ASHRAM: Active Summarization and Markup

Mary S. Neff and James W. Cooper
IBM Thomas J. Watson Research Center
MaryNeff@watson.ibm.com, jwcnmr@watson.ibm.com

Abstract

Typically, searching for information in a document collection amounts to refining a query and then scanning a large number of documents to determine their relevance. Active Summarization Having Related Active Markup (ASHRAM) is a facility for representing and automatically selecting, marking, and linking useful and/or salient items in a document, to make it easier for the user to determine the main points in a document or navigate through documents without having to read all of them. ASHRAM is a novel client server system and user interface consisting of dynamically generated HTML, JavaScript and Java which requests information from a document database stored on a server.

We describe a system for summarization by sentence extraction and a user interface for representation that allows the user to exploit the summary not only as an aid for relevance assessment of documents, but as an active aid to document navigation.

The server-based scalable text summarization and keyword extraction system uses Natural Language Processing (NLP) technology and corpus-based NLP techniques in the foreground and databases constructed using NLP technology in the background.

Introduction

The problem of finding important and relevant documents in an online document collection becomes increasingly difficult as documents proliferate. Our group has previously described the technique of Prompted Query Refinement (Cooper & Byrd, 1997, 1998) to assist users in focusing or directing their queries more effectively. However, even after a query has been refined, the problem of having to read too many documents still remains.

Once a list of document titles is presented to users, they would like to scan the documents quickly to see how important they are to the area they are investigating. In this paper we propose that generating a summary based on sentences containing the most salient single- or multi-word terms and presenting the terms and that summary in a novel user interface can be an extremely useful way for these users to move on to understanding the general contents of these documents.

The Search and Summarization User Interface

As with most search and retrieval systems, ASHRAM also starts with a query from the user. The server then returns a list of document titles in a simple list box. However, unlike systems that return the first few hundred characters of the document, it can present a computed summary of each document as the user clicks on that document's title.

Then, when a document on the list is selected for viewing, the server performs an analysis on it, finding the salient terms and marking them for display. The contents of the query, the statistical profile of items in the document vis-a-vis their profiles in the collection, and the location of items in different parts of the document, all work to determine salience of vocabulary items and sentences.

In the same fashion that salient items are identified, ASHRAM can extract the most salient sentences ahead of time and store them in the database as a summary that can then be presented to users when they click on the document title. In the interface shown in Figure 1, the summary appears in a box below the list of document titles.

The number of keywords displayed and length of the extracted summary are controlled by the user. When the user selects a document of interest for display, the document appears with both the salient terms and the summary at the top, as shown in Figure 2.

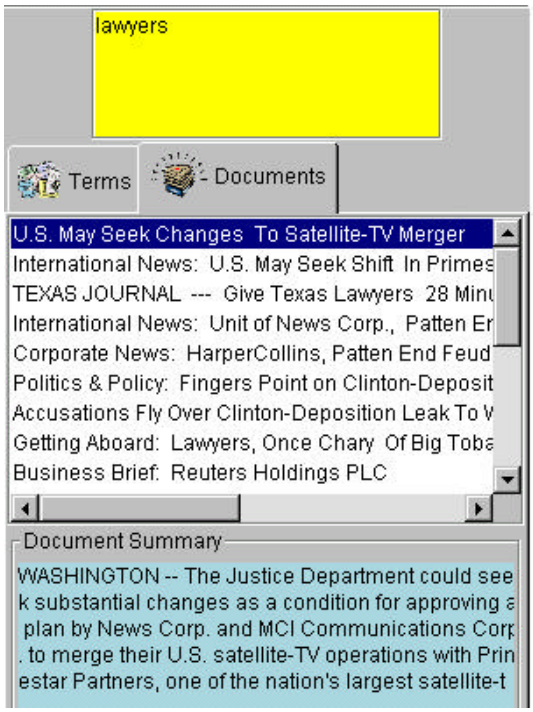

Figure 1. A list of document hits, showing the generated summary of the selected document.

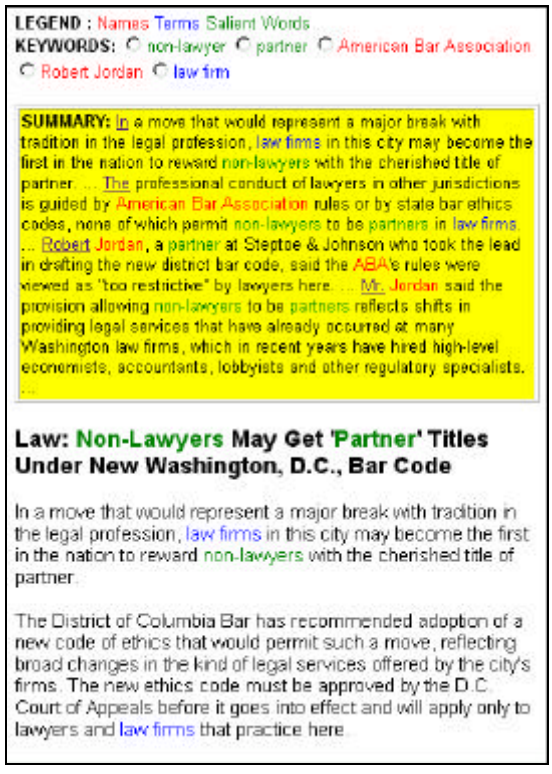

Figure 2. A marked document showing the major key words and the generated summary.

The user also has control over the way the terms, names, and salient words are highlighted in the keyword list, summary, and document. Three different colors distinguish items by category (terms, names, words); or a graduated scale of RGB color values, from yellow to red to blue to black, to distinguish items by salience, with yellow for highest salience and black for lowest. In addition, we show the terms marked throughout the document wherever they occur. For monochrome presentations, the program can also mark up the document to use italic, bold and bold-italic fonts.

Active Summary Hyperlinks

The summary in Figure 2 is *active* in that the first word in each sentence is hyper-linked to the location of the corresponding sentence in the document. Thus, clicking on the beginning of each sentence causes the browser display to jump to that location in the document. While this does not allow navigation to other documents or terms, it does allow the user to see the abstracted sentences in the context of the full document.

Active Term Markup

One of the most powerful methods of navigation through a group of documents utilizes a technique we call Active Markup. In Active Markup, the list of salient terms posted at the top of the displayed document are themselves active page components which can cause the server to return related information. In this implementation, these active components are used to query the server for a list of related terms to display. We show a portion of a document illustrating this active markup in Figure 3.

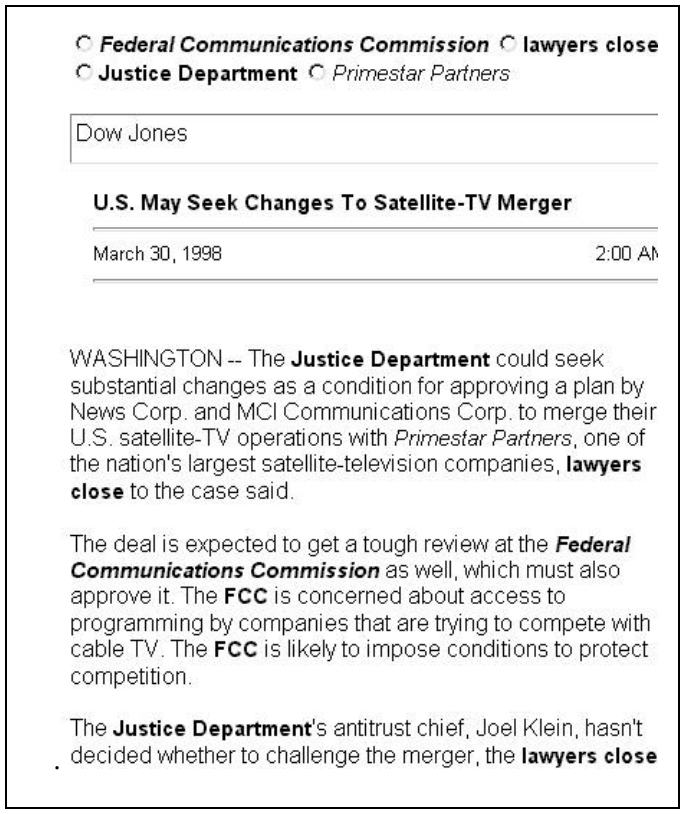

Figure 3. A marked up document, showing the salient terms. Here we use font differences to show the markup better on a black and white page. We omit the summary for simplicity.

Figure 3 shows the markup and document generated from a search for "lawyers" in a collection of Information Technology documents. Selecting "Federal Com- munications Commission" produces the window in Figure 4. The top list box contains the Context Thesaurus (see next section) of terms that commonly co-occur with the original query term. This term list provides a locale in which to begin exploration of the document space. Then, selecting any term in that list (here, "1996 Telecommunications Act") produces a list of documents which contain that term.

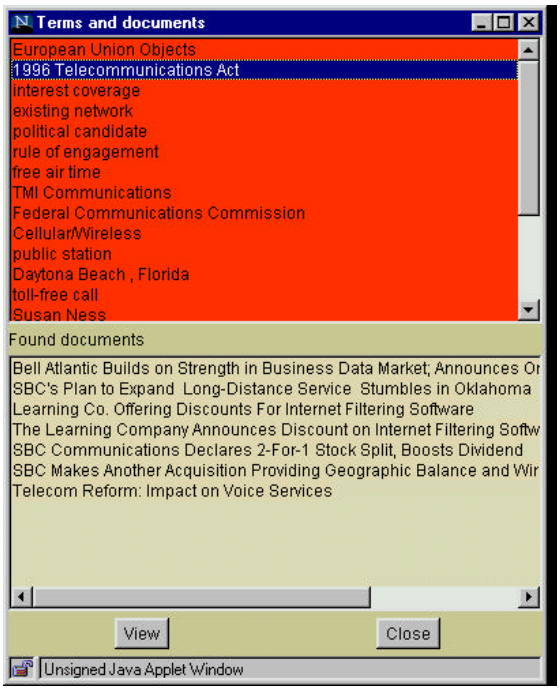

Figure 4. The terms and documents window

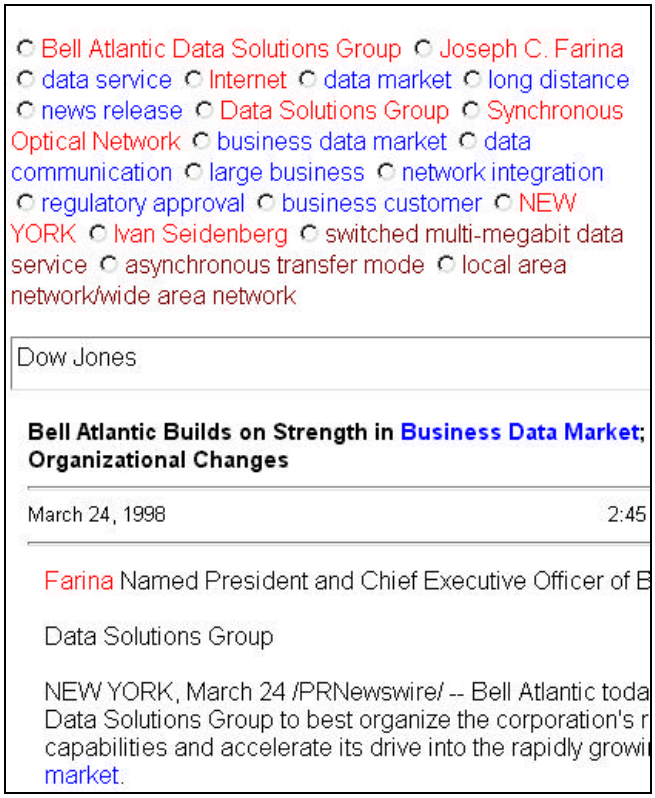

Figure 5. The next marked up document. Again, we omit the summary for simplicity.

Users can continue exploring the document collection by viewing one of the documents, which is itself marked up, as shown in Figure 5. Thus, it is possible for this navigation through document space to continue as long as the user is interested in following the leads that it provides.

This active markup approach coupled with the computer generated summaries provides a form of "query-free" searching, allowing the user to explore both the lexical space around terms and the document space in the locale of specific terms.

Relations Among Keywords

In constructing the list of related terms, we make use of an index we have termed the Context Thesaurus (Cooper & Byrd, 1997, 1998). This index was inspired by the "phrase finder" procedure described by Jing and Croft (1994) and consists of an ordinary information retrieval document index, where the documents are "pseudo-documents" derived from the original document collection. There is one pseudo document for each vocabulary item, and it contains the contexts in which that item occurs within the collection.

A suite of term extraction tools collectively known as *Textract* is used to process the collection and recognize these single and multi-word terms and count their frequency in the collection. The complete set of vocabulary items we discover in the collection is termed the *collection vocabulary* and is stored as part of a larger relational database.

Textract also parses the documents to find terms which participate in named relations with other terms, such as "makes," "president of," "is located in." It also uses statistical measures of co-occurrence to compute the strength of bi-directional unnamed relations between terms. We store these relationships between the items in the collection vocabulary in the relational database and refer to them as the *Lexical Network.*

Other Active Markup Displays

In Figure 4, we have displayed the Context Thesaurus of terms that commonly co-occur in the document collection with the original query term. As we noted, this term list provides a starting point from which to begin exploration of the document space. It would also be possible to display only terms having named or strong bi-directional unnamed relations with the initially selected term, by querying the lexical network database instead of the Context Thesaurus. In addition, one could imagine the user selecting several of these terms to get a display of just those documents containing the terms that were selected. This again amounts to a rapid form of document discovery without either entering any additional query or indeed needing to read the intervening documents. We have referred to this process as "query free searching."

Technology of Active Markup

Active markup has been implemented as a three-stage process. First, when the document is processed, the markup program inserts a small JavaScript program and a reference to a Java program, which is downloaded with the resulting page. All of the active terms are enclosed in an HTML form, and when any of them is selected, the JavaScript OnClick event is called. This allows the included JavaScript program to call the Java program, which in turn can contact the server to provide the terms that are displayed in the upper list box of the pop-up window. A click on any of these terms again calls the server, implemented using Java Remote Method Invocation (RMI), which returns the list of document titles when any term is selected.

Similarly, clicking on the document title and on the View button causes the Java program to ask the server to fetch that document and mark it up. It then launches a new instance of the browser window to download and display the new marked-up document.

The ASHRAM Server

The ASHRAM server provides a connection to a collection vocabulary and Lexical Network extracted from the collection, as well as a searchable document index and the Context Thesaurus.

A simple diagram of the major tables and relations is shown in Figure 6. The tables consist of terms, relations, and the names of those relations and collectively represent the Lexical Network.

The client and server are both written using Java 1.1 and communicate using Remote Method Invocation (RMI), (Cooper, 1997).

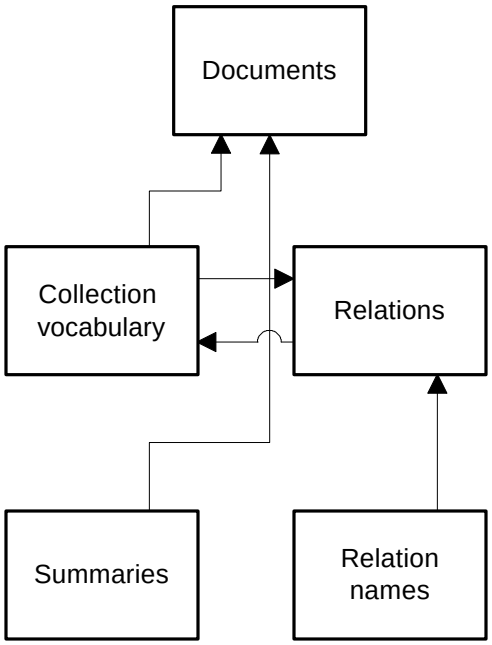

Figure 6. The structure of the relational database that constitutes the Lexical Network.

Once all the documents have been analyzed and indexed and the collection vocabulary built, each document is summarized, and the summary is stored in a database so that later the summary can be displayed to the user without a document fetch. The summary consists of the most salient sentences in the document, usually at least four sentences or ten percent of the length of the document. Summaries of other lengths can be delivered on the fly, but require perceptible processing time. As can be inferred from the description below, summarization must be done on a second pass, because sentence salience is, among other things, a function of term salience, which is, in turn, derived in part from the frequency of vocabulary items in the collection.

ASHRAM Text Extraction Techniques

ASHRAM relies on a suite of natural language processing (NLP) techniques for summarization, keyword extraction, and analysis text for markup. These are described below in the context of summarization and keyword extraction.

Text Summarization: Background

Even before the concept of digital documents, before the on-line information glut, there was interest in automatic text summarization. The early efforts were aimed at abstracting scientific papers in an informative way so that readers would know the experiment and the findings without having to read the paper. Summarizers for scientific papers have, in general, been the most successful in capturing all the important information in the text because they have tended to be domain- and/or genre-specific.

Today, there is a focus on general-purpose, domain-independent and/or customizable approaches which are usable in creative ways in interactive environments. But at the same time, automatically generating an informative summary that is rich enough and coherent enough to serve as a document surrogate (e.g. an executive summary) is beyond the state of the art for deployable general-purpose systems. For a system to produce a readable and accurate summary, it needs to analyze the text to a level that is sufficiently deep to determine the relative importance of the information presented, and generate coherent, readable output. These tasks encompass discourse understanding, abstraction, and language generation, all of which push the envelope in natural language processing.

The strategies employed in text analysis fall into three general categories. Using *word frequency* corresponds to the notion that important things are mentioned more often than less important things. Analyzing the structure, cohesion, and coherence (*discourse)* of the text captures the idea that important things occur in different contexts than unimportant things. Building a domain *knowledge model* exploits knowledge of what is important in the domain in the first place. Systems can be classified by their primary analysis strategy as being frequency-based, discourse-based, or knowledge-based. For generation, systems either extract sentences from the text, generate text from an abstract representation, or, as in message understanding systems, produce an instantiated template.

The earliest attempts at automatic text summarization (Luhn, 1958; Edmundson, 1969; Rush, Salvador, and Zamora, 1971) relied on the frequency of words, their proximity, and their location to determine the important parts of the text. Of the heuristics that were found to be most reliable for locating material for a summary were discourse cues specific to the domain or genre: so-called *cue words*, *cue phrases*, or *indicators*, such as "in conclusion." Least reliable was word frequency. Recent improvements include combining feature sets using classification techniques (Kupiec, et al., 1995), and application of information retrieval indexing techniques (first proposed by Brandow, et al., 1995; followed by Aone, et al., 1997) to find *signature*

words in the document, a principled improvement on the older intuitive but disappointing frequency measure.

Knowledge-intensive methods, relying on rich domain knowledge for text analysis, have been successful only in restricted domains (see, for example, Paice and Jones, 1993; Jacobs and Rau, 1990; DeJong, 1982; Reimer and Hahn, 1988; Tait, 1985; Riloff, 1995). These systems exploit knowledge of the domain to build conceptual representations of the text. A message understanding system, which instantiates a template, is similar in approach and might also be considered to be a summarizer. From the rich conceptual representation, there is more than one possible strategy for creating the output summary. The SUMMONS system (McKeown and Radev, 1995), which summarizes multiple news stories on the same event, generates a summary from a template representation; Maybury (1995) describes a number of methods for selecting events and presenting event summaries.

Work in discourse-based approaches to summarization has been motivated by the lack of coherence in the sentence-extraction approaches. Most of this work attempts to identify the best cohesive sentence candidates (Paice, 1990; Johnson, et al., 1993) or the best paragraphs for representing the discourse structure of the text (Miike et al., 1994). Both approaches parse the text and analyze discourse relations and in the end select sentences for extraction.

For the most part, the frequency-based approaches are inexpensive and shallow and do not depend on deep knowledge of the domain, or on discourse processing, although sometimes elements of each are brought to bear. These systems avoid the complexities of full-scale analysis and generation NLP by defining the abstraction problem as one of sentence extraction. This approach assumes that there is a set of sentences in the document that is representative of its contents. The summarization task becomes one of applying a scoring mechanism to find the most salient sentences, and the summary generation task becomes one of concatenating the sentences together. The resulting summary is not guaranteed to be coherent; in fact, it probably is not. Various techniques are used to improve on coherence, e.g., identification of backward references (anaphora) such as pronouns, adverbs (e.g. *here*) definite noun phrases (e.g. *the man*, as opposed to *a man*, refers to something earlier in the text). Identification of anaphora allows a system to either eliminate sentences containing unresolved anaphoric references or add in preceding ones to resolve them. Sometimes an extract *seems* to be coherent, but is, in fact incomplete and misleading (see Boguraev, et al., 1998, for discussion). While our summarizer does not completely escape the problems of incoherence and incompleteness, those issues are less critical because summary and text appear together, and because the reader can spot salient words in the gap between summary sentences. Further, since the summarized sentences are actively linked to the document text in the "active summary" model, the reader can quickly determine their context.

The optimal length of a document extract that is as informative as the full document is reported to be about 20% (Morris, 1992). It has become the common wisdom that still shorter extracts may be useful for indicative summaries; in fact, the on-line environment and the amount of real estate available on the monitor demand brevity. Search engines such as Lycos display only the first few hundred characters in the text as a “summary”; we have found experimentally that we need at least four sentences to get a subjectively acceptable extract. It seems that the developers of Verity have arrived at the same conclusion.

Evaluating summarization results is not trivial, and is currently a hot topic. There is evidence that the optimal extract is not unique (Rath, et al., 1961; Chen et al., 1992). The purpose of the extract varies; human extractors vary. Sentence extraction systems are evaluated by comparing the extract with sentences selected by human subjects (Rath, et al.1961; Edmundson, 1969; Kupiec et al., 1995), an objective measure on the surface that ignores the possibility of multiple right answers; or the extract is rated for summary acceptability (Brandow, et al.), a subjective measure that is even less satisfying. Other evaluation protocols are task-based, comparing user performance using abstracts and full-text originals in terms of browsing and search time (Miike et al., 1994; Sumita, et al., 1993); recall and precision in document retrieval (Brandow, et al.); or recall, precision, and time required in document categorization (SUMMAC 1998, see Hand, 1997). Our summarizer has been tested using all three evaluation methods (see Neff, 1998).

Our work takes as a starting point the work of Kupiec, et al. (1995) and Brandow et al. (1995) and leverages our earlier work in text analysis, information extraction, and corpus-based statistical NLP. The summarizer uses the sentence extraction approach but brings a richer source of domain knowledge and discourse structure than most other frequency approaches. We acknowledge here its strong resemblance to DimSum (Aone, et al., 1997) in that both systems exploit many of the same kinds of text analysis tools and functions.

Summarizer system description

Our summarization system is based on a representation of the text produced by the Textract information extractors. A document structure builder produces a structural representation of the document, identifying sections, headings, paragraphs, tables, etc. Currently, it is rudimentary, preferring text with structural tags to text with white space cues; however, there are plans to make it more robust. Textract locates, counts, and extracts items of interest, such as names, multiword terms, and abbreviations, allowing related (but not identical) items to be counted together. Summarizer compares the frequency of the vocabulary items found in the text (including also single words but ignoring stop words) to the frequency of the same vocabulary in the collection vocabulary, using a *tf*idf* measure (proposed by Brandow, et al. (1995), adapted from Salton and McGill (1993)).

Simply described, this version of *tf*idf* (term frequency times inverted document frequency) measures how much more frequent, relatively, a term is in the document than it is in the collection. Items whose *tf*idf* exceeds an experimental threshold are identified as signature terms. Further, items occurring in the title and in headings are added to the list of signature terms, regardless of their *tf*idf*. The score for a sentence (simplified here) is a function of the sum of the *tf*idf*'s of the signature words in it, how near the beginning of the paragraph the sentence is, and how near the beginning of the document its paragraph is. Sentences with no signature words get no "location" score; however, low-scoring or non-scoring sentences that immediately precede higher-scoring ones in a paragraph are promoted under certain conditions. Sentences are disqualified if they are too short (five words or less) or contain direct quotes (more than a minimum number of words enclosed in quotes). Documents with multiple sections are a special case. For example, a longer one with several headings or a news digest containing multiple stories must be treated specially. To ensure that each section is represented in the summary, its highest scoring sentences are included, or, if there are none, the first sentence(s) in the section.

Although earlier researchers (e.g. Brandow, et al.) have asserted that morphological processing and identification of multi-words would introduce complication for no measurable benefit, we believe that going beyond the single word alleviates some of the problems noted in earlier research. For example, it has been pointed out (Paice, 1990) that failure to perform some type of discourse processing has a negative impact on the quality of a generated abstract. Some discourse knowledge can be acquired inexpensively using shallow methodology. Morphological processing allows linking of multiple variants of the same word. Our name identifier, Nominator (Ravin and Wacholder, 1996) distinguishes between *bill* and *Bill*, thus reducing noise in the frequency counting. Further, its ability to identify *Bill Clinton* and *Clinton* as variants of the same name boosts the frequency of the concept (and its *tf*idf*) in the document. The interaction of Nominator with Abbreviator allows recognition of *American Bar Association* and its variant *ABA* as referring to the same thing. Our term identifier (Justeson & Katz 1995) recognizes multi-word concepts like *interest rate*. The interaction of Nominator with Terminator finds *Treasury bill*, *Java script*, and *Alzheimer's disease*.

Textract applied to the document gives us some knowledge of the document; Textract applied to a collection gives us some knowledge of the domain. An open question is the size of the collection that is needed for good results. While Brandow, et al. (1995) used a corpus of some 70 megabytes of text, we have obtained satisfactory results for summarization using less. We plan further research into this question.

Keyword Extraction

The keyword list that Active Markup displays at the top of the document is simply a list of the **n** items found by Textract that had the highest *tf*idf* score. The number of items to display at the top of document is a parameter that the client passes to the server.

Relations Among Keywords

The most promising area for improvements to Summarizer lies in the area of richer discourse analysis. Recognition of discourse antecedents (anaphora), following Kennedy and Boguraev (1996) will have two effects. On the one hand, it will affect the *tf*idf* of items once considered different that will now be the same. On the other hand, it will improve summary coherence. A related issue is an improvement to the Term recognizer to identify single-word variants of multiword terms (e.g. determining when to equate: "the *Java script* … the *script*"), something that Nominator already does for names. Text segmentation, or identification of topic shifts (e.g. Hearst, 1994), will improve identification of the most important material, particularly in multi-story documents or in feature articles or magazine articles, which typically begin with an anecdotal attention-getter and arrive at the statement of the topic only later.

Additional enhancements to the user interface will

include the ability to select and navigate through document space based on multiple term selections, and a simple set of tools for adjusting the summary length. In addition, we hope to apply the graphical layout tools we have previously described for term navigation to visualizing document relations and clustering.

Summary

Document summarization is a complex and continuously evolving field. One major approach to summarization is the extraction of a number of sentences containing terms which our text extraction techniques regard as important in that document. Since such sentence-based extraction may not always find all of the major concepts in the document, the visual interface between the summary and the document becomes extremely important in helping the user scan the document quickly without having to read all of it. In the system we describe here, the sentences in the summary are hyperlinked to those in the actual document. In addition the same extraction techniques provide a powerful new visual metaphor called Active Markup. Active markup allows the user to investigate both lexical and document space around the documents returned from an initial query without having to enter or modify any more queries.

Acknowledgements

We are deeply indebted to our colleagues at IBM Research for their collaboration on various aspects of the underpinnings of this system. These include Yael Ravin and Roy Byrd for their substantial contributions to the Textract system, Zunaid Kazi for his refining of the algorithms for unnamed relations extraction, Herb Chong and Aaron Kershenbaum for the original dictionary work we have extended here, and Alan Marwick for his leadership.

References

Aone, Chinatsu, Mary Ellen Okurowski, James Gorlinsky, Bjornar Larsen, 1997. A scalable summarization system using robust NLP. *Intelligent Scalable Text Summarization, Proceedings of a Workshop Sponsored by the Association for Computational Linguistics*, 11 July, 1997, 66 73.

Boguraev, Branimir K., Rachel Bellamy, and Christopher Kennedy, 1998. Dynamic Presentation of Phrasally Based Document Abstractions, unpublished paper.

Brandow, Ron, Karl Mitze, and Lisa Rau, 1995. Automatic condensation of electronic publications by sentence selection. *Information Processing and Management*, 31, No. 5, 675-685.

Chen, F.R. and M.M. Withgott, 1992. The use of emphasis to automatically summarize a spoken discourse. In *Proceedings of the IEEE Intl. Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 229-232.

Cooper, James W. and Byrd, Roy J. “Lexical Navigation: Visually Prompted Query Expansion and Refinement.” Proceedings of DIGLIB97, Philadelphia, PA, July, 1997.

Cooper, James W. and Byrd, Roy J., OBIWAN - A Visual Interface for Prompted Query Refinement, Proceedings of HICSS-31, Kona, Hawaii, 1998.

Cooper, James W., Principles of Object Oriented Programming Using Java 1.1, Ventana, 1997.

DeJong, G. An overview of the FRUMP system. In W.G. Lehnert and M.H. Ringle, editors, *Strategies for Natural Language Parsing*, pages 149-176, 1982.

Edmundson, H.P., 1969. New methods in automatic abstracting. *Journal of the ACM*, 16(2):264-285.

Hand, Therese Firmin, 1997. A proposal for task based evaluation of text summarization systems. *Intelligent Scalable Text Summarization, Proceedings of a Workshop Sponsored by the Association for Computational Linguistics*, 11 July, 1997, 31-38.

Hearst, Marti, 1994. Multi-paragraph segmentation of expository text. *In 32nd Annual Meeting of the Association for Computational Linguistics*, Las Cruces, New Mexico.

Jacobs, P. and Rau, L., 1990. SCISOR: Extracting information from on-line news. *Communications of the ACM.*, 33(11) 88-97.

Jing, Y. and W. B. Croft "An association thesaurus for information retrieval", in *Proceedings of RIAO* 94, 1994, pp. 146-160.

Johnson, F.C., C.D. Paice, W.J. Black, and A.P. Neal, 1993. The application of linguistic processing to automatic abstract generation. *Journal of Documentation and Text Management*, 1(3):215-241.

Justeson, J. S. and S. Katz "Technical terminology: some linguistic properties and an algorithm for identification in text.” *Natural Language Engineering*, **1**, 9-27, 1995.

Kennedy, Christopher and Branimir Boguraev, 1996. Anaphora for everyone: Pronominal anaphora resolution without a parser. In *Proceedings of COLING-96 (16th International Conference on Computational Linguistics)*, Copenhagen, Denmark.

Kupiec, Julian, Jan Pedersen, and Francine Chen, 1995. A trainable document summarizer. In *Proceecings of the 18th Annual International SIGIR Conference on Research and Development in Information Retrieval*, 68-73.

Luhn, H.P., 1958. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2, 159-165.

Maybury, Mark T., 1995. Automated even summarization techniques. In B. Endres-Niggemeyer, J. Hobbs, and Karen Sparck Jones, editors, *Summarizing Text for Intelligent Communication*, pages 101-149.

McKeown, Kathleen and Dragomir Radev, 1995. Generating summaries of multiple news articles. In *Proceedings of the 18th Annual International SIGIR Conference on Research and Development in Information Retrieval*, pages 74-78.

Miike, Seije, Etsuo Itho, Kenji Ono, and Kazuo Sumita, 1994. A full text retrieval system with a dynamic abstract generation function. *In Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 152-161.

Morris, A.G., Kasper, G. M., and Adams, D. A. The effects and limitations of automated text condensing on reading comprehension performance. *Information Systems Research*, pages 17-35, March 1992

Morris, A.H., G.M. Kasper, and D.A. Adams, 1992. The effects and limitations of automated text condensing on reading comprehension performance. *Information Systems Research*,17-35.

Neff, Mary S., 1998. Document Summarization for Active Markup, *IBM Research Report*, to appear.

Paice, C., 1990. Constructing literature abstracts by computer: Techniques and prospects. *Information Processing and Management*, 26(1): 171-186.

Paice, C.D. and P.A. Jones, 1993. The Identification of important concepts in highly structured technical papers. In R. Korfhage, E. Rasmussen, and P. Willet, eds, *Proceedings of the Sixteenth Annual International ACM Sigir Conference on Research and Development in Information Retrieval*, pages 69-78. ACM Press.

Rath, C.J, A. Resnick, and T.R. Savage, 1961. The formation of abstracts by the selection of sentences. *American Documentation*, 12(2):139-143.

Ravin, Y. and Wacholder, N. 1996, “Extracting Names from Natural-Language Text,” IBM Research Report 20338.

Reimer, U. and U. Hahn, 1988. Text condensation as knowledge base abstraction. In *IEEE Conference on AI Applications*, pages 338-344, 1988.

Riloff, Ellen, 1995. A corpus-based approach to domain-specific text summarization. In B. Endres-Niggemeyer, J. Hobbs, and K. Sparck Jones, editors, *Summarizing Text for Intelligent Communication*, pages 69-84.

Rush, J. E., Salvador, R., and Zamora, A., 1971. Automation abstraction and indexing: Production of indicative abstracts by application of contextual inference and syntactic criteria. *Journal of the American Society for Information Science*, 22(4): 260-274.

Salton, Gerald and M. McGill, editors, 1993. *An Introduction to Modern Information Retrieval*. McGraw-Hill.

Sumita, Kazuo, Ono, Kenji, and Miike, Seiji, 1993. Document structure extraction for interactive document retrieval systems. *Proceedings of SIGDOC*, 1993, 301-310.

Tait, J.I., 1985. Generating summaries using a script-based language analyzer. In L. Steels and J.A. Campbell, editors, *Progress in Artificial Intelligence*, pages 312-318, Ellis Horwood, 1985.