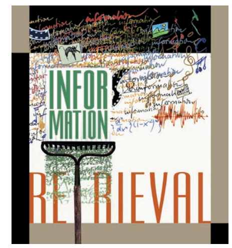
The Challenges of Automatic Summarization



Researchers are investigating summarization tools and methods that automatically extract or abstract content from a range of information sources, including multimedia.

Udo Hahn
Albert Ludwigs University

Inderjeet Mani
Mitre Corp.

Summarization—the art of abstracting key content from one or more information sources—has become an integral part of everyday life. People keep abreast of world affairs by listening to news bites. They base investment decisions on stock market updates. They even go to movies largely on the basis of reviews they've seen. With summaries, they can make effective decisions in less time.

Although some summarizing tools are already available, with the increasing volume of online information, it is becoming harder to generate meaningful and timely summaries. Tools such as Microsoft's AutoSummarize option in Office 97, IBM's Intelligent Text Miner, Oracle's Context, and Inxight's Summarizer (part of Alta Vista's search tools) are useful, but their application is limited to *extraction*—selecting original pieces from the source document and concatenating them to yield a shorter text. *Abstraction*, in contrast, paraphrases in more general terms what the text is about.

The concatenation approach to extraction does little to ensure that the summary is coherent, which can make the text hard to read. Moreover, the source may not always have text—for example, a sports event on videotape or tables displaying economic data—and current tools cannot summarize nontextual media. Finally, these tools do not currently handle multiple sources. For example, there may be many stories on the Web about a particular news event, and it would be useful if the summarizer could capture common and new information.

To address these limitations, researchers are looking at a variety of approaches, which roughly fall into two categories. *Knowledge-poor* approaches rely on not having to add new rules for each new application domain or language. *Knowledge-rich* approaches assume that if you grasp the meaning of the text, you can reduce it more effectively, thus yielding a better summary. They rely on a sizeable knowledge base of rules, which must be acquired, maintained, and then adapted to new applications and languages. The two classes of methods are not necessarily mutually exclusive. In fact, some approaches use a hybrid.

In both methods, the main constraint is the compression requirement. Extracts of single documents usually aim to be five to 30 percent of the source length. However, compression targets in summarizing multiple sources or in providing summaries for handheld devices are much smaller. These high reduction rates pose a challenge because they are hard to attain without a reasonable amount of background knowledge.

Another challenge is how to evaluate summarizers. If you are to trust that the summary is indeed a reliable substitute for the source, you must be confident that it does in fact reflect what is relevant in that source. Hence, methods for creating and evaluating summaries must complement each other.

HOW SUMMARIES DIFFER

At the most basic level, summaries differ according to whether they are extracts or abstracts. An extract of Abraham Lincoln's Gettysburg Address may begin with "Four score and seven years ago our fathers brought forth upon this continent a new nation." An abstract of the same material may include a quotation along with the paraphrase: "This speech by Abraham Lincoln commemorates soldiers who laid down their lives in the Battle of Gettysburg." Both kinds of summarization have two core tasks: determine what is salient (or relevant or important) in the source being

0018-9162/00/$10.00

summarized and decide how to reduce (or condense or abridge) its content. But within and across these two categories, summaries differ according to function[1] and target reader. For example, a summary can be indicative, informative, or critical:

- *Indicative* summaries follow the classical information retrieval approach: They provide enough content to alert users to relevant sources, which users can then read in more depth.
- *Informative* summaries act as substitutes for the source, mainly by assembling relevant or novel factual information in a concise structure.
- *Critical* summaries (or reviews), besides containing an informative gist, incorporate opinion statements on content. They add value by bringing expertise to bear that is not available from the source alone. A critical summary of the Gettysburg Address might be: "The Gettsyburg Address, though short, is one of the greatest of all American speeches, with its ending words being especially powerful—'that government of the people, by the people, for the people, shall not perish from the earth.'"

A summary can also be generic or user-focused. *Generic* summaries address a broad community; there is no focus on special needs because the summarizer is not targeting any particular group. *User-focused* summaries, in contrast, are tailored to the specific needs of an individual or a particular group (children, for example). A user-focused extract of the Gettysburg Address aimed at someone interested in the Civil War might include: "Now we are engaged in a great civil war....We are met on a great battlefield of that war."

Until recently, generic summaries were more popular, but with the prevalence of full-text searching and personalized information filtering, user-focused summaries are gaining importance. Many tools support both user-focused and generic summarization.

METHODS AND ARCHITECTURES

The summarization process has three phases: analyzing the source text, determining its salient points, and synthesizing an appropriate output. Most current work focuses on the more developed technology of summarizing a single document.

Extraction

The emphasis in extraction methods is usually on determining salient text units (typically sentences) by looking at the text unit's lexical and statistical relevance or by matching phrasal patterns. Synthesis consists of concatenating original parts of the source.

Most methods adopt a linear weighting model. At the core of the analysis phase in this model is a scheme that weights each text unit according to such features as the unit's location in the source text, how often it occurs in the source text, appearance of cue phrases, and statistical significance metrics. The sum of these individual weights, usually modified by specific tuning parameters attached to the weights, is the overall *weight* of the text unit *U*:

Weight(U) := Location(U) + CuePhrase(U) + StatTerm(U) + AddTerm(U)

The model determines *location* weight according to whether the text unit is in the initial, middle, or final position in a paragraph or the entire docu-

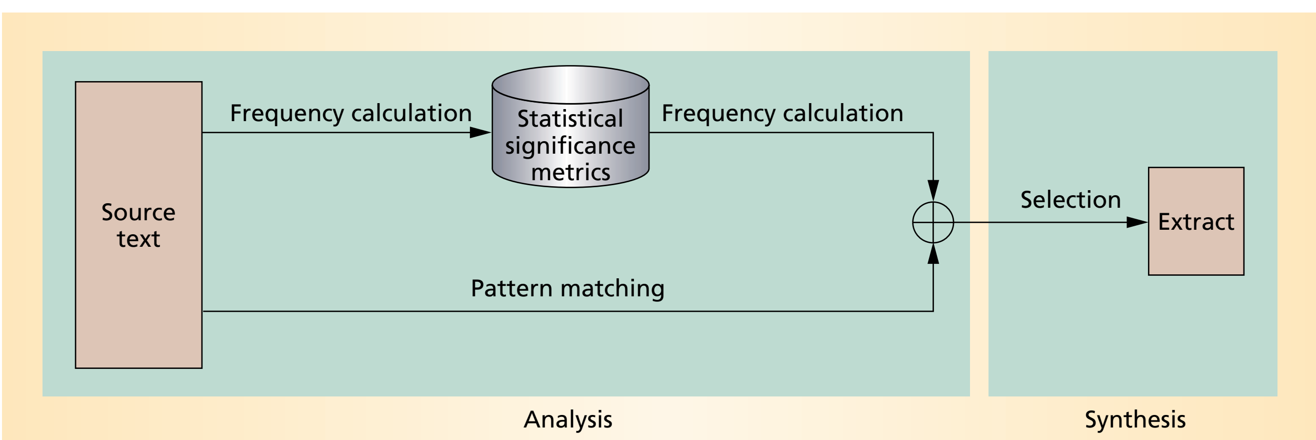


Figure 1. Architecture for extraction (knowledge-poor summarization). The analysis phase Processes each sentence from a source in turn. A sentence's weight is based on statistical significance metrics (from term frequency counts and pattern-matching operations), the presence of specific terms, and the sentence's location. The sentence weighting from the analysis phase is fed directly to a synthesis component, which extracts the top-weight sentences on the basis of compression rate.

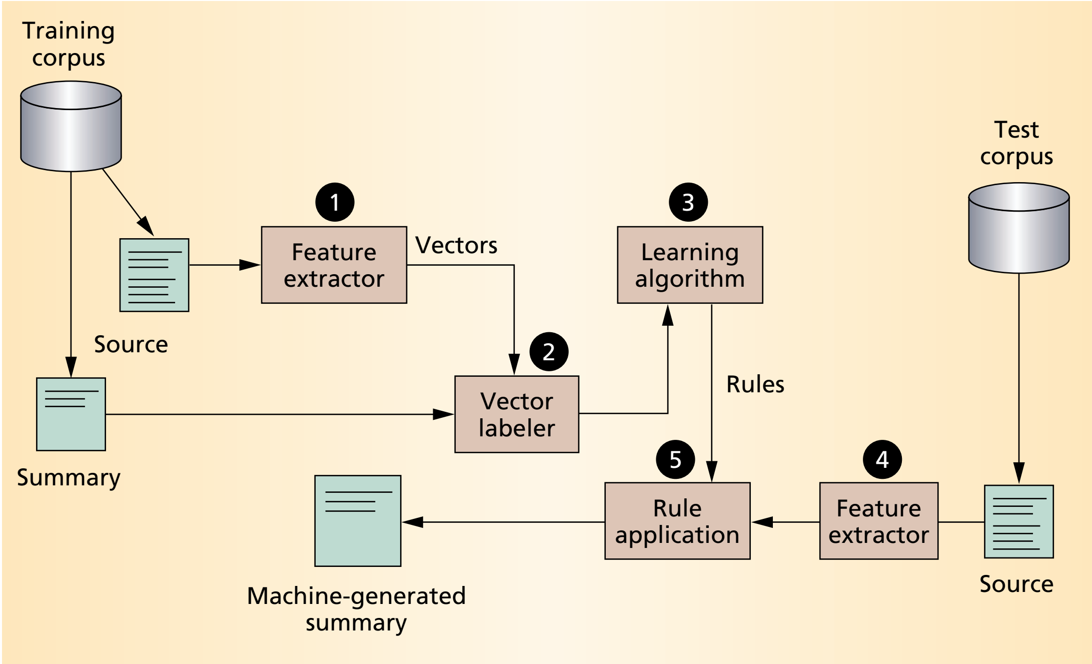


Figure 2. A classifier that learns how to summarize. (1) During training, a vector of features represents each sentence in the source. (2) The classifier labels each vector according to similarity of content between the sentence and the abstract. It then feeds the set of training examples to (3) a learning algorithm that learns the classification rules for determining whether or not a sentence should be part of the summary. During testing, the summaries are absent. Instead, the classifier (4) turns each sentence from the test corpus into a feature vector and (5) matches it against the learned rules to generate the extract.

ment, or whether it occurs in prominent sections, such as the document's introduction or conclusion.

The *cue phrase* is a lexical or phrasal summary cue such as "in conclusion," "in this paper," "our investigation has shown," or "a major result is." The cue phrase weight can also be based on domain-specific bonus terms like "excellent" (higher weight) and stigma terms like "unimportant" (much lower weight).

The model also assigns a weight according to the unit's *statistical salience (StatTerm)*. Statistical salience is based on metrics from studies of automatic indexing, where researchers have investigated and validated a variety of term-weighting measures to help discriminate a document from others in a collection. One prominent group of metrics, *tf.idf measures*, for example, balances a term's frequency in the document against its frequency in a collection (usually along with other frequency and length normalization measures).

Finally, the model looks at the terms in the text unit and weights the unit according to the terms' *additional presence (AddTerm)*—do the terms also appear in the title, headline, initial paragraph, or user's profile or query? Favoring terms related to the user's interest is one way to tailor the summary to a particular user.

Figure 1 shows a general architecture for knowledge-poor summarization. The analysis phase characterizes the linear weighting model in terms of a series of frequency calculations and string- or pattern-matching operations, which, for each text unit, compute the weights for the four different feature types (Location, CuePhrase, StatTerm, AddTerm) for that text unit. It then sums these weights for each text unit, selecting the *n* best units (*n* could also be determined from the compression rate) for inclusion in the extract.

Most extraction systems still use the approach in Figure 1, which dates back to foundational research in the 1960s and 1970s.[2] Researchers who have carried out feature comparison studies to evaluate the model's performance[3] have found that the text unit's location tends to be one of the most useful features, especially when combined with the cue phrase feature.

In many systems, the user sets the tuning parameters manually, and parameter selection tends to be ad hoc because the relative contribution of different features can vary among text genres. In an attempt to automate this process and possibly improve its performance, researchers such as Julian Kupiec and others at Xerox Parc have developed a classifier that learns how to extract. Figure 2 shows how this classifier uses a collection of human-generated summaries and their corresponding full text sources to automatically learn criteria for adequate extraction.[3]

The corpus-based method, which the Inxight summarizer uses, is suitable for different text genres, but only if users have a corpus of both the full text source and associated summaries for that genre. Researchers are pushing hard to make such corpora available.

Overall, the attractiveness of the linear feature model lies in its easy implementation. However, extracting sentences (or paragraphs) without considering the relations among them can result in incoherent summaries. Sentences may be missing or there may be dangling anaphors (a word or phrase that takes its reference from another word or phrase). For example, if an argument extends across two sentences and

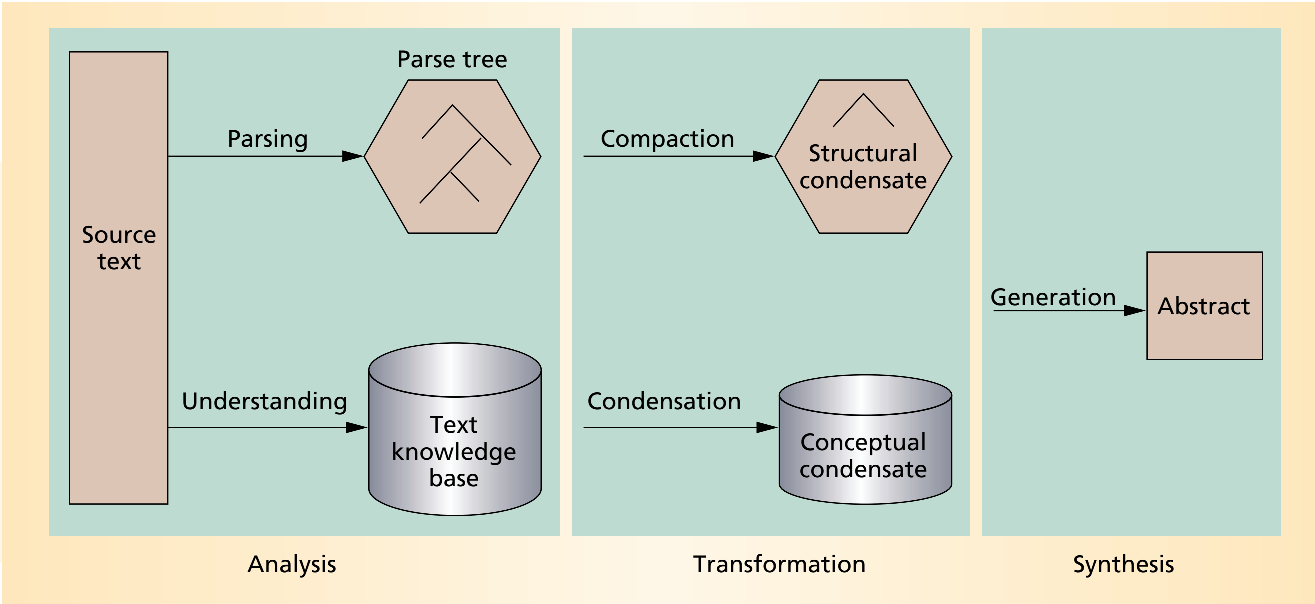


Figure 3. Architecture for abstraction (knowledge-rich summarization). Abstraction has two basic approaches that methods often combine. In the first approach (top), the analysis phase creates a parse tree, which the transformation phase compacts by eliminating or regrouping parts of trees according to structural criteria such as parenthetical phrases or subordinate clauses. The result is a structural condensate, a less complex structural representation. In the second approach (bottom), the analysis phase assembles conceptual representations of the document's content into a text knowledge base. The transformation phase compresses the knowledge base by aggregating or generalizing information. The result is a conceptual condensate, a less detailed conceptual representation. The synthesis phase for both approaches generates a natural language abstract.

only one of them is extracted, the argument will appear fragmented in the summary and thus will be incomprehensible or biased. The following text fragment illustrates a problem with dangling anaphors: "Bill Dixon joined Procter & Gamble in 1994. In 1996, *he* became vice president of *the company*." If the extract contained only the second sentence, the anaphoric references—"he" (Bill Dixon) and "the company" (Procter & Gamble)—would have to be resolved for the text to be coherent and informative.

A variety of research activities are attempting to address this problem, mostly by patching the summary. Some methods include a window of previous sentences when they detect a gap or an anaphor. Others exclude sentences that contain anaphoric references or attempt to resolve or readjust anaphoric references through a shallow linguistic analysis. Compression can be lost with this approach, however, because it introduces extraneous material. Moreover, since the core summary is already set, it is hard to recover the original compression percentage at this stage.

Abstraction

When Calvin Coolidge was asked what a clergyman had said in his sermon on sin, Coolidge replied, "He said he was against it."[4] This response shows the powerful intuition underlying abstraction—that grasping the meaning will let a person identify the essence of a text more effectively and thus produce a better summary.

Unlike the linear model in extraction methods, abstraction requires using heavy machinery from natural language processing (NLP), including grammars and lexicons for parsing and generation. It also requires some commonsense and domain-specific ontologies for reasoning during analysis and salience computation.

As Figure 3 shows, abstraction has two basic approaches. The first (top of figure) uses a traditional linguistic method that parses sentences syntactically. This method can also use semantic information to annotate parse trees. Compaction procedures operate directly on these trees to eliminate and regroup parts of them—for example, by pruning subtrees according to structural criteria such as parentheticals and embedded relative clauses or clause subordination. After compaction, the original parse tree is considerably simpler—becoming in essence a *structural* condensate.

The second abstraction approach has its roots in artificial intelligence and focuses on natural language understanding.[5] Syntactic parsing is also part of analysis, but the results are not parse trees. Rather, they are conceptual representation structures of the entire source content, which are assembled in a text knowledge base. The structures can be predicate calculus formulas or representations such as a semantic network or a collection of frames. An example is a template for a banking transaction (a prespecified event) that lists the institutions and customers involved, date, amount of money transferred, type of transaction, and so on.

The transformation phase in Figure 3 is unique to knowledge-rich(er) abstraction approaches. Transformation alters the conceptual representation in several ways. It eliminates redundant or irrelevant information by removing overly detailed assertions or pruning conceptual subgraphs. It also further aggregates information by merging graphs (or templates) or by generalizing information, for example, using taxonomic hierarchies of subclass relations. To aid transformation, researchers have proposed inference-based methodologies such as macro rules that operate on logical assertions[6] or operators that determine characteristic activity and connectivity patterns in a text knowledge base.[7] The transformation phase yields a conceptual representation structure of the summary—in essence, the *conceptual* condensate.

These formal representation layers (structural and conceptual condensates) are what set a knowledge-rich approach apart from a knowledge-poor one.

As Figure 3 shows, the synthesis phase is the same for both approaches: A text generator translates the structural or conceptual representation to produce a fluent natural language abstract. In some systems, the user can inspect the condensate directly via a point-and-click interaction, without the generation step, provided the source text units associated with the condensate are available.

This type of inference-based summarization relies on prespecified knowledge structures that tell the summarizer a priori which concept is more specific than another, or which conceptual properties (roles or slots) a concept has. The summarization explicitly encodes semantic information into the links between nodes in a concept graph—for example, as taxonomic (subclass or instance-of) or metonymic (part-of) relations. In this way, it imposes direction and selectivity on the search or reasoning procedures. Summary-oriented inference rules or general inference schemes (such as terminologic classification) use this information to distinguish what is relevant, which lets them traverse generalization hierarchies and collapse concept subgraphs as needed. Figure 4 shows the basis for this collapsing process, or *generalization-based condensation*.

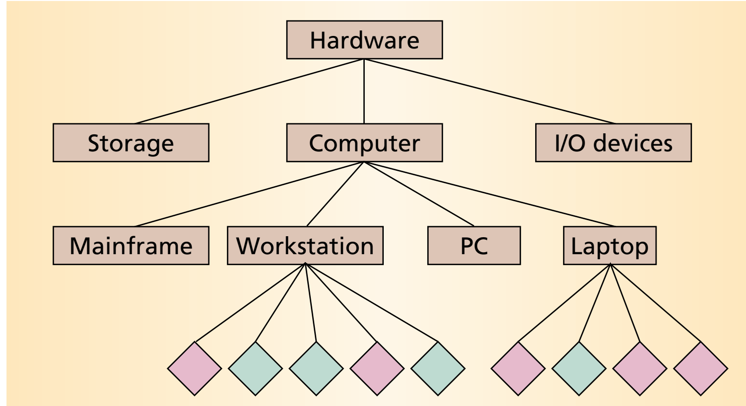


Figure 4. How generalization-based condensation works. A text knowledge base consists of concept classes (such as Laptop) and instances of them (such as a specific product that is a laptop). Instances that have an activation weight greater than zero (the value of a weight reflects the number of references to the concept in the text) are considered active. A concept is salient if the ratio of the number of its instances to the number of its active instances is less than the number of its active instances. In the diagram, the concept class Workstation is salient because three of its five immediate instances are active (green squares). Laptop, on the other hand, is not salient because only one of its instances is active. The ability to detect salient concepts means that the summarizer can generalize to more abstract characterizations of the text content. For example, it might cautiously generalize the main topic from Workstation to Computer or even to Hardware.

Extraction versus abstraction

Extraction approaches are easy to adapt to larger sources. Because they are limited to the extraction of passages, sentences, or phrases, however, the resulting summaries may be incoherent. Abstraction approaches, on the other hand, provide more sophisticated summaries, which often contain material that enriches the source content. Because they are based on a formal representation of the document's content, they adapt well to high compression rates, such as those needed for wireless personal digital assistants (PDAs) and similar technologies.

Systems based on the knowledge-rich paradigm vary in their requirements. Template-filling approaches work only for text centered on a particular template, although summarizers can use statistical techniques to implement them in the analysis phase. In general, however, knowledge-rich methods demand full-blown knowledge sources—a requirement that has hampered their widespread application. Recent trends in corpus-based NLP are providing broad-coverage parsers, comprehensive lexical resources (such as WordNet), and ontologies (such as CYC or the Penman Upper Model). There is also a wealth of corpora for training NLP systems, including plain text corpora such as that from *The Wall Street Journal* or grammatically annotated corpora such as the Linguistic Data Consortium's Penn Treebank. Finally, summarization efforts are becoming increasingly hybrid; researchers are successfully coupling statistical and knowledge-based methods to get the best of both worlds.

Evaluation methods

Summary evaluation methods attempt to determine how adequate (and reliable) or how useful a summary is relative to its source. At present, there are two types of evaluation methods. The first is *intrinsic* (or normative) evaluation in which users judge the quality of summarization by directly analyzing the summary. Users judge fluency, how well the summary covers stipulated key ideas, or how it compares to an ideal summary written by the author of the source text or a human abstractor. None of these measures are entirely satisfactory. The ideal summary, in particular, is hard to construct and rarely unique. Just as there are many ways to describe an event or a scene, users can produce many generic or user-focused extracts or abstracts that they consider acceptable. Indeed, empirical evidence shows that people rarely agree on which sentences or paragraphs a summary should include.[8]

The second type of evaluation method is *extrinsic*. Users judge a summary's quality according to how it affects the completion of some other task, such as how

Table 1. Relevance assessment using summaries, as opposed to full text.

Summary type	Length reduction	Time reduction	Accuracy loss
User-focused	77%	50%	5%
Generic	90%	60%	0%

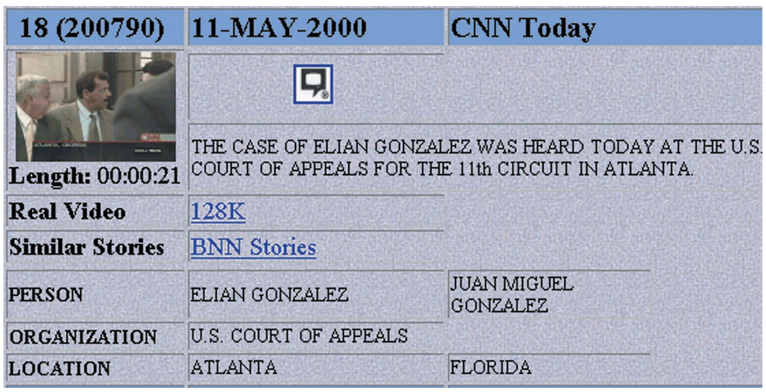


Figure 5. Multimedia summarization using the Broadcast News Navigator, which searches for, browses, and summarizes TV news broadcasts. The display shows a summary of the content of a video segment retrieved in response to a search engine query. The summary includes a key sentence along with the most salient people, organizations, and places mentioned in the closed-captioned text accompanying the video. Clicking on the video thumbnail brings up the video within a multimedia player. The system also offers a link to news stories it has judged to be similar.

well it helps them determine the source's relevance to topics of interest or how well they can answer certain questions relative to the full source text.

Recently, the US government conducted a large-scale evaluation of summarization systems as part of its Tipster program, which aimed to advance the state of the art in text-handling technologies.[9] The program involved two evaluations. In one session, each user saw either a source or a user-focused summary and had to decide whether it was relevant to a topic. In the other session, the users saw either a source or a generic summary and had to either select a topic (from among several presented) to which they felt the document was relevant or decide whether it was relevant to any topic. As Table 1 shows, automatic text summarization was very effective in this relevance assessment. Users could assess relevance as accurately from summaries, which discarded 77 to 90 percent of the source text, as from full text—but in almost half the time (the 5 percent accuracy difference was not statistically significant).

Although the evaluation did not test specific summarization methods, all 16 summarization systems evaluated used knowledge-poor methods. They differed in their ability to generate user-focused summaries; the most accurate user-focused systems displayed similar sentence extraction behavior.

NEW APPLICATION AREAS

At least four areas of summarization are becoming increasingly relevant. In all four, summarizers must be able to deal with a variety of document formats such as HTML and XML. They must also be able to exploit information in the tags associated with these documents. Developments in summarizations involving multiple languages and hybrid sources are less mature; the first practical prototypes are in multidocument and multimedia summarization.

Multiple languages

High-quality machine translation of unrestricted input (comparable to a human translator) is still out of reach. What is feasible and possibly useful for this type of summarization is a filtering mechanism. Users could apply such a filter to produce a monolingual summary that contains content from multilingual sources. They could then decide if they need more detailed translations.

Hybrid sources

In this application, summarization fuses information from formatted data and unformatted free text. An example is a summary that links the statistics for a baseball player from a database to news stories involving that player. This application is still very new, and little research is being done.

Multiple documents

This type of summarization extends the single-document methods to a document collection. The collection may range from gigabytes to bytes, so different methods may be needed for different sizes. Each method involves analyzing each document in the collection and then fusing information across documents in the transformation and synthesis phases. Summarizers still carry out elimination, aggregation, and generalization operations on representation structures, but across a collection of documents instead of from a single source. Simply concatenating summaries of each document will not suffice because there may be too many summaries, and they may contain redundant information.

Summarizers can identify similarities and differences among documents (what's common, what's unique, how they differ) by comparing and merging representations of document content from the analysis phase.[8,10,11] For example, using natural language generation, a summarizer can produce this summary from the template of a terrorist incident that has the same location as another, although the incidents come from different news sources:[11]

> On the afternoon of 26 February, Reuters reported that a bomb killed at least six people. However, AP later announced that only five people were killed.

Because the same news item often appears in slightly different forms in multiple news stories, summarizers have been developed that can eliminate redundant information across stories to provide a concise summary.[12] Other summarizers can track the threads of common topics across stories and present them using charts and graphs.[13]

Multimedia

Although research is still in a very early stage, the growing availability of multimedia facilities makes this possibly the most important new application for summarization. Techniques can leverage cross-media information during analysis or transformation, when fusing information across media, or during synthesis, when integrating information across media. Current methods exploit information from the audio or closed-captioned text (silence, speaker changes, anchor-reporter handoffs, and content analysis), as well as video (anchor and logo detection, for example) to help determine what is salient. The goal of one current project is to identify the content of videos—for example, by using pattern-recognition software to determine the parts that show interesting events (accidents, fights, appearance of main characters, and so on).[14]

Figure 5 shows a sample summary from the Broadcast News Navigator system,[15] a tool for searching, browsing, and summarizing TV news broadcasts. BNN uses a number of mixed-media presentation strategies, combining key frames extracted automatically from the video with summaries of the accompanying closed-captioned text along with key organizations, locations, and people involved. Advances in automatic speech recognition of audio sources might improve this kind of summarization.

At least for the short term, knowledge-poor approaches are likely to dominate applications, particularly when augmented with extraction learning mechanisms. Knowledge-rich approaches will begin to catch up and eventually replace extraction when we have reasonably sized grammars and domain knowledge sources. Providing these sources requires either large-scale knowledge engineering or more emphasis on machine-learning methods. Additional text (and summary) corpora will be required to make it feasible to empirically evaluate automatically generated summaries.

Overall, summarization research is still young. There is some consensus on the need for more evaluation, but many challenges remain, including the need to scale techniques for generating abstracts. Nevertheless, many of the techniques we describe here are already useful, and we expect summarization tools to be key in conquering the vast information universes ahead. ✷

Automatic Summarization Resources

- *AAAI Spring Symposium on Intelligent Text Summarization* (Stanford, Calif., March 1998); http://www.cs.columbia.edu/~radev/aaai-sss98-its/
- *ANLP/NAACL 2000 Workshop on Automatic Summarization* (Seattle, Wash., May 2000); http://www.isi.edu/~cyl/was-anlp2000/
- *COLING-ACL98 Tutorial on Text Summarization* (Montreal, Canada, August 1998); http://www.isi.edu/~marcu/coling-acl98-tutorial.html
- *Summarizing Text for Intelligent Communication Symposium* (Dagstuhl, Germany, 1993); http://www.ik.fh-hannover.de/ik/projekte/Dagstuhl/Abstract
- *TIPSTER Summarization Evaluation Conference* (Baltimore, Md., August 1998); http://www.itl.nist.gov/div894/894.02/related_projects/tipster_summac/index.html
- *Workshop on Intelligent Scalable Text Summarization* (Madrid, Spain, July 1997); http://www.cs.columbia.edu/~radev/ists97/
- *General resources:* http://www.cs.columbia.edu/~radev/summarization/; http://www.mitre.org/resources/centers/iime/aats.html

References

1. H. Borko and C.L. Bernier, *Abstracting Concepts and Methods*. Academic Press, New York, 1975.
2. R.E. Wyllys, "Extracting and Abstracting by Computer," *Automated Language Processing*, H. Borko, ed., John Wiley & Sons, New York, 1967, pp. 127-179.
3. J. Kupiec, J. Pedersen, and F. Chen, "A Trainable Document Summarizer," *Proc.18th Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, E.A. Fox, P. Ingwersen, and R. Fidel, eds., ACM Press, New York, 1995, pp. 68-73.
4. *The Giant Book of American Quotations*, Grammercy Books, New York, 1992, p. 209.
5. J. Hutchins, "Summarization: Some Problems and Methods," *Proc. Informatics 9: Meaning—The Frontier of Informatics*, K.P. Jones, ed., Aslib, London, 1987, pp. 151-173.
6. T.A. van Dijk, "Semantic Macro-Structures and Knowledge Frames in Discourse Comprehension," *Cognitive Processes in Comprehension*, M.A. Just and P.A. Carpenter, eds., Lawrence Erlbaum, Hillsdale, N.J., 1977, pp. 3-32.
7. U. Hahn and U. Reimer, "Knowledge-Based Text Summarization: Salience and Generalization Operators for Knowledge-Based Abstraction," *Advances in Automatic Text Summarization*, I. Mani and M. Maybury, eds., MIT Press, Cambridge, Mass., 1999, pp. 215-232.
8. G. Salton et al., "Automatic Text Structuring and Summarization," *Information Processing & Management*, Vol. 33, No. 2, 1997, pp. 193-207.
9. I. Mani et al., "The Tipster Summac Text Summarization Evaluation," *Proc. 9th Conf. European Chapter of the*

Assoc. for Computational Linguistics, Morgan Kaufmann, San Francisco, 1999, pp. 77-85.

10. I. Mani and E. Bloedorn, "Summarizing Similarities and Differences Among Related Documents," *Information Retrieval*, Vol. 1, No. 1, 1999, pp. 35-67.
11. D.R. Radev and K.R. McKeown, "Generating Natural Language Summaries from Multiple Online Sources," *Computational Linguistics*, Vol. 24, No. 3, 1998, pp. 469-500.
12. J.G. Carbonell and J. Goldstein, "The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries," *Proc. 21st Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, ACM Press, New York, 1998, pp. 335-336.
13. R.K. Ando et al., "Multidocument Summarization by Visualizing Topical Content," *Proc. ANLP/NAACL 2000 Workshop on Automatic Summarization,* 2000, pp. 79-88*;* http://www.isi.edu/~cyl/was-anlp2000/.
14. R. Lienhart, S. Pfeiffer, and W. Effelsberg, "Video Abstracting," *Comm. ACM*, Vol. 40, No. 12, 1997, pp. 54-62.
15. A. Merlino, D. Morey, and M. Maybury, "Broadcast News Navigation Using Story Segments," *Proc. ACM Multimedia 97*, ACM Press, New York, 1997, pp. 381-391.

Udo Hahn *is an associate professor for natural language processing at Albert Ludwigs University, Freiburg, Germany. His research focus is on the intersection of text understanding and information systems and includes text summarization, intelligent text retrieval, acquisition of knowledge from texts, and text mining. He was involved in the development of a large-scale German-language text summarization system (Topic) and has written or coauthored four books and more than 140 journal articles, contributions to collected volumes, and proceedings papers. He received a PhD in information science from Constance University in Germany. Contact him at hahn@coling.uni-freiburg.de.*

Inderjeet Mani *is a senior principal scientist in Mitre Corp.'s Information Systems and Technology Division and an adjunct faculty member in Georgetown University's Linguistics Department. His research has addressed problems related to machine translation, nonliteral language, proper names, temporal expressions, lexical semantics, and information retrieval. His work on automatic text summarization includes numerous papers and two books:* Automatic Summarization *(John Benjamins, to appear), and a coedited volume,* Advances in Automatic Text Summarization *(MIT Press, 1999). Contact him at imani@mitre.org.*