Combining Different Summarization Techniques for Legal Text

Filippo Galgani **Paul Compton** **Achim Hoffmann**
School of Computer Science and Engineering
The University of New South Wales
Sydney, Australia
{galganif,compton,achim}@cse.unsw.edu.au

Abstract

Summarization, like other natural language processing tasks, is tackled with a range of different techniques - particularly machine learning approaches, where human intuition goes into attribute selection and the choice and tuning of the learning algorithm. Such techniques tend to apply differently in different contexts, so in this paper we describe a hybrid approach in which a number of different summarization techniques are combined in a rule-based system using manual knowledge acquisition, where human intuition, supported by data, specifies not only attributes and algorithms, but the contexts where these are best used. We apply this approach to automatic summarization of legal case reports. We show how a preliminary knowledge base, composed of only 23 rules, already outperforms competitive baselines.

1 Introduction

Automatic summarization tasks are often addressed with statistical methods: a first type of approach, introduced by Kupiec et al.(1995), involves using a set of features of different types to describe sentences, and supervised learning algorithms to learn an empirical model of how those features interact to identify important sentences. This kind of approach has been very popular in summarization; however the difficulty of this task often requires more complex representations, and different kinds of models to learn relevance in text have been proposed, such as discourse-based (Marcu, 1997) or network-based (Salton et al., 1997) models and many others. Domain knowledge usually is present in the choice of features and algorithms, but it is still an open issue how best to capture the domain knowledge required to identify what is relevant in the text; manual approaches to build knowledge bases tend to be tedious, while automatic approaches require large amounts of training data and the result may still be inferior.

In this paper we present our approach to summarize legal documents, using knowledge acquisition to combine different summarization techniques. In summarization, different kinds of information can be taken in account to locate important content, at the sentence level (e.g. particular terms or patterns), at the document level (e.g. frequency information, discourse information) and at the collection level (e.g. document frequencies or citation analysis); however, the way such attributes interact is likely to depend on the context of specific cases. For this reason we have developed a set of methods for identifying important content, and we propose the creation of a Knowledge Base (KB) that specifies which content should be used in different contexts, and how this should be combined. We propose to use the Ripple Down Rules (RDR) (Compton and Jansen, 1990) methodology to build this knowledge base: RDR has already proven to be a very effective way of building KBs, had has been used successfully in several NLP task (see Section 2). This kind of approach differs from the dominant supervised learning approach, in which we first annotate text to identify relevant fragments, and then we use supervised learning algorithms to learn a model; one example in the legal domain being the work of Hachey and Grover (2006). Our approach eliminates the need for separate manual annotation of text, as the rules are built by a human who judges the relevance of text and directly creates the set of rules as the one process, rather than annotating the text and then separately tuning the learning model.

We apply this approach to the summarization of legal case reports, a domain which has an increasing need for automatic text processing, to cope with the large body of documents that is case law.

Table 1: Examples of catchphrases list for two cases.

COSTS - proper approach to admiralty and commercial litigation - goods transported under bill of lading incorporating Himalaya clause - shipper and consignee sued ship owner and stevedore for damage to cargo - stevedore successful in obtaining consent orders on motion dismissing proceedings against it based on Himalaya clause - stevedore not furnishing critical evidence or information until after motion filed - whether stevedore should have its costs - importance of parties cooperating to identify the real issues in dispute - duty to resolve uncontentious issues at an early stage of litigation - stevedore awarded 75% of its costs of the proceedings
MIGRATION - partner visa - appellant sought to prove domestic violence by the provision of statutory declarations made under State legislation - ”statutory declaration” defined by the Migration Regulations 1994 (Cth) to mean a declaration ”under” the Statutory Declarations Act 1959 (Cth) in Div 1.5 - contrary intention in reg 1.21 as to the inclusion of State declarations under s 27 of the Acts Interpretation Act - statutory declaration made under State legislation is not a statutory declaration ”under” the Commonwealth Act - appeal dismissed

Countries with “common law” traditions, such as Australia, the UK and the USA, rely heavily on the concept of precedence: on how the courts have interpreted the law in individual cases, in a process that is known as *stare decisis* (Moens, 2007), so legal professionals: lawyers, judges and scholars, have to deal with large volumes of past court decisions.

Automatic summarization can greatly enhance access to legal repositories; however, legal cases, rather than summaries, often contain lists of catchphrases: phrases that present the important legal points of a case. The presence of catchphrases can aid research of case law, as they give a quick impression of what the case is about: “*the function of catchwords is to give a summary classification of the matters dealt with in a case. [...] Their purpose is to tell the researcher whether there is likely to be anything in the case relevant to the research topic*” (Olsson, 1999). For this reason, rather than constructing summaries, we aim at extracting catchphrases from the full text of a case report. Examples of catchphrases from two case reports are shown in Table 1.

In this paper we present our approach towards automatic catchphrase extraction from legal case reports, using a knowledge acquisition approach according to which rules are manually created to combine a range of diverse methods to locate catchphrase candidates in the text.

2 Related Work

Different kinds of language processing have been applied to the legal domain, for example, automatic summarization, retrieval (Moens, 2001), machine translation (Farzindar and Lapalme, 2009), and citation analysis (Zhang and Koppaka, 2007; Galgani and Hoffmann, 2010). Among these tasks, the most relevant to catchphrase extraction is the work on automatic summarization, with the difference that catchphrases usually cover many dimensions of one case, giving a broader representation than summaries. Examples of automatic summarization systems developed for the legal domain are the work of Hachey and Grover (Hachey and Grover, 2006) to summarize the UK House of Lords judgements, and PRODSUM (Yousfi-Monod et al., 2010), a summarizer of case reports for the CanLII database (Canadian Legal Information Institute) (see also (Moens, 2007) for an overview). Both systems rely on supervised learning algorithms, using sentences tagged as important to learn how to recognize important sentences in the text: in this case the domain knowledge is incorporated mainly in the choice of features. This contrasts with our approach where the human intuition goes also in the weights given to different attributes in different contexts.

Ripple Down Rules

As we propose to use rules manually created for specifying how to identify relevant text, our approach is based on incremental Knowledge Acquisition (KA). A KA methodology which has already been applied to language processing tasks is Ripple Down Rules (RDR) (Compton and Jansen, 1990). In RDR, rules are created by domain experts without a knowledge engineer, the knowledge base is built with incremental refinements from scratch, while the system is in use; the domain expert monitors the system and whenever it performs incorrectly he or she flags the error and provides a rule based on the case which generated the error, which is added to the knowledge base and corrects the error. 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 exception rule structure.

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 all cases previously correctly classified, so that unwanted indirect effects of rule interactions are avoided (Compton and Jansen, 1990). The manual creation of rules, in contrast with machine learning, requires a smaller quantity of annotated data, as the human in the loop can identify the important features in a single case, whereas learning techniques require multiple instances to identify important features.

RDR have been used to tackle natural language processing tasks with the system KAFTIE (Pham and Hoffmann, 2004) (for summarization in (Hoffmann and Pham, 2003)). Knowledge bases built with RDR were shown to outperforms machine learning in legal citation analysis (2010) and in open information extraction (Kim et al., 2011); while Xu and Hoffmann (2010) showed how a knowledge base automatically built from data can be improved using manual knowledge acquisition from a domain expert with RDR.

3 Dataset

We use as the source of our data the legal database AustLII[1], the Australasian Legal Information Institute (Greenleaf et al., 1995), one of the largest sources of legal material on the net, which provides free access to reports on court decisions in all major courts in Australia.

We created an initial corpus of 2816 cases accessing case reports from the Federal Court of Australia, for the years 2007 to 2009, for which author-made catchphrases are given and extracted the full text and the catchphrases of every document. Each document contains on average 221 sentences and 8.3 catchphrases. In total we collected 23230 catchphrases, of which 15359 (92.7%) were unique, appearing only in one document in the corpus. These catchphrases are used to evaluate our extracts using Rouge, as described in Section 4.

To have a more complete representation of these cases, we also included citation information. Citation analysis has proven to be very useful in automatic summarization (Mei and Zhai, 2008; Qazvinian and Radev, 2008). We downloaded citation data from LawCite[2]. It is a service provided by AustLII which, for a given case, lists cited cases and more recent cases that cite the case. We downloaded the full texts and the catchphrases (where available) from AustLII, of both cited (previous) cases and more recent cases that cite the current one (citing cases). Of the 2816 cases, 1904 are cited at least by one other case (on average by 4.82 other cases). We collected the catchphrases of these citing cases, searched the full texts to extract the location where a citation is explicitly made, and extracted the containing paragraph(s). For each of the 1904 cases we collected on average 21.17 citing sentences, and we extracted an average of 35.36 catchphrases (from one or more other documents). From previous cases referenced by the judge, we extracted on average 67.41 catchphrases for each case.

We also extracted, using LawCite, references to any type of legislation made in the report. We located in the full text the sentences where each section or Act is mentioned; then we accessed the full texts of the legislation on AustLII, and extracted the title of the sections (for example, if section 477 is mentioned in the text, we extract the corresponding title: *CORPORATIONS ACT 2001 - SECT 477 Powers of liquidator*).

Our dataset thus contains the initial 2816 cases with given catchphrases, and all cases related to them by incoming or outgoing citations, with catchphrases and citing sentences explicitly identified, and the references to Acts and sections of the law.

4 Evaluation method

As it was not reasonable to involve legal experts in this sort of exploratory study, we looked for a simple way to evaluate candidate catchphrases automatically by comparing them with the author-made catchphrases from our AustLII corpus (considered as our "gold standard"), to quickly assess the performances of various methods on a large number of documents. As our system extracts sentences from text as candidate catchphrases, we propose an evaluation method which is based on Rouge (Lin, 2004) scores between extracted sentences and given catchphrases. This method was used also in (Galgani et al., 2012). Rouge includes several measures to quantitatively compare system-generated summaries to human-generated summaries, counting the number of overlapping n-grams of various lengths, word pairs and word sequences between two or more summaries.

Somewhat different from the standard use of Rouge (which would involve comparing the whole block of catchphrases to the whole block of extracted sentences), we evaluated extracted sentences individually so that the utility of any one catchphrase is minimally affected by the others, or by their particular order. On the other hand we want to extract sentences that contain an entire individual catchphrase, while a sentence that

[1]http://www.austlii.edu.au/
[2]http://www.lawcite.org

contains small pieces of different catchphrases is not as useful.

We therefore compare each extracted sentence with each catchphrase individually, using Rouge. If the recall (on the catchphrase) is higher than a threshold, the catchphrase-sentence pair is considered a match. For example if we have a 10-word catchphrase, and a 15 words candidate sentence, if they have 6 words in common we consider this as a match using Rouge-1 with a threshold of 0.5, but not a match with a threshold of 0.7 (requiring at least 7/10 words from the catchphrase to appear in the sentence). Using other Rouge scores (Rouge-SU or Rouge-W), the order and sequence of tokens are also considered in defining a match. In this way, once a matching criterion is defined, we can divide all the sentences in "**relevant**" sentences (those that match at least one catchphrase) and "**not relevant**" sentences (those that do not match any catchphrase).

Once the matches between single sentences and catchphrases are defined for a single document and a set of extracted (candidate) sentences, we can compute precision and recall as:

$$Recall = \frac{MatchedCatchphrases}{TotalCatchphrases}$$

$$Precision = \frac{RelevantSentences}{ExtractedSentences}$$

The recall is the number of catchphrases matched by at least one extracted sentence, divided by the total number of catchphrases; the precision is the number of sentences extracted which match at least one catchphrase, divided by the number of extracted sentences. This evaluation method gives us a way to compare the performance of different extraction systems automatically, by giving a simple but reasonable measure of how many of the desired catchphrases are generated by the systems, and how many of the sentences extracted are useful. This is different from the use of standard Rouge overall scores, where precision and recall do not relate to the number of catchphrases or sentences, but to the number of smaller units such as n-grams, skip-bigrams or sequences, which makes it more difficult to interpret the results.

5 Relevance Identification

Different techniques can be used to extract important fragments from text. Approaches such as (Hoffmann and Pham, 2003; Galgani and Hoffmann, 2010) used regular expressions to recognize patterns in the text, based on cue phrases or particular terms/constructs. However, when manually examining legal texts, we realised that to recognize important content, several aspects of the text need to be considered. Looking at one sentence by itself is clearly not enough to decide its importance: we must consider also document-scale information to know what the present case is about, and at the same time we need to look at corpus-wide information to decide what is peculiar to the present case. For this reason we developed several ways of locating potential catchphrases in legal text, based on different kinds of attributes, which form the building blocks for our rule system.

Using the NLTK library[3] (Bird et al., 2009), we collected all the words in the corpus, and obtained a list of stemmed terms (we used the Porter stemmer). Then for each term (stem) of each document, we computed the following numerical attributes:

1. Term frequency (**Tf**): the number of occurrences of the term in this document.
2. **AvgOcc**: the average number of occurrences of the term in the corpus.
3. Document frequency (**Df**): computed as the number of document in which the term appear at least once divided by the total number of documents.
4. **TFIDF**: computed as the rank of the term in the document (i.e. TFIDF(term)=10 means that the term has the 10 highest TFIDF value for this document).
5. **CpOcc**: how many times the term occurs in the set of all the known catchphrases present in the corpus.
6. The **FcFound** score: from (Galgani 2012), this uses the known catchphrases to compute the ratio between how many times (that is in how many documents) the term appears both in the catchphrases and in the text of the case, and how many times in the text [4] :

$$FcFound(t) = \frac{NDocs_{text\&catchp.}(t)}{NDocs_{text}(t)}$$

[3]http://www.nltk.org/

[4]Attributes 5 and 6 use information from the set of existing catchphrases. We consider this set as a general resource and believe that the corpus of catchphrases comprises most of the relevant words and phrases, and as such can be deemed a general resource and can be applied to new data without loss of performances, as it was shown in (Galgani et al., 2012).

7. **CitSen**: how many times the term occurs in all the sentences (from other documents) that cite the target case.

8. **CitCp**: how many times the term occurs in all the catcphrases of other documents that cite or are cited by the target case.

9. **CitLeg**: how many times the term occurs in the section titles of the legislation cited by the target case.

Three more non-numeric attributes were also used for each term:

10. The Part Of Speech (**POS**) tag of the term (obtained using the NLTK default part of speech tagger, a classifier-based tagger trained on the PENN Treebank corpus).

11. We extracted a set of legal terms from (Olsson, 1999), which lists a set of possible titles and subtitles for judgements. The existence of a term in this set is used as an attribute (**Legal**).

12. If the term is a proper noun (**PrpNoun**), as indicated by the POS tagger.

Furthermore, we also use four sentence-level attributes:

13. Specific words or phrases that must be present in the sentence, i.e. "court" or "whether".

14. If the sentence contains a citation to another case (**HasCitCase**).

15. If the sentence contains a citation to an act or a section of the law (**HasCitLaw**).

16. A constraint on the length of the sentence (**Length**).

When constructing our set of features, we included different kinds of information that can be used to recognize important content. Each of the different features can be used to locate potential catchphrases in a case. In (Galgani et al., 2011) automatic extraction methods based on these attributes were compared to each other, and it was shown that citation-based methods in general outperform text-only methods. However, we believe that different methods best apply to different contexts (for different documents and sentences), and we propose to combine them using manually created rules.

6 Building a Knowledge Base

Our catchphrase extraction system is based on creating a knowledge base of rules that specify which sentences should be extracted from the full text, as candidate catchphrases. These rules are acquired and organized in a knowledge base according to the RDR methodology.

As the rules are created looking at examples, we built a tool to facilitate the inspection of legal cases. The user, for each document, can explore the relevant sentences and see which ones are most similar to the (given) catchphrases of the case. The interface also shows citation information, the catchphrases, relevant sentences of cited/citing cases, and which parts of the relevant legislation are cited. For a document the user can see the "best" sentences: those that are more similar to the catchphrases, or those similar to one particular catchphrase. For each sentence, frequency information is also shown, according to the attributes described in Section 5.

In order to make a rule, the user looks at one example of a relevant sentence, together with all the frequency and citation information, the catchphrases and other information about the document. The user can then set different constraints for the attributes: attributes 1 to 12 refer to a single term, with attributes 1-9 being numeric (for these the user can specify a maximum and/or minimum value) while attributes 10-12 require an exact value (a POS tag or a True/False value). The user specifies how many terms which satisfy that constraint, must be present in a single sentence for it to be extracted (for example, there must be at least 3 terms with $FcFound > 0.1$). It is also possible to insert proximity constraints, such as: the 3 terms must be no more than 5 tokens apart (they must be within a window of 5 tokens). We call this set of constraints on terms, a condition. A rule is composed of a conjunction of conditions (for example: there must be 3 terms with $FcFound > 0.1$ and $AvgOcc < 1$ AND 2 terms with $CpOcc > 20$ and $CitCp > 1$). There is no limit on the number of conditions that form a rule. The conclusion of a rule is always "the sentence is relevant".

To acquire rules from the user, we follow the RDR approach, according to which the user looks at an instance that is currently misclassified and formulates a rule to correct the error. In our case, the user is presented with a sentence that matches at least one catchphrase (a relevant sentence), but is not currently selected by the knowledge base.

Looking at the sentence at hand, and at the attributes values for the different terms, the user specifies a possible rule condition, and can then test it on the entire dataset. This gives an immediate idea on how useful the condition is, as the user can see how many sentences would be selected by that condition and how many of these sentences are relevant (similar enough to at least one catchphrase, as defined in Section 4). At the same time the user can inspect manually other sentences matched by the condition, and refine the condition accordingly. When he/she is satisfied with one condition, they can add and test more conditions for the rule, and see other examples, to narrow down the number of cases matched by the rule and improve the precision while at the same time trying to include as many cases as possible.

When looking at the number of sentences matched by adding a condition, we can also compute the probability that the improvement given by the rule/condition is random. As initially described in (Gaines and Compton, 1995), for a two class problem (sentence is relevant/not relevant), we can use a binomial test to calculate the probability that such results could occur randomly. That is, when a condition is added to an existing rule, or added to an empty rule we compute the probability that the improvement is random. The probability of selecting randomly n sentences and getting x or more relevant sentences is:

$$r = \sum_{k=x}^{n} \binom{n}{k} p^k (1-p)^{n-k} = \frac{n! p^x (1-p)^{n-x}}{x!(n-x)!}$$

where p is the random probability, i.e. the proportion of relevant sentences among all sentences selected by the current rule. If we know how many relevant sentences the new condition select (x), we can calculate this probability which can guide the user in creating a condition that minimize the value of r.

As an example, the user may be presented with the following sentence:

> As might have been expected, the <u>bill of lading</u> contains a <u>"Himalaya" clause</u> in the widest terms which is usual in such transactions.

which we know to be relevant, being similar to a given catchphrase:

> *goods transported under <u>bill of lading</u> incorporating <u>Himalaya clause</u>*

Looking at the attributes the user proposes a condition, for example based on the term *lading* and *Himalaya* (that are peculiar of this document), a possible condition is:

> SENTENCE contains at least 2 terms with $CpOcc > 1$ and $FcFound > 0.1$ and $CitCp > 1$ and $TFIDF < 4$ and $AvgOcc < 1$

Testing the condition on the dataset we can see that it matches 1392 sentences, of which 849 are relevant (precision = 0.61), those sentences cover a total of 536 catchphrases (there are cases in which a number of sentences match the same catchphrase). The probability that a random condition would have this precision is also computed (10e-136). To improve the precision we can look at the two other terms that occurs in the catchphrase (*bill* and *clause*) and add another condition, for example:

> SENTENCE also contains at least 2 terms with $CpOcc > 20$ and $FcFound > 0.02$ and $CitCp > 1$ and $isLegal$ and $TFIDF < 16$

The rule with two conditions now matches 429 sentences of which 347 are relevant (precision=0.81), covering 331 catchphrases. The probability that a random condition added to the first one would bring this improvement is 10e-19. The user can look at other matches of the rule, for example:

> That is to say, the <u>Tribunal</u> had to determine whether the applicant was, by reason of his <u>war-caused</u> incapacity alone, prevented from continuing to undertake <u>remunerative</u> <u>work</u> that he had been undertaking.

remunerative and *war-caused* are matched by the first condition, and *Tribunal* and *work* by the second. If the user is satisfied the rule is committed to the knowledge base. In this way the creation, testing and integration of the rule in the system is done at the same time.

During knowledge acquisition this same interaction is repeated: the user looks at examples, creates conditions, tests them on the dataset until he/she is satisfied, and then commits the rule to the knowledge base, following the RDR approach. When creating a rule the user is guided both by particular examples shown by the system, and by statistics computed on the large dataset. Some rules of our KB are presented in Table 2.

Table 2: Examples of rules inserted in the Knowledge Base

SENTENCE contains at least 2 terms with $Tf > 30$ and $CpOcc > 200$ and $AvgOcc < 2.5$ and $TFIDF < 10$ within a window of 2
SENTENCE contains at least 2 terms with $Tf > 5$ and $CpOcc > 20$ and $FcFound > 0.02$ and $CitCp > 1$ and $TFIDF < 15$ and contains at least 2 terms with $Tf > 5$ and $CpOcc > 2$ and $FcFound > 0.11$ and $AvgOcc < 0.2$ and $TFIDF < 5$
SENTENCE contains at least 10 terms with $CitCp > 10$ and contains at least 6 terms with $CitCp > 20$
SENTENCE contains the term $corporations$ with $Tf > 15$ and $CitCp > 5$

7 Preliminary Results and Future Development

After building the knowledge acquisition interface, we conducted a preliminary KA session to verify the feasibility of the approach, and the appropriateness of the rule language. We conducted a KA session creating a total of 23 rules (which took on average 6.5 minutes for each to be specified, tested and commited). These 23 rules extracted a total of 12082 sentences, of which 10565 were actually relevant, i.e. matched a least one catchphrase, where we used Rouge-1 with a similarity threshold of 0.5 to define a match. These sentences are distributed among 1455 different documents. The overall precision of the KB is thus is 87.44% and the total number of catchphrases covered is 6765 (29.12% of the total).

Table 3 shows the comparison of this Knowledge Base with four other methods: Random is a random selection of sentences, Citations is a methods that use only citation information to select sentences (described in (Galgani et al., 2011)); in particular it selects those sentences that are most similar to the catchphrases of cited and citing documents. As a state-of-the-art general purpose summarizer, we used LexRank (Erkan and Radev, 2004), an automatic tool that first builds a network in which nodes are sentences and a weighted edge between two nodes shows the lexical cosine similarity, and then performs a random walk to find the most central nodes in the graphs and takes them as the summary. We downloaded the Mead toolkit[5] and applied LexRank to all the documents to rank the sentences. For every method we extracted the 5 top ranked sentences. Finally, because our rules have matches in only 1455 documents (out of a total of 2816), we used a mixed approach in which for each document, if there is any sentence(s) selected by the KB we select those, otherwise we take the best 5 sentences as given by the Citation method. This method is called KB+Citations. We can see from the Table that the Knowledge Base outperforms all other methods in precision, followed by KB+Citations, while KB+Citations obtains higher recall.

Table 3: Performances measured using Rouge-1 with threshold 0.5. SpD is the average number of extracted sentences per document.

Method	SpD	Precision	Recall	F-measure
KB	4.29	0.874	0.291	0.437
Citations	4.56	0.789	0.527	0.632
KB+CIT	7.29	0.828	0.553	0.663
LexRank	4.87	0.563	0.402	0.469
Random	5.00	0.315	0.233	0.268

Table 4: Performances measured using Rouge-1 with threshold 0.7. SpD is the average number of extracted sentences per document.

Method	SpD	Precision	Recall	F-measure
KB	4.29	0.690	0.161	0.261
Citations	4.56	0.494	0.233	0.317
KB+CIT	7.28	0.575	0.265	0.363
LexRank	4.87	0.351	0.216	0.267
Random	5.00	0.156	0.098	0.120

Note that we can vary the matching criterion (as described in Section 4) and only consider more strict matches, in this case only sentences more similar to catchphrases are considered relevant. We can see the results of setting a higher similarity threshold (0.7) in Table 4. All the approaches give lower precision and recall, but the margin of the knowledge base over the other methods increases, with a relative improvement of precision of 40% over the citation method.

While the precision level of the KB alone is higher than any other method, the recall is low when compared to other approaches. We only conducted a preliminary KA session, which took slightly more than 2 hours. Figure 1 shows precision and recall of the KB as new rules are in-

[5]www.summarization.com/mead/

serted into the system. We can assume that a more comprehensive set of rules, capturing more sentences and addressing different types of contexts, should cover a greater number of catchphrases, while keeping the precision at a high value; however, the rules constructe so far only fire for some cases, and many cases are not covered at all.

Even with this limited KB, we can use the citation method as fall-back to select sentences for those cases that are not matched by the rules. Using this approach, as we can see from Tables 3 and 4 (method KB+CIT), that obtain the highest recall while keeping the precision very close to the precision of the KB alone.

For future work we plan not only to expand the KB in general with more rules, in order to improve recall, but also to construct rules specifically for those cases that are not already covered, applying those rules in a selective way, only for these of documents (and not for those which already have a sufficient number of catchphrases candidates). In doing this we will seek to generalize our experience of applying the citation approach to documents where the KB did not produce catchphrases. We also hypothesize that the recall level of the rules is low because they select several sentences that are similar among them, and thus match the same catchphrases, so that for some documents we have a set of relevant sentences which cover only some aspects of the case. Using a similarity-based re-ranker would allow us to discard sentences to similar to those already selected.

In future developments we also plan to develop further the structure of the knowledge base into an RDR tree, writing exception rules (rule with conclusion "not relevant") that can patch the existing rules whenever an error is found. The current knowledge base only consists of a list of rules while the RDR methodology will let us organize the rules so they are used in different situations depending on which previous rule has fired.

8 Conclusion

This paper presents our hybrid approach to text summarization, based on creating rules to combine different types of statistical information about text. In contrast to supervised learning, where human intuition applies only to attribute and algorithm selection, here human intuition also applies to the organization of features in rules, but still guided by the available dataset.

We have applied our approach to a particular summarization problem: creating catchphrases for legal case reports. Catchphrases are considered to be a significant help to lawyers searching through cases to identify relevant precedents and are routinely used when browsing documents. We created a large dataset of case reports, corresponding catchphrases and both incoming and outgoing citations to cases and legislation. We created a Knowledge Acquisition framework based on Ripple Down Rules, and defined a rich rule language that includes different aspects of the case under consideration. We developed a tool that facilitates the inspection of the dataset and the creation of rules by selecting and specifying features depending on the context of the present case and using different information for different situations. A preliminary KA session shows the effectiveness of the rule approach: with only 23 rules we can obtain a significantly higher precision (87.4%) than any automatic method tried. We are confident that a more extensive knowledge base would further improve the performances and cover a larger portion of the cases, improving the recall.

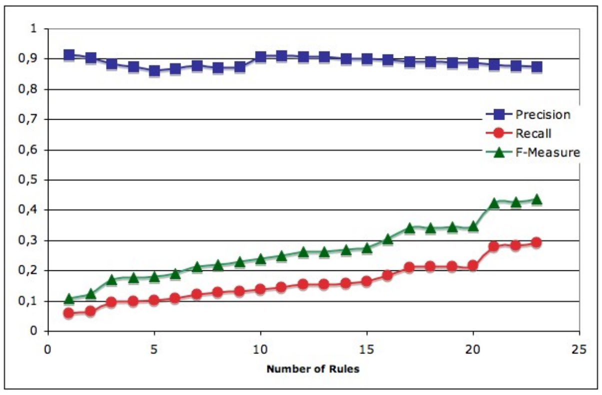

Figure 1: Precision, Recall and F-measure as the size of the KB increases

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media.

P. Compton and R. Jansen. 1990. Knowledge in context: a strategy for expert system maintenance. In *AI '88: Proceedings of the second Australian Joint Conference on Artificial Intelligence*, pages 292–306, New York, NY, USA. Springer-Verlag New York, Inc.

G. Erkan and D.R. Radev. 2004. LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22(2004):457–479.

Atefeh Farzindar and Guy Lapalme. 2009. Machine translation of legal information and its evaluation. *Advances in Artificial Intelligence*, pages 64–73.

B. R. Gaines and P. Compton. 1995. Induction of ripple-down rules applied to modeling large databases. *J. Intell. Inf. Syst.*, 5:211–228, November.

Filippo Galgani and Achim Hoffmann. 2010. Lexa: Towards automatic legal citation classification. In Jiuyong Li, editor, *AI 2010: Advances in Artificial Intelligence*, volume 6464 of *Lecture Notes in Computer Science*, pages 445 –454. Springer Berlin Heidelberg.

Filippo Galgani, Paul Compton, and Achim Hoffmann. 2011. Citation based summarization of legal texts. Technical Report 201202, School of Computer Science and Engineering, UNSW, Australia.

Filippo Galgani, Paul Compton, and Achim Hoffmann. 2012. Towards automatic generation of catchphrases for legal case reports. In Alexander Gelbukh, editor, *the 13th International Conference on Intelligent Text Processing and Computational Linguistics*, volume 7182 of *Lecture Notes in Computer Science*, pages 415–426, New Delhi, India. Springer Berlin / Heidelberg.

G. Greenleaf, A. Mowbray, G. King, and P. Van Dijk. 1995. Public Access to Law via Internet: The Australian Legal Information Institute. *Journal of Law and Information Science*, 6:49.

Ben Hachey and Claire Grover. 2006. Extractive summarisation of legal texts. *Artif. Intell. Law*, 14(4):305–345.

Achim Hoffmann and Son Bao Pham. 2003. Towards topic-based summarization for interactive document viewing. In *K-CAP '03: Proceedings of the 2nd international conference on Knowledge capture*, pages 28–35, New York, NY, USA. ACM.

Myung Hee Kim, Paul Compton, and Yang Sok Kim. 2011. Rdr-based open ie for the web document. In *Proceedings of the sixth international conference on Knowledge capture*, K-CAP '11, pages 105–112, New York, NY, USA. ACM.

Julian Kupiec, Jan Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *SIGIR '95: Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 68–73, New York, NY, USA. ACM.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In Stan Szpakowicz Marie-Francine Moens, editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain, July. Association for Computational Linguistics.

Daniel Marcu. 1997. From discourse structures to text summaries. In *In Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*, pages 82–88.

Q. Mei and C.X. Zhai. 2008. Generating impact-based summaries for scientific literature. *Proceedings of ACL-08: HLT*, pages 816–824.

Marie-Francine Moens. 2001. Innovative techniques for legal text retrieval. *Artificial Intelligence and Law*, 9(1):29–57, 03.

Marie-Francine Moens. 2007. Summarizing court decisions. *Inf. Process. Manage.*, 43(6):1748–1764.

Justice Leslie Trevor Olsson. 1999. *Guide To Uniform Production of Judgments*. Australian Institute of Judicial Administration, Carlton South, Vic, 2nd edition.

Son Bao Pham and Achim Hoffmann. 2004. Incremental knowledge acquisition for building sophisticated information extraction systems with kaftie. In *in 5th International Conference on Practical Aspects of Knowledge Management*, pages 292–306. Springer-Verlag.

Vahed Qazvinian and Dragomir R. Radev. 2008. Scientific Paper Summarization Using Citation Summary Networks. *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 689–696.

Gerard Salton, Amit Singhal, Mandar Mitra, and Chris Buckley. 1997. Automatic text structuring and summarization. *Inf. Process. Manage.*, 33(2):193–207.

Han Xu and Achim Hoffmann. 2010. Rdrce: Combining machine learning and knowledge acquisition. In Byeong-Ho Kang and Debbie Richards, editors, *Knowledge Management and Acquisition for Smart Systems and Services*, volume 6232 of *Lecture Notes in Computer Science*, pages 165–179. Springer Berlin / Heidelberg.

Mehdi Yousfi-Monod, Atefeh Farzindar, and Guy Lapalme. 2010. Supervised machine learning for summarizing legal documents. In *Canadian Conference on Artificial Intelligence 2010*, volume 6085 of *Lecture Notes in Artificial Intelligence*, pages 51–62, Ottawa, Canada, may. Springer.

Paul Zhang and Lavanya Koppaka. 2007. Semantics-based legal citation network. In *ICAIL '07: Proceedings of the 11th international conference on Artificial intelligence and law*, pages 123–130, New York, NY, USA. ACM.