CaseSummarizer: A System for Automated Summarization of Legal Texts

Seth Polsley **Pooja Jhunjhunwala*** **Ruihong Huang**
Department of Computer Science and Engineering, Texas A&M University
spolsley, pj861992, huangrh@cse.tamu.edu

Abstract

Attorneys, judges, and others in the justice system are constantly surrounded by large amounts of legal text, which can be difficult to manage across many cases. We present CaseSummarizer, a tool for automated text summarization of legal documents which uses standard summary methods based on word frequency augmented with additional domain-specific knowledge. Summaries are then provided through an informative interface with abbreviations, significance heat maps, and other flexible controls. It is evaluated using ROUGE and human scoring against several other summarization systems, including summary text and feedback provided by domain experts.

1 Introduction

Legal systems across the world generate massive amounts of unstructured text everyday; judges, lawyers, and case workers process and review millions of cases each year in the United States alone. These case files may be very long, often including hundreds of pages of dense legal text. Some form of automating or simplifying the review process could help legal workers manage this workload better. In this work, we consider automated text summarization as one means to this end.

Summarization is a challenging sub-task of the broader text-to-text generation field of natural language processing (NLP). Summaries are usually generated by extracting 'important' portions of the text. Extraction-based methods are often used because abstraction-based summarization is an open problem in NLP. Abstraction-based summarization is intended to generate summaries based on abstract representations of the text, inspired by how humans generate summaries based on their own understanding of text; there is a great deal of ongoing research devoted to developing these methods (Moratanch and Chitrakala, 2016).

In extraction-based methods, the most relevant sentences or phrases of a document may be found through a metric like TF*IDF (Nenkova and McKeown, 2012), and while this is a useful approach to general text summarization, it can miss a lot of critical information in certain domains. For instance, legal documents have a large amount of technical content. Domain-specific summarizing systems have been developed for many different fields as one means of addressing this limitation of general summarizers; they use knowledge of the content specifically in that domain to boost performance. CaseSummarizer is a summary engine specific to the legal domain that builds on existing methods paired with domain-specific constructs to present an interface with scalable summary text, lists of entities and abbreviations from the document, and a significance heat map of the entire text.

2 Background

Several systems have been built for the explicit purpose of summarizing legal documents. One of the earliest works in this area is the "Fast Legal EXpert CONsultant" (FLEXICON) system developed by Gelbart and Smith (Gelbart and Smith, 1991a). FLEXICON is keyword-based, referencing against large database of terms to find important regions of text (Gelbart and Smith, 1991b). Moens et al. later introduced SALOMON which uses cosine similarity to group regions of the text that are similar (Moens et al., 1999). The goal of this approach is to extract relevant portions of different topics in the text, similar to some other abstraction-oriented methods (Barzilay and Elhadad, 1999; Erkan and Radev, 2004). LetSum, developed by Farzindar and Lapalme, more closely resembles a keyword-based system, employing a set of "cue phrases" to identify portions of the text associated with specific themes like 'Introduction', 'Context', and 'Conclusion' (Farzindar and Lapalme, 2004). While LetSum performed relatively well against the human-provided summaries, the shortened text was found to be too long. Other extraction-based

*Software Engineer, Google Inc.

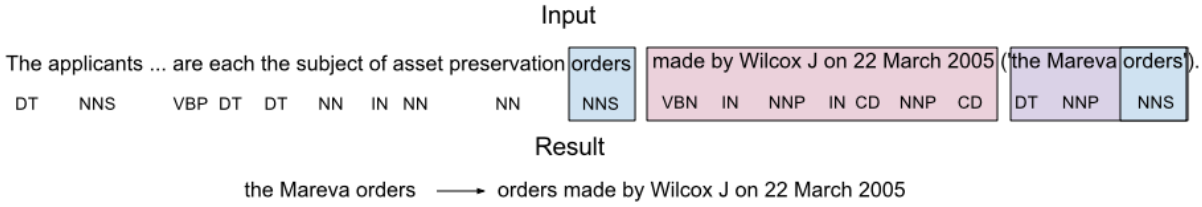

Figure 1: An example of an abbreviation being extracted with its source text. The algorithm attempts to match the POS of the head word of a parenthetical phrase with the first matching POS before the abbreviation.

methods have been developed to overcome a reliance on language-dependent keywords using graph-based ranking (Mihalcea, 2005; Wong et al., 2008).

A large body of recent work has been presented by Galgani and Hoffmann through LEXA, a system which uses citation analysis to generate summaries (Galgani et al., 2012a; Galgani and Hoffmann, 2010). LEXA includes an interface for continued system learning using Ripple-down Rules (RDR), which allows domain experts to evaluate sentence selections live and agree or disagree with the selections. When the experts agree on a relevant sentence, a new extraction pattern is added (Galgani et al., 2015). Galgani et al. continued their work in this domain with the development of a multi-technique approach to summarization, including 'catchphrase' analysis (Galgani et al., 2012b). CaseSummarizer is a multi-technique approach with a goal of providing a comprehensive interface that pairs scalable controls with supplemental details like abbreviations and significance heat maps.

3 Implementation

CaseSummarizer's internal pipeline consists of three distinct steps: preprocessing, scoring of sentence relevance, and domain processing; summaries are then presented externally through the user interface.

3.1 Internal Pipeline

CaseSummarizer is built in Python and uses the feature-rich Natural Language ToolKit (NLTK) module for pre-processing by splitting documents into sentences which are stemmed, lemmatized, case-normalized, and cleared of stop words (Bird et al., 2009). Sentences are scored using a TF*IDF matrix built from thousands of legal case reports, which counts term frequency using $TF * IDF_t = TF_t * 1/\log \frac{N}{DF_t}$ where N refers to the number of documents, TF_t is the total count of term t, and DF_t is the number of documents in which t appears. These scores are summed over each sentence and normalized by the sentence length. This normalization step ensures the system does not bias long sentences.

In order to include additional domain information, CaseSummarizer first extracts a list of all entities from the text. Parties can be extracted from case titles because of the document structure. Similarly, abbreviations of entity names are identified by CaseSummarizer to aid the reader's understanding of summaries. This is done by determining the Part-Of-Speech (POS) of the head words of parenthetical phrases and reading right-to-left until the earliest non-consecutive occurrence of that POS is found in the text. See Figure 1 as an example.

CaseSummarizer does not use specific cue words or catchphrases, but adjusts sentence scores using occurrences of known entities, dates, and proximity to section headings. The adjustment function is $w_{new} = w_{old} + \sigma(0.2d + 0.3e + 1.5s)$, where σ is the standard deviation among sentence scores, d refers to the number of dates present, e is the number of entity mentions, and s is a boolean indicating the start of a section. The weights primarily were selected through trial-and-error to reflect the relative importance of each term, e.g. dates are less useful than entities, and feedback from experts indicated that section headings should carry heavier weight.

3.2 User Interface

User interaction is performed through a web interface which provides all extracted information and some adjustable controls. After selecting the case to summarize, the fields are populated with the parties and date, followed by the list of all recognized entities. A listing of abbreviations matches all phrases to their original full form in the text; this information can help the reader quickly discern which entities are being referenced when an abbreviation appears. These fields are shown in Figure 2.

The sentence scores are manifest in two forms: the summaries themselves and significance heat maps. The summaries are fully scalable using a slider, allowing the user to show only the most important sentences at any compression level. The significance heat map presents the full document text but assigns a color to each sentence based on the pertinence score it received during the weighting stage. By using the summary text and the heat map together, CaseSummarizer provides a helpful reference to users for identifying important regions in the text. See Figure 3 for an example.

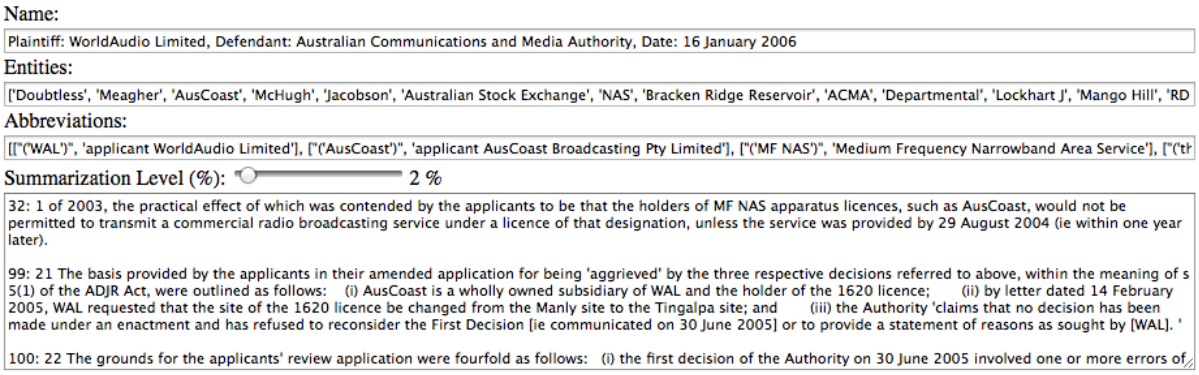

Figure 2: The extracted fields for a sample case, showing the names of parties, entities, a listing of abbreviations and their full forms, and the scalable summary text.

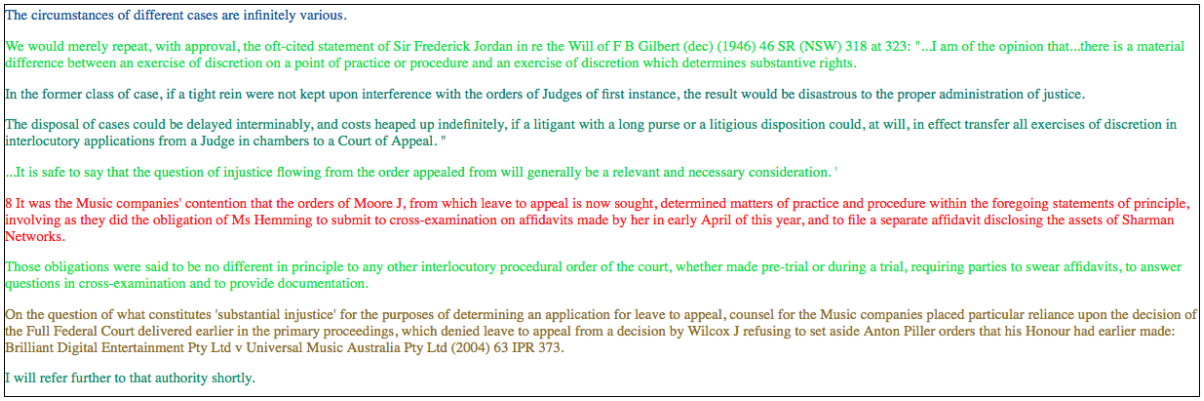

Figure 3: A snippet of the case's full text in the significance heat map. Each sentence is color-coded based on the its score, ranging from low (blue) to high (red).

4 Evaluation

Because summaries are very subjective, evaluation can be difficult; Lin et al. introduced a set of metrics called the ROUGE package in (Lin, 2004) that provide a pairwise comparison method for evaluating candidate summaries against human-provided ones. The ROUGE metric has multiple variants and may be applied at the word, phrase, or sentence level. In this case, we used ROUGE-N, which measures the overlap of n-grams between summaries, with N = 1, 2, 3, and 4. We also computed the ROUGE-L score, a metric similar to an F-measure based on sentence-level similarity of two summaries. In addition to ROUGE scores, we asked domain experts to rate several summaries using a set of six evaluation questions based on the original set of questions presented by Liu and Liu for ranking summaries in (Liu and Liu, 2008). We also consulted the experts for feedback on the system.

CaseSummarizer uses the same data set as LEXA, which was created and released by Galgani et al. It contains 3890 legal cases from the Federal Court of Australia (FCA) from the years 2006-2009. Evaluation was performed on a set of 5 randomly-selected documents. Six automated tools were chosen for comparison. Four were online programs, AutoSummarizer, TextSummarizer, SplitBrain, and SMMRY[1]. The other two were Apple Inc.'s Summarizer program and Galgani et al.'s summaries included with the data set. We also asked domain experts to provide summaries for randomly selected cases. For consistency in the ROUGE metrics, we selected a compression rate of 3% in the automated systems. The domain experts were asked to generate sentence-level summaries by extracting approximately 3% of the sentences from the document.

Table 1 shows the ROUGE scores of each system against the expert summaries. We can see that CaseSummarizer performs very favorably against the other systems when evaluated against expert summaries. The domain expert ratings are shown in Figure 4 alongside each evaluation question. While the automated summaries are still lacking across the board when compared to the expert-generated ones, CaseSummarizer is most effective in capturing a coherent flow of events and obtaining a good coverage of important points in a case. It also received the best average rating among all the automatic systems.

[1] autosummarizer.com, textsummarization.net/text-summarizer, splitbrain.org/services/ots, and smmry.com/, respectively

Table 1: ROUGE scores indicating the similarity between automatically-generated summaries and the expert-generated summaries.

	CaseSum	AutoSum	TextSum	SplitBrain	SMMRY	Apple Sum	Galgani et al.
Rouge-1	0.194	0.207	0.183	0.241	0.248	0.175	0.132
Rouge-2	0.114	0.089	0.072	0.146	0.137	0.097	0.049
Rouge-3	0.091	0.059	0.049	0.123	0.104	0.075	0.026
Rouge-4	0.085	0.048	0.043	0.117	0.090	0.068	0.019
Rouge-L	0.061	0.017	0.015	0.056	0.062	0.033	0.017

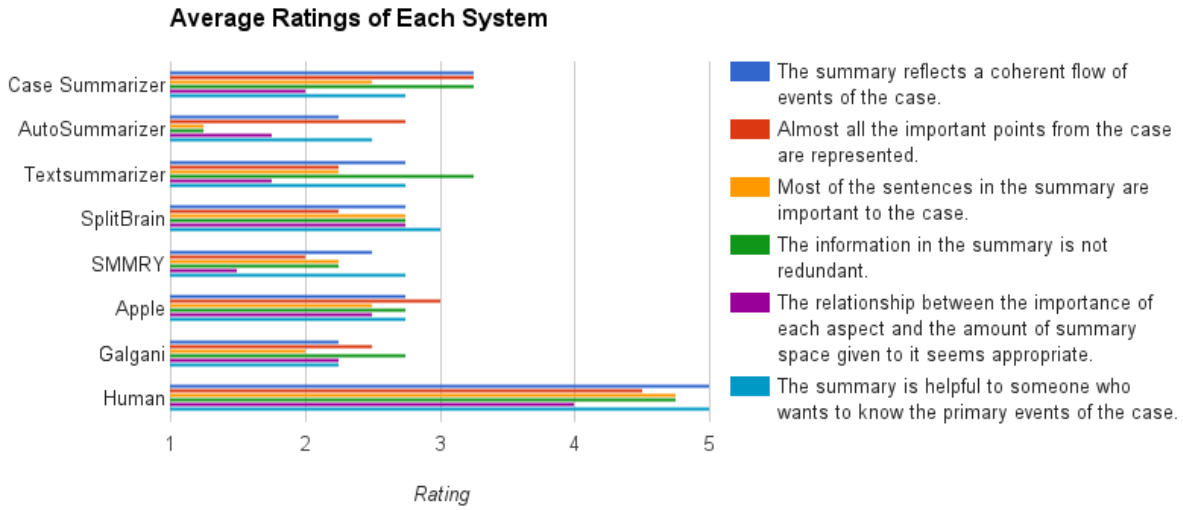

Figure 4: Domain expert ratings of each summary, including the expert-generated ones. The evaluation questions are shown on the right-hand side with the average scores for each method shown on the left-hand side.

5 Future Work

One of the most interesting findings from the summary scoring study is that the expert-generated summaries received very high ratings from other experts, as shown in Figure 4. These summaries were also generated entirely from sentence extraction, like the automated systems. This indicates both the value of sentence-level summarization on legal documents and provides some validation that sentence extraction methods can indeed generate helpful summaries. However, the disparaty between expert summary scores and the automated systems highlights the need for future improvements in summarization methods. To further explore these ideas, we consulted with domain experts regarding the CaseSummarizer system. The following points outline some of their primary suggestions.

- Extracted sentences need to be more representative of the different sections of a case file, e.g. premise, arguments, findings, judgements, etc.

- A considerable amount of repetition of ideas was observed in the summaries generated by the system, which should be discouraged.

- Most domain-experts believed that a better summary would be generated by selecting sentences that are closer to the end of the document as these sentences often tend to summarize the points discussed in the whole document.

- Experts also pointed out the need for different kinds of summaries in the legal field. For instance, in one use case, a lawyer may wish to have highlights of key factual points to refresh his or her memory of the details of a case, but another attorney may wish to see only the findings to determine the relevance to some current proceedings.

6 Conclusion

We found that CaseSummarizer performs favorably against non-domain specific summarizers. The summaries generated are able to provide a reasonable idea about the context of a case, even though some important points are missed. While not able to perform as well as human experts, it fared the best among several other systems when evaluated by humans, and the domain experts suggested several improvements we hope to explore in the future work. Foremost, we seek to dissuade repetition by penalizing similar sentences. We also plan to add incentives to favor sentences near the end of documents as they may include vital information, and finally, we wish to explore extracting better representations of different sections using cue words. CaseSummarizer shows a promising start in combining summarization techniques into a multi-faceted interface with domain-inspired information.

Acknowledgments

The authors would like to thank each of the domain experts who provided high-quality summaries and ratings on the systems, as well as all those who gave feedback on CaseSummarizer's various iterations.

References

Regina Barzilay and Michael Elhadad. 1999. Using lexical chains for text summarization. *Advances in automatic text summarization*, pages 111–121.

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

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:457–479.

Atefeh Farzindar and Guy Lapalme. 2004. Letsum, an automatic legal text summarizing system. *Legal knowledge and information systems, JURIX*, pages 11–18.

Filippo Galgani and Achim Hoffmann. 2010. Lexa: Towards automatic legal citation classification. In *AI 2010: Advances in Artificial Intelligence*, pages 445–454. Springer.

Filippo Galgani, Paul Compton, and Achim Hoffmann. 2012a. Citation based summarisation of legal texts. In *PRICAI 2012: Trends in Artificial Intelligence*, pages 40–52. Springer.

Filippo Galgani, Paul Compton, and Achim Hoffmann. 2012b. Combining different summarization techniques for legal text. In *Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data*, pages 115–123. Association for Computational Linguistics.

Filippo Galgani, Paul Compton, and Achim Hoffmann. 2015. Lexa: Building knowledge bases for automatic legal citation classification. *Expert Systems with Applications*, 42(17):6391–6407.

Daphne Gelbart and JC Smith. 1991a. Flexicon, a new legal information retrieval system. *Can. L. Libr.*, 16:9.

Dephne Gelbart and JC Smith. 1991b. Beyond boolean search: Flexicon, a legal tex-based intelligent system. In *Proceedings of the 3rd international conference on Artificial intelligence and law*, pages 225–234. ACM.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*, volume 8.

Feifan Liu and Yang Liu. 2008. Correlation between rouge and human evaluation of extractive meeting summaries. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 201–204. Association for Computational Linguistics.

Rada Mihalcea. 2005. Language independent extractive summarization. In *Proceedings of the ACL 2005 on Interactive poster and demonstration sessions*, pages 49–52. Association for Computational Linguistics.

Marie-Francine Moens, Caroline Uyttendaele, and Jos Dumortier. 1999. Abstracting of legal cases: the potential of clustering based on the selection of representative objects. *Journal of the Association for Information Science and Technology*, 50(2):151.

N Moratanch and S Chitrakala. 2016. A survey on abstractive text summarization. In *Circuit, Power and Computing Technologies (ICCPCT), 2016 International Conference on*, pages 1–7. IEEE.

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. In *Mining Text Data*, pages 43–76. Springer.

Kam-Fai Wong, Mingli Wu, and Wenjie Li. 2008. Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 985–992. Association for Computational Linguistics.