complex relationships between two legal cases, a certain degree of reasoning or semantic understanding is needed, i.e. a certain degree of knowledge of the content of both cases is needed. This kind of reasoning on case content is not possible using our rule language or the vectorial representations of machine learning. Our results may indicate what is reasonable to expect by only a shallow processing of the text, without any kind of legal reasoning. However, our results seem to suggest that perhaps humans are better in building rules from a somewhat poor set of attributes than machine learning algorithms, which will pick attributes that are not well-aligned with the classification task to be solved but follow rather spurious patterns which happen to be present in the given training data set. The results where the machine learners were trained on all data suggest that this is still a problem even with relatively large training sets.

All the rules were made by one of the authors of this paper, who has not received any formal legal training. We tried to find an optimal balance between simplicity and expressiveness in creating the rule language, so that a legal expert could create rules. While the rules created employ simple terms, such as *different*, *unlike*, *not applicable* (see Fig. 2), we would expect that a legal expert might build different kinds of rules, using more semantic features rather than syntactic ones, and linking different elements in the text at the discourse level. This could improve the quality of the rules and thus further enhance the performance of the system. Unfortunately we did not have the resources to test this hypothesis but it is an opportunity for interesting further research.

Although the knowledge base outperforms the machine learning approaches, the difference is not very large, so that other factors will also influence the choice between the two approaches. The main advantage of our knowledge based approach is that, using RDR, the system can easily be extended with more rules. Intuitively if the KB was to be expanded the number of correctly identified cases would further increase, although the effect on the precision is not clear (see Fig. 4 which depicts the trend of precision and recall on the test set, with respect to the number of rules). While the rules can be improved in a controllable way, improving the results of a pure machine-learning system would require providing it with additional training data, rather than just individual cases; however, adding more data is of decreasing value, while the cost of manually annotating the data grows linearly. In our case labelled training cases were already available, but this may not be the case for situations where training data is scarce or need to be constructed by hand. In this scenario the expert time can be used to build an RDR KB as part of the annotation process, producing not only a knowledge base but more consistent annotation. Another advantage of using a rule base is that the manually acquired patterns are more accessible to human understanding.

It is difficult to compare the cost of building the two approaches. The preprocessing of cases was common to both approaches. We spent around 40 hours to build the knowledge base manually, and a similar time to prepare the data for machine learning and trying different optimizations (different algorithms, stemming, oversampling, different training sets). Thus, it is not clear which approach is better time-wise, even when an annotated dataset is available.

The question arises of whether using RDR has any advantage over other knowledge acquisition methods. It is difficult to answer this question in terms of proper quantitative comparative studies, because such studies are virtually non-existent in knowledge acquisition research. It is difficult for domain experts to find the time to build knowledge bases, let alone build repeat knowledge bases using different knowledge acquisition technologies. Using the same experts would also be problematic, as they will learn from their first experience. Consequently most research on comparative knowledge acquisition has been about understanding the different ways different techniques approach the same problem (Menzies & Van Harmelen, 1999; Schreiber & Birmingham, 1996; Shadbolt, O'hara, & Crow, 1999). These comparative studies have been valuable in understanding how different techniques apply to a problem, but not their relative performance. Zacharias (2008) provides some semi-quantitative data on knowledge acquisition from a survey of developers: overall the average time to build a knowledge base was 9 person months, and the average size was 1969 rules, with a median time of 5.5 person months and median size of 120 rules. This data represents a number of different knowledge acquisition techniques, so one cannot identify specific methods. One can make an informal comparison with RDR from (Compton, Peters, Lavers, & Kim, 2011) where the median time to add a rule and debug the knowledge base was 78 seconds and the average 136 seconds. This data comes from logs from about 55,000 rules added to many knowledge bases, some with thousands of rules. The data from these studies is not sufficient for making a proper comparison; however, the data is strongly suggestive that for rule addition and debugging (i.e. excluding previous cases being misclassified) RDR compares very favorably with other techniques, particularly since both Zacharias' paper and an earlier survey by O'Leary (1991) emphasize that the main difficulty knowledge engineers have is in debugging knowledge bases, the central strength of RDR.

8. Automatic synonyms expansion

At the end of the knowledge acquisition, we experimented with automatically expanding synonyms of words in the rules, to increase the number of matches of the rule set. During the knowledge acquisition phase the user is already presented with suggested synonyms (see the example in Table 4), here we try to automatically insert all synonyms in the rules, hoping to increase the recall on unseen data. To obtain synonyms for a given rule, we look at all terms present in the rule, then we take synonyms from all other rules using two methods:

- We take all synonyms existing in any other rule, where the synonym relation is defined by the LOIS and WordNet databases.
- If in any other rule the word is present in a disjunction of several terms, those terms are deemed to be synonyms of the words. For example if in one rule we have a disjunction (*issue* | *facts*), the words "*facts*" and "*issue*" are considered synonyms.

Possible synonyms are only the words existing in any other rule (no new words are considered). Our system finds all the possible synonyms of any word in the rules with these two methods, and then adds these terms as disjunctions to the original one, creating thus a new set of rules (each rule will be the same or become more general). Overall the following expansions were inserted (only the group of synonyms which modify at least one rule are listed):

[*judgment*, *decision*]
[*bear*, *support*]
[*fact*, *matter*, *issue*, *case*]
[*different*, *unlike*, *dissimilar*, *contrary*, *distinguishable*]
[*pertinent*, *relevance*, *applicable*, *relevant*, *apposite*]
[*address*, *reference*]
[*just*, *simply*, *solely*, *only*, *good*, *merely*]
[*argue*, *submit*, *seek*]
[*help*, *assist*]
[*provide*, *offer*]

This means that, for example, wherever the word *judgement* is present in a rule, it substituted with the disjunction (*judgement* | *decision*) and vice versa.

Running the set of modified rules on our training corpus, we found an increase in recall, but at the same time the precision was reduced. Most of the rules have more matches, but the number of incorrect new matches is always higher than the number of correct new matches. Considering all the rules together, the recall (with respect to the Distinguished class) increases by just 3 cases (1.3%) while the precision drops by 16.3% (54 new cases incorrectly classified). The results are given in Table 7.

We expected not to get better results for the training set, as when we built the rules these synonyms were already tested on the corpus (i.e. if a synonym would increase performance, the system would have indicated it and the user would have added it to the rule). To check the performance of the new rules over unseen data (to measure if we can increase the generalization power of the rules) we tested the new rules on our test corpus. On the test dataset the new rules have a precision of 55.1% and recall 52.8% (compared to 70.0% and 48.6% for the original rule set) with the F-Measure relative to the Distinguished class going from 57.4% to 53.9%. Only 3 new cases are correctly identified (by rule number 54), while 16 new cases are incorrectly classified (8 of which are from rule 54). The pattern for rule 54 was:

[toBE] *distinguishable* − > Distinguished.

The expanded rule 54 is:

[toBE] (*distinguishable* | *different* | *unlike* | *dissimilar* | *contrary*) − > Distinguished

which appears to be too general and matches more incorrect cases than the original rule.

Table 7
Comparison with automatic synonym expansion.

	Training set		Test set	
	LEXA	LEXAsyn	LEXA	LEXAsyn
Precision(D)	0.84	0.68	0.70	0.55
Recall (D)	0.81	0.83	0.49	0.53
F-measure (D)	0.83	0.74	0.57	0.54
Precision (FA)	0.96	0.96	0.87	0.87
Recall (FA)	0.97	0.91	0.94	0.88
F-measure (FA)	0.96	0.94	0.90	0.87
Accuracy	0.94	0.90	0.84	0.80
D correctly extracted	183	186	35	38
FA extracted as D	35	89	15	31

Even in cases where on manual inspection the synonym expansion looks intuitive or reasonable, we did not find any performance increase. For example rule 33 which was:

no relevance [GAP8] (*circumstances* | [REFCASE]) − > Distinguished

became:

no (*relevance* | *pertinent* | *applicable* | *relevant* | *apposite*) [GAP8] (*circumstances* | [REFCASE]) − > Distinguished

and produces some incorrect matches, for example:

In Wang, as here, futility is not to be judged by asking what decision the Tribunal would have made even if it had complied with s 359A of the Act or by asking if a decision by the Tribunal in the future to refuse the application were inevitable because there has been ***no relevant change of circumstances****.*

These results suggest that automatic synonym expansion, without the supervision of a human expert, introduces too many errors. An explanation for this may be the difficulty in obtaining high quality synonyms from the lexicon used, or in the lack of sense disambiguation. On the other hand the human during the KA phase can benefit from having synonyms suggested that they may not have thought of, but can use their own judgement to decide if the synonym is appropriate.

We could not evaluate the supervised synonym suggestion component, as no rules were built without it. To get an idea of how much the synonym suggestion component, supervised by the expert, impacts the result, we can compare our results with those presented in Galgani and Hoffmann (2010). In that previous experiment we built a small knowledge base not using any synonym suggestion. The results were inferior to those presented here: precision and recall for the D class were 67.4% and 40.3%, respectively, but the performance gap may also depend on other factors.

These results are consistent with the finding of Roth and Small (2009) in a entity relation extraction task: the authors concluded that abstracting lexical features to what they call semantically related word lists, can improve the classification, but only if the process is supervised by an expert. Automatic expansion did not give a performance increase due to increased ambiguity. A difference from our work is that in that study the expert was not presented with statistics from the corpus to guide his selection, relying only on their intuition.

9. Automatic exception suggestion

In order to reduce the expert knowledge acquisition effort, we experimented with a module to aid in the creation of exception rules by suggesting terms to be included in rules. When we create a first level rule there is no guidance as to which part of the text is relevant, but when we create an exception rule, the case has already satisfied the parent rule, so that the most significant words are probably around the existing CLASS annotation from the parent rule. This suggests that if a number of cases have incorrectly satisfied the parent rule, it might be useful to automatically check which words appear around the existing CLASS annotation to find common patterns. In particular, for a given rule, we look at the occurrences of each word in three contexts: words preceding the annotation, words inside the annotation (but not specified by the rule, i.e. in the gaps), and words following the annotation. If a word appears more often in one class (D or FA) than in the other, we propose it as suggestion to the user to build an exception rule.

The system counts the number of word occurrences in the three contexts (and their union) for the two different classes (FA and D), considering 50 words on the left and 50 on the right from the match. It then ranks the list of words according to the differences between the occurrences of the two classes (employing also a minimum threshold on the ratio). We also experimented with stemming all the words, looking only at particular word categories (i.e. nouns or verbs) and putting together synonyms. The user inspects the list and can test some words to look at specific matches.

The exception suggestion module was used successfully to create exceptions for Rule 1, which is the rule with the highest number of incorrect matches. However for the rest of the rules we did not find the term list very useful. We believe that this is due to the fact the all the other rules have a smaller number of incorrect cases associated with them so there are not enough examples to collect meaningful statistics and suggest meaningful terms. Rule 1 on the other hand had 45 incorrect matches. Although not very applicable here, apart from rule 1, we believe that this procedure could suggest useful terms in other situations where there were many incorrectly classified cases for a node. Rule 1 is:

> [CASE] ([Split])? (*distinguished* | *distinguishable*) − > Distinguished.

where ([Split])? indicates that the two parts must be in the same or adjacent sentences. Among all terms, the system came up with some good suggestions that were turned into a rule, for example:

- **submitted** (context=preceding) cases: FA = 10 D = 1.
  Relevant example:

  > *Mr. Wood, who appears for the applicant, **submitted** that the decision of* [CASE ***Giles J***]***is distinguishable***. *He drew attention to what appears at 259 to 260 where his Honour observed that he did not think that Pt 36 rule 16 of the Supreme Court Rules should be read down and that it empowers the service of a notice to produce on a foreign party. However, I do not consider those observations of Giles J bear upon the issue before me this morning.*

- **however** (context=following) cases: FA = 8 D = 0.
  Relevant example:

  > [CASE ***David Grant***] *and the other cases, involving as they did statutory provisions concerning applications to a Court, may be **distinguished** from cases such as the present, on the basis that conformity with the procedural time limit was a pre-condition to the jurisdiction of the Court from which relief was claimed:* [CASE *David Grant at 276–277*]; [CASE *Emanuele at 152 per Kirby J*]; [CASE *Newtronics at [25]*]. ***However***, that distinction does not delineate the limits of the present enquiry.

- Token.stem=**appli** (context=union) cases: FA = 9 D = 2.
  Relevant example:

  > *According to the **applicant**,* [CASE ***Lodge***] ***is distinguishable*** *because "a person does not earn income by having their child looked after but in the case of Youth Allowance a person does earn income through study". Furthermore, it was submitted, this was not a case in which the applicant was able to choose the manner and form in which she performed the activity from which her income was derived.*

- And the list of synonyms: **figure** (FA = 1 D = 0) **pattern** (FA = 1 D = 0) **shape** (FA = 5 D = 0) **design** (FA = 3 D = 0).
  Relevant example:

  > *The reasons of the Court in* [CASE ***Koninklijke Philips Electronics NV v Remington Products Australia Pty Ltd [2000] FCA 876; 100 FCR 90***] *make clear that: (a) Very often, the concept of use as a trade mark in relation to goods involves the physical application of the trade mark to the goods (e.g. by stamping or embossing thereon the manufacturer's name or logo) in order to state the origin of the goods for all to see. However, goods may be **distinguished** by other visible characteristics which they have (e.g. their **shape** or color); . . .*

Note that we also have several matches with high scores but that appear to be the result of coincidence:

- **law** (context=following) cases: FA = 15 D = 1.
- **FCAFC** (context=preceding) cases: FA = 10 D = 2.
- **later** (context=union) cases: FA = 11 D = 1.

The system relies on the judgement of the user who inspects the list and decides which term should be used to make a rule. For example looking at the list, the following exception rule was created:

> [Token.stem=*submit*] [GAP5] [CLASS.Distinguished] − > Followed/Applied

which gave the correct classification, among the others, for the case:

> *Mr. Gould **submits** that* [CLASS.Distinguished [CASE ***Sleiman***]] *and a case relied on in it,* [CLASS.Distinguished [CASE *Re Wharton and Australian Securities and Investments Commission [2002] AATA 443; (2002) 69 ALD 419*]], *were either wrongly decided or should be **distinguished**.*

10. Conclusion

In this paper we describe our approach to legal citation classification, based on efficient knowledge acquisition. Citations are an important aspect of most judicial decisions, and their correct classification, characterizing the relation between the two cases, is very useful for legal research. Classified citations can also aid a range of other applications such as information retrieval and automatic summarization (Galgani et al., 2015).

Our system LEXA supports the creation of a knowledge base to identify Distinguished citations, using the efficient RDR methodology that strongly supports the creation of rules by a user. The RDR structure of the knowledge base makes it easy to patch the rules whenever an error is found, and the powerful rule language we designed, based on regular expressions of annotations, enables the identification of patterns at different levels. Given the availability of an annotated dataset, we extended the traditional RDR approach to provide automatic support during rule acquisition. The support includes: (i) real time feedback on the performance of the rule on the whole dataset (as opposed to checking only cornerstone cases); (ii) assisting manual validation of the rule by retrieving unseen relevant examples (as opposed to only cornerstone cases already seen by the experts); (iii) suggestion of generalizations and synonyms automatically validated on the corpus; (iv) exception re-use guided by automatic suggestion and validation. We also experimented with automatic synonym expansion and the suggestion of new exception rules. These methods enabling powerful automatic support, built on top of RDR for situations where an annotated dataset is available, are the main contribution of this work, as they extend traditional expert-systems methodologies.

A second major contribution is a comparison of the performance obtained with knowledge acquisition and that attained with machine learning methods. We used LEXA to build a knowledge base of 78 rules that recognize the important Distinguished (D) citations with a precision of 70% and recall of 48.6% on the cleaned test set. The knowledge base outperformed our best machine learning model considerably on recall (precision of 70%

and recall of 38.9%). We also tested the methods on two different noisier versions of the dataset, and found that the performance gap between LEXA and machine learning widens. For example, when selecting D citations from a pool of citations of different classes (as opposed to using only FA as negative class) LEXA outperforms the best machine learner by 78% on the F-score (0.213 vs 0.120), identifying more D citations and less non-D citations. While it is not clear which approach requires less expert time, other advantages of our RDR-based approach compared to machine learning are:

i. the possibility of easily extending the knowledge base, even without annotated data;
ii. the readability of the created rules, which provide a more transparent system;
iii. the fact that the expert time can be used directly to label more data and
iv. the possibility of clarifying labels during knowledge acquisition.

The performance ceiling reached by both methods seems to come in part from the inconsistency of class labels and in part from the difficulty of the task which requires some kind of legal reasoning.

In conclusion, our work demonstrates that the incremental approach of RDR, given a suitable rule language, can be used to tackle difficult NLP problems such as classifying legal citations. A key insight is using the available dataset to guide the manual acquisition of rules.

In future work we would like to build a better interface for rule creation to allow rule acquisition from legal experts directly, in order to have legal experts directly build a more comprehensive knowledge base. Ideally it should be as easy for a legal expert to create rules as it was for the authors. We also would expect that involving legal expertise would increase the quality of the rules, adding a 'legal' semantic level to the rules (such as terms or constructs specific to the legal language).

We are also extending this approach to other legal text processing tasks and are building an automatic summarization system, using a knowledge base of rules that combine different kinds of attributes, including citation information (Galgani, Compton, & Hoffmann, 2014).

Acknowledgments

The authors would like to thank Philip Chung, Andrew Mowbray and Graham Greenleaf from AustLII for their valuable help and constructive comments.

References

Athar, A. (2011). Sentiment analysis of citations using sentence structure-based features. In *HLT-SS '11 proceedings of the ACL 2011 student session* (pp. 81–87). Association for Computational Linguistics.

Athar, A., & Teufel, S. (2012). Context-enhanced citation sentiment detection. In *Proceedings of the 2012 conference of the north American chapter of the association for computational linguistics: human language technologies* (pp. 597–601). Association for Computational Linguistics.

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with python*. O'Reilly Media.

Compton, P., & Jansen, R. (1990). Knowledge in context: A strategy for expert system maintenance. In *AI '88: Proceedings of the second Australian joint conference on artificial intelligence* (pp. 292–306). New York, NY, USA: Springer-Verlag New York, Inc..

Compton, P., Peters, L., Lavers, T., & Kim, Y. S. (2011). Experience with long-term knowledge acquisition. In *Proceedings of the sixth international conference on knowledge capture* (pp. 49–56). New York, NY, USA: ACM.

Conrad, J. G., & Dabney, D. P. (2001). Automatic recognition of distinguishing negative indirect history language in judicial opinions. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management* (pp. 287–294). New York, NY, USA: ACM.

Cunningham, H., Maynard, D., Bontcheva, K., & Tablan, V. (2002). Gate: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th anniversary meeting of the association for computational linguistics (ACL'02), Philadelphia* (pp. 168–175).

de Maat, E., Winkels, R., & van Engers, T. (2006). Automated detection of reference structures in law. *Frontiers in Artificial Intelligence and Applications*, 41.

Ding, Y., Zhang, G., Chambers, T., Song, M., Wang, X., & Zhai, C. (2014). Content-based citation analysis: The next generation of citation analysis. *Journal of the Association for Information Science and Technology, 65*, 1820–1833.

Edwards, G., Compton, P., Malor, R., Srinivasan, A., & Lazarus, L. (1993). PEIRS: A pathologist-maintained expert system for the interpretation of chemical pathology reports. *Pathology, 25*, 27–34.

Elkiss, A., Shen, S., Fader, A., Erkan, G., States, D., & Radev, D. (2008). Blind men and elephants: What do citation summaries tell us about a research article? *Journal of the American Society for Information Science and Technology, 59*, 51–62.

Farzindar, A., & Lapalme, G. (2009). Machine translation of legal information and its evaluation. In *Advances in artificial intelligence* (pp. 64–73). Springer.

Galgani, F., Compton, P., & Hoffmann, A. (2012). Towards automatic generation of catchphrases for legal case reports. In *The 13th international conference on intelligent text processing and computational linguistics* (pp. 415–426). New Delhi, India: Springer Berlin Heidelberg.

Galgani, F., Compton, P., & Hoffmann, A. (2014). Hauss: Incrementally building a summarizer combining multiple techniques. *International Journal of Human–Computer Studies, 72*, 584–605.

Galgani, F., Compton, P., & Hoffmann, A. (2015). Summarization based on bi-directional citation analysis. *Information Processing & Management, 51*, 1–24.

Galgani, F., & Hoffmann, A. (2010). Lexa: Towards automatic legal citation classification. In J. Li (Ed.), *AI 2010: Advances in artificial intelligence* (pp. 445–454). Springer Berlin Heidelberg.

Gotti, F., Farzindar, A., Lapalme, G., & Macklovitch, E. (2008). Automatic translation of court judgments. In *AMTA'2008 the eighth conference of the association for machine translation in the Americas, Waikiki, Hawai'i* (pp. 1–10).

Greenleaf, G., Mowbray, A., King, G., & Van Dijk, P. (1995). Public access to law via internet: The Australian legal information institute. *Journal of Law and Information Science, 6*, 49.

Hachey, B., & Grover, C. (2006). Extractive summarisation of legal texts. *Artificial Intelligence and Law, 14*, 305–345.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The weka data mining software: An update. *SIGKDD Exploration Newsletter, 11*, 10–18.

Hoffmann, A., & Pham, S. B. (2003). Towards topic-based summarization for interactive document viewing. In *K-CAP '03: Proceedings of the second international conference on knowledge capture* (pp. 28–35). New York, NY, USA: ACM.

Ho, V., Wobcke, W., & Compton, P. (2003). Emma: An e-mail management assistant. In *Proceedings of the IEEE/WIC international conference on intelligent agent technology* (pp. 67). Washington, DC, USA: IEEE Computer Society.

Kaplan, D., Iida, R., & Tokunaga, T. (2009). Automatic extraction of citation contexts for research paper summarization: A coreference-chain based approach. In *NLPIR4DL '09: Proceedings of the 2009 workshop on text and citation analysis for scholarly digital libraries* (pp. 88–95). Morristown, NJ, USA: Association for Computational Linguistics.

Kim, M., & Compton, P. (2004). Evolutionary document management and retrieval for specialized domains on the web. *International Journal of Human–Computer Studies, 60*, 201–241.

Kim, M., & Compton, P. (2006). The perceived utility of standard ontologies in document management for specialized domains. *International Journal of Human–Computer Studies, 64*, 15–26.

Kim, M. H., Compton, P., & Kim, Y. S. (2011). Rdr-based open ie for the web document. In *Proceedings of the sixth international conference on knowledge capture* (pp. 105–112). New York, NY, USA: ACM.

Krzywicki, A., & Wobcke, W. (2009). Incremental e-mail classification and rule suggestion using simple term statistics. In *Proceedings of the 22nd Australasian joint conference on advances in artificial intelligence* (pp. 250–259). Berlin, Heidelberg: Springer-Verlag.

Martínez-González, M., de la Fuente, P., & Vicente, D. J. (2005). Reference extraction and resolution for legal texts. In *Pattern recognition and machine intelligence* (pp. 218–221). Springer.

Mei, Q., & Zhai, C. (2008). Generating impact-based summaries for scientific literature. In *Proceedings of ACL-08: HLT* (pp. 816–824).

Menzies, T., & Van Harmelen, F. (1999). Editorial: Evaluating knowledge engineering techniques. *International Journal of Human–Computer Studies, 51*, 715–727.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM, 38*, 39–41.

Moens, M. F. (2001). Innovative techniques for legal text retrieval. *Artificial Intelligence and Law, 9*, 29–57.

Moens, M. F. (2007). Summarizing court decisions. *Information Processing & Management, 43*, 1748–1764.

Moens, M. F., & Angheluta, R. (2003). Concept extraction from legal cases: The use of a statistic of coincidence. In *ICAIL '03: Proceedings of the ninth international conference on artificial intelligence and law* (pp. 142–146). New York, NY, USA: ACM.

Mohammad, S., Dorr, B., Egan, M., Hassan, A., Muthukrishan, P., Qazvinian, V., Radev, D., & Zajic, D. (2009). Using citations to generate surveys of scientific paradigms. In *Proceedings of human language technologies: The 2009 annual conference of the north American chapter of the association for computational linguistics, Boulder, Colorado* (pp. 584–592).

Mowbray, A., Chung, P., & Greenleaf, G. (2009). Free-access case law enhancements for australian law. In *Free access, quality of information, effectiveness of rights (Proc. IX international conference 'law via the internet')* (pp. 285–300). Florence, Italy: European Press Academic Publishing.

Murphy, J. E., Steele, R., & Shen, R. (2008). Exploiting the rich document structures and network topology of legal information systems. In *PACIS 2008 Proceedings* (pp. 234–244).

Nakov, P. I., Schwartz, A. S., & Hearst, M. A. (2004). Citances: Citation sentences for semantic analysis of bioscience text. In *Proceedings of the SIGIR'04 workshop on search and discovery in bioinformatics* (pp. 81–88).

Nanba, H., & Okumura, M. (1999). Towards multi-paper summarization using reference information. In *IJCAI '99: Proceedings of the sixteenth international joint conference on artificial intelligence* (pp. 926–931). San Francisco, CA, USA: Morgan Kaufman Publishers Inc..

O'Leary, D. E. (1991). Design, development and validation of expert systems: A survey of developers. In *Validation, verification and test of knowledge-based systems* (pp. 3–19). John Wiley & Sons, Inc..

Palau, R. M., & Moens, M. F. (2009). Argumentation mining: The detection, classification and structure of arguments in text. In *ICAIL '09: Proceedings of the 12th international conference on artificial intelligence and law* (pp. 98–107). New York, NY, USA: ACM.

Palmirani, M., Brighi, R., & Massini, M. (2003). Automated extraction of normative references in legal texts. In *Proceedings of the ninth international conference on artificial intelligence and law* (pp. 105–106). ACM.

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval, 2*, 1–135.

Peters, W., Sagri, M. T., & Tiscornia, D. (2007). The structuring of legal knowledge in LOIS. *Artificial Intelligence and Law, 15*, 117–135.

Pham, S., & Hoffmann, A. (2005). Efficient knowledge acquisition for extracting temporal relations. In *Proceedings of the Australasian language technology workshop* (pp. 87–95).

Pham, S. B., & Hoffmann, A. (2003). A new approach for scientific citation classification using cue phrases. In *AI 2003: Advances in artificial intelligence* (pp. 759–771). Springer.

Pham, S. B., & Hoffmann, A. (2004). Incremental knowledge acquisition for building sophisticated information extraction systems with kaftie. In *Fifth international conference on practical aspects of knowledge management* (pp. 292–306). Springer-Verlag.

Piao, S., Ananiadou, S., Tsuruoka, Y., Sasaki, Y., & McNaught, J. (2007). Mining opinion polarity relations of citations. In *International workshop on computational semantics (IWCS)* (pp. 366–371).

Porter, M. (1980). An algorithm for suffix stripping. *Program, 14*, 130–137.

Posner, R. A. (1999). The theory and practice of citations analysis, with special reference to law and economics. University of Chicago Law School John M. Olin Law and Economics Working Paper No. 83.

Qazvinian, V., & Radev, D. R. (2008). Scientific paper summarization using citation summary networks. In *Proceedings of the 22nd international conference on computational linguistics (Coling 2008)* (pp. 689–696).

Qazvinian, V., Radev, D. R., & Ozgur, A. (2010). Citation summarization through keyphrase extraction. In *Proceedings of the 23rd international conference on computational linguistics (Coling 2010), Coling 2010 organizing committee, Beijing, China* (pp. 895–903).

Qazvinian, V., & Radev, D. R. (2010). Identifying non-explicit citing sentences for citation-based summarization. In *ACL '10: Proceedings of the 48th annual meeting of the association for computational linguistics* (pp. 555–564). Morristown, NJ, USA: Association for Computational Linguistics.

Rennie, J. D., Shih, L., Teevan, J., & Karger, D. R. (2003). Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of the twentieth international conference on machine learning* (pp. 616–623).

Richards, D. (2009). Two decades of ripple down rules research. *The Knowledge Engineering Review, 24*, 159–184.

Ritchie, A., Robertson, S., & Teufel, S. (2008). Comparing citation contexts for information retrieval. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management* (pp. 213–222). New York, NY, USA: ACM.

Ritchie, A., Teufel, S., & Robertson, S. (2006). How to find better index terms through citations. In *Proceedings of the workshop on how can computational linguistics improve information retrieval?* (pp 25–32). Sydney, Australia: Association for Computational Linguistics.

Roth, D., & Small, K. (2009). Interactive feature space construction using semantic information. In *Proceedings of the thirteenth conference on computational natural language learning* (pp. 66–74).

Ruiz-Sánchez, J. M., Valencia-Garcıa, R., Fernández-Breis, J. T., Martınez-Béjar, R., & Compton, P. (2003). An approach for incremental knowledge acquisition from text. *Expert Systems with Applications, 25*, 77–86.

Schreiber, A. T., & Birmingham, W. P. (1996). Editorial: The sisyphus-vt initiative. *International Journal of Human–Computer Studies, 44*, 275–280.

Shadbolt, N., O'hara, K., & Crow, L. (1999). The experimental evaluation of knowledge acquisition techniques and methods: History, problems and new directions. *International Journal of Human–Computer Studies, 51*, 729–755.

Teufel, S., Siddharthan, A., & Tidhar, D. (2006). Automatic classification of citation function. In *Proceedings of the 2006 conference on empirical methods in natural language processing* (pp. 103–110). Sydney, Australia: Association for Computational Linguistics.

Valencia-Garcıa, R., Ruiz-Sánchez, J. M., Vivancos-Vicente, P. J., Fernández-Breis, J. T., & Martınez-Béjar, R. (2004). An incremental approach for discovering medical knowledge from texts. *Expert Systems with Applications, 26*, 291–299.

van Opijnen, M. (2010). Canonicalizing complex case law citations. In *Proceedings of the 2010 conference on legal knowledge and information systems: JURIX 2010* (pp. 97–106).

Wang, J., Boland, M., Graco, W., & He, H. (1996). Use of ripple-down rules for classifying medical general practitioner practice profiles repetition. In *Proceedings of Pacific Knowledge Acquisition Workshop PKAW* (pp. 23–25).

Xu, H., Martin, E., & Mahidadia, A. (2013). Using heterogeneous features for scientific citation classification. In *PACLING: Conference of the Pacific association for computational linguistics*.

Xu, H., Martin, E., & Mahidadia, A. (2014). Exploiting paper contents and citation links to identify and characterize specialisations. In *2014 IEEE international conference on data mining workshops, ICDM workshops 2014, Shenzhen, China, December 14, 2014* (pp. 613–620).

Xu, H., & Hoffmann, A. (2010). Rdrce: Combining machine learning and knowledge acquisition. In B. H. Kang & D. Richards (Eds.), *Knowledge management and acquisition for smart systems and services* (pp. 165–179). Berlin/Heidelberg: Springer.

Yousfi-Monod, M., Farzindar, A., & Lapalme, G. (2010). Supervised machine learning for summarizing legal documents. In *Canadian conference on artificial intelligence 2010* (pp. 51–62). Ottawa, Canada: Springer.

Zacharias, V. (2008). Development and verification of rule based systems – a survey of developers. In *Rule representation, interchange and reasoning on the web* (pp. 6–16). Springer.

Zhang, P., & Koppaka, L. (2007). Semantics-based legal citation network. In *ICAIL '07: Proceedings of the 11th international conference on artificial intelligence and law* (pp. 123–130). New York, NY, USA: ACM.

Zhu, X., Turney, P., Lemire, D., & Vellino, A. (2015). Measuring academic influence: Not all citations are equal. *Journal of the Association for Information Science and Technology, 66*, 408–427.