For any of these answer-extraction methods, the exact answer phrase can just be presented to the user by itself. In practice, however, users are rarely satisfied with an unadorned number or noun as an answer; they prefer to see the answer accompanied by enough passage information to substantiate the answer. Thus, we often give the user an entire passage with the exact answer inside it highlighted or boldfaced.

23.2.4 Evaluation of Factoid Answers

A wide variety of techniques have been employed to evaluate question-answering systems. By far the most influential evaluation framework has been provided by the TREC Q/A track first introduced in 1999.

Mean reciprocal rank
MRR

The primary measure used in TREC is an **intrinsic** or **in vitro** evaluation metric known as **mean reciprocal rank**, or **MRR**. As with the ad hoc information retrieval task described in Section 23.1, MRR assumes a test set of questions that have been human-labeled with correct answers. MRR also assumes that systems are returning a short **ranked** list of answers or passages containing answers. Each question is then scored according to the reciprocal of the **rank** of the first correct answer. For example if the system returned five answers but the first three are wrong and hence the highest-ranked correct answer is ranked fourth, the reciprocal rank score for that question would be $\frac{1}{4}$. Questions with return sets that do not contain any correct answers are assigned a zero. The score of a system is then the average of the score for each question in the set. More formally, for an evaluation of a system returning M ranked answers for test set consisting of N questions, the MRR is defined as

$$\text{MRR} = \frac{\sum_{i=1}^{N} \frac{1}{rank_i}}{N} \tag{23.22}$$

23.3 Summarization

The algorithms we have described so far in this chapter present the user an entire document (information retrieval) or a short factoid answer phrase (factoid question answering). But sometimes the user wants something that lies in between these extremes: something like a **summary** of a document or set of documents.

Text summarization

Text summarization is *the process of distilling the most important information from a text to produce an abridged version for a particular task and user* (definition adapted from Mani and Maybury (1999)). Important kinds of summaries that are the focus of current research include the following:

- **outlines** of any document
- **abstracts** of a scientific article
- **headlines** of a news article
- **snippets** summarizing a Web page on a search engine results page
- **action items or other summaries** of a (spoken) business meeting
- **summaries** of email threads

- **compressed sentences** for producing simplified or compressed text
- **answers** to complex questions, constructed by summarizing multiple documents

These kinds of summarization goals are often characterized by their position on two dimensions:

- **single-document** versus **multiple-document** summarization
- **generic** summarization versus **query-focused** summarization

Single-document summarization

In **single-document summarization** we are given a single document and produce a summary. Single-document summarization is thus used in situations like producing a headline or an outline, for which the final goal is to characterize the content of a single document.

Multiple-document summarization

In **multiple-document summarization**, the input is a group of documents, and our goal is to produce a condensation of the content of the entire group. We might use multiple-document summarization when we are summarizing a series of news stories on the same event or when we have Web content on the same topic that we'd like to synthesize and condense.

Generic summary

A **generic summary** is one in which we don't consider a particular user or a particular information need; the summary simply gives the important information in the document(s). By contrast, in **query-focused summarization**, also called **focused summarization**, **topic-based summarization**, and **user-focused summarization**, the summary is produced in response to a user query. We can think of query-focused summarization as a kind of longer, non-factoid answer to a user question.

Query-focused summarization

In the remainder of this section we briefly review the architecture of automatic text summarization systems; the following sections then give details.

Extract

Abstract

One crucial architectural dimension for text summarizers is whether they are producing an **abstract** or an **extract**. The simplest kind of summary, an **extract**, is formed by the combination of phrases or sentences selected (extracted) from the document to be summarized. By contrast, an **abstract** uses different words to describe the contents of the document. We'll illustrate the difference between an extract and an abstract by using the well-known Gettysburg address, a famous speech by Abraham Lincoln, shown in Fig. 23.12.[1] Figure 23.13 shows an extractive summary from the speech followed by an abstract of the speech.

Most current text summarizers are extractive, since extraction is much easier than abstracting; the transition to more sophisticated abstractive summarization is a key goal of recent research.

Text summarization systems and, as it turns out, **natural language generation** systems as well, are generally described by their solutions to the following three problems.

1. **Content Selection:** What information to select from the document(s) we are summarizing. We usually make the simplifying assumption that the granularity of extraction is the sentence or clause. Content selection thus mainly consists of choosing which sentences or clauses to extract into the summary.
2. **Information Ordering:** How to order and structure the extracted units.

[1] In general, one probably wouldn't need a summary of such a short speech, but a short text makes it easier to see how the extract maps to the original for pedagogical purposes. For an amusing alternative application of modern technology to the Gettysburg Address, see Norvig (2005).

Fourscore and seven years ago our fathers brought forth on this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal. Now we are engaged in a great civil war, testing whether that nation, or any nation so conceived and so dedicated, can long endure. We are met on a great battle-field of that war. We have come to dedicate a portion of that field as a final resting-place for those who here gave their lives that this nation might live. It is altogether fitting and proper that we should do this. But, in a larger sense, we cannot dedicate...we cannot consecrate...we cannot hallow... this ground. The brave men, living and dead, who struggled here, have consecrated it far above our poor power to add or detract. The world will little note nor long remember what we say here, but it can never forget what they did here. It is for us, the living, rather, to be dedicated here to the unfinished work which they who fought here have thus far so nobly advanced. It is rather for us to be here dedicated to the great task remaining before us...that from these honored dead we take increased devotion to that cause for which they gave the last full measure of devotion; that we here highly resolve that these dead shall not have died in vain; that this nation, under God, shall have a new birth of freedom; and that government of the people, by the people, for the people, shall not perish from the earth.

Figure 23.12 The Gettysburg Address. Abraham Lincoln, 1863.

Extract from the Gettysburg Address:

Four score and seven years ago our fathers brought forth upon this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal. Now we are engaged in a great civil war, testing whether that nation can long endure. We are met on a great battle-field of that war. We have come to dedicate a portion of that field. But the brave men, living and dead, who struggled here, have consecrated it far above our poor power to add or detract. From these honored dead we take increased devotion to that cause for which they gave the last full measure of devotion — that government of the people, by the people, for the people, shall not perish from the earth.

Abstract of the Gettysburg Address:

This speech by Abraham Lincoln commemorates soldiers who laid down their lives in the Battle of Gettysburg. It reminds the troops that it is the future of freedom in America that they are fighting for.

Figure 23.13 An extract versus an abstract from the Gettysburg Address (abstract from Mani (2001)).

3. **Sentence Realization:** What kind of cleanup to perform on the extracted units so they are fluent in their new context.

In the next sections we show these components in three summarization tasks: **single-document** summarization, **multiple-document** summarization, and **query-focused summarization**.

23.4 Single-Document Summarization

Let's first consider the task of building an extractive summary for a single document. Assuming that the units being extracted are at the level of the sentence, the three summarization stages for this task are as follows.

1. **Content Selection:** Choose sentences to extract from the document.
2. **Information Ordering:** Choose an order in which to place these sentences in the summary.
3. **Sentence Realization:** Clean up the sentences, for example, by removing non-essential phrases from each sentence, by fusing multiple sentences into a single sentence, or by fixing problems in coherence.

Figure 23.14 illustrates the basic architecture underlying this approach.

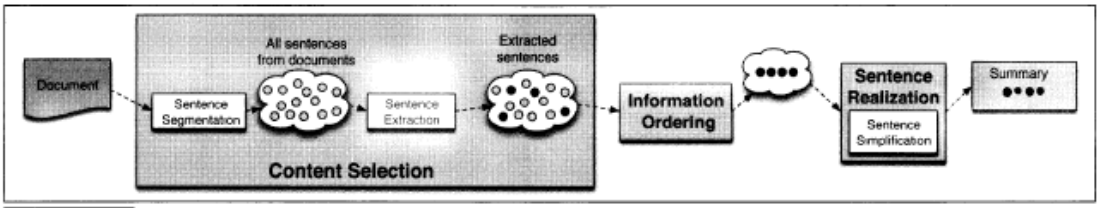

Figure 23.14 The basic architecture of a generic single-document summarizer.

We'll first describe basic summarization techniques with only one of these components: *content selection*. Indeed, many single-document summarizers have no information-ordering component, simply ordering the extracted sentences in the order they appeared in the original document. In addition, we'll assume for now that sentences are not combined or cleaned up after they are extracted, although we'll briefly mention later how this is done.

23.4.1 Unsupervised Content Selection

Content selection

The **content selection** task of extracting sentences is often treated as a classification task. The goal of the classifier is to label each sentence in a document with a binary label: *important* versus *unimportant* (or *extract worthy* versus *not extract worthy*). We begin with some unsupervised algorithms for sentence classification and then turn to supervised algorithms in the next section.

The simplest unsupervised algorithm, based on an intuition that dates back to the early summarizer of (Luhn, 1958), selects sentences that have more **salient** or **informative** words. Sentences that contain more informative words tend to be more extract worthy. Saliency is usually defined by computing the **topic signature**, a set of **salient** or **signature terms**, each of whose saliency scores is greater than some threshold θ.

Topic signature
Signature terms

Saliency could be measured in terms of simple word frequency, but frequency has the problem that a word might have a high probability in English in general but not be particularly topical to a particular document. Therefore, weighting schemes like **tf-idf** or **log-likelihood ratio** are more often used.

Recall from page 771 that the tf-idf scheme gives a high weight to words that appear frequently in the current document but rarely in the overall document collection, suggesting that the word is particularly relevant to this document. For each term i that occurs in the sentence to be evaluated, we compute its count in the current document j $\text{tf}_{i,j}$ and multiply by the inverse document frequency over the whole collection idf_i:

$$weight(w_i) = \text{tf}_{i,j} \times \text{idf}_i \tag{23.23}$$

Log-likelihood ratio

A better-performing method for finding informative words is the **log-likelihood ratio** (LLR). The LLR for a word, generally called $\lambda(w)$, is the ratio between the probability of observing w both in the input and in the background corpus assuming equal probabilities in both corpora, and the probability of observing w in both assuming different probabilities for w in the input and the background corpus. See Dunning (1993), Moore (2004), and Manning and Schütze (1999) for details on log-likelihood and how it is calculated.

It turns out for the log-likelihood ratio that the quantity $-2\log(\lambda)$ is asymptotically well approximated by the χ^2 distribution, which means that a word appears in the input significantly more often than in the background corpus (at $\alpha = 0.001$) if $-2log(\lambda) > 10.8$. Lin and Hovy (2000) suggested that this made the log-likelihood ratio particularly appropriate for selecting a topic signature for summarization. Thus, the word weight with the log-likelihood ratio is generally defined as follows:

$$weight(w_i) = \begin{cases} 1 & \text{if } -2\log(\lambda(w_i)) > 10 \\ 0 & \text{otherwise.} \end{cases} \tag{23.24}$$

Equation 23.24 is used to set a weight of 1 or 0 for each word in the sentence. The score for a sentence s_i is then the average weight of its non-stop words:

$$weight(s_i) = \sum_{w \in s_i} \frac{weight(w)}{|\{w|w \in s_i\}|} \tag{23.25}$$

The summarization algorithm computes this weight for every sentence, and then ranks all sentences by their score. The extracted summary consists of the top ranked sentences.

The family of algorithms that this thresholded LLR algorithm belongs to is called **centroid-based summarization** because we can view the set of signature terms as a pseudo-sentence that is the "centroid" of all the sentences in the document and we are looking for sentences that are as close as possible to this centroid sentence.

Centrality

A common alternative to the log-likelihood ratio/centroid method is to use a different model of sentence **centrality**. These other centrality-based methods resemble the centroid method described above, in that their goal is to rank the input sentences in terms of how central they are in representing the information present in the document. But rather than just ranking sentences by whether they contain salient words, centrality-based methods compute distances between each candidate sentence and each other sentence and choose sentences that are on average closer to other sentences. To compute centrality, we can represent each sentence as a bag-of-words vector of length N, as described in Chapter 20. For each pair of sentences x and y, we compute the tf-idf weighted cosine as described in Eq. 23.12 on page 771.

Each of the k sentences in the input is then assigned a centrality score that is its average cosine with all other sentences:

$$centrality(x) = \frac{1}{K}\sum_{y} tf\text{-}idf\text{-}cosine(x,y) \tag{23.26}$$

Sentences are ranked by this centrality score, and the sentence that has the highest average cosine across all pairs, that is, is most like other sentences, is chosen as the most "representative" or "topical" of all the sentences in the input.

It is also possible to extend this centrality score to use more complex graph-based measures of centrality like PageRank (Erkan and Radev, 2004).

23.4.2 Unsupervised Summarization Based on Rhetorical Parsing

The sentence-extraction algorithm we introduced above for content extraction relied solely on a single shallow feature, word saliency, ignoring possible higher-level cues such as discourse information. In this section we briefly summarize a way to get more sophisticated discourse knowledge into the summarization task.

The summarization algorithm we describe makes use of **coherence relations** such as the RST (Rhetorical Structure Theory) relations described in Chapter 21. Recall that RST relations are often expressed in terms of a **satellite** and a **nucleus**; nucleus sentences are more likely to be appropriate for a summary. For example, consider the following two paragraphs taken from the *Scientific American* magazine text that we introduced in Section 21.2.1:

> With its distant orbit – 50 percent farther from the sun than Earth – and slim atmospheric blanket, Mars experiences frigid weather conditions. Surface temperatures typically average about −70 degrees Fahrenheit at the equator, and can dip to −123 degrees C near the poles.
>
> Only the midday sun at tropical latitudes is warm enough to thaw ice on occasion, but any liquid water formed in this way would evaporate almost instantly because of the low atmospheric pressure. Although the atmosphere holds a small amount of water, and water-ice clouds sometimes develop, most Martian weather involves blowing dust or carbon dioxide.

The first two discourse units in this passage are related by the RST JUSTIFICATION relation, with the first discourse unit justifying the second unit, as shown in Fig. 23.15. The second unit (*"Mars experiences frigid weather conditions"*) is thus the nucleus and captures better what this part of the document is about.

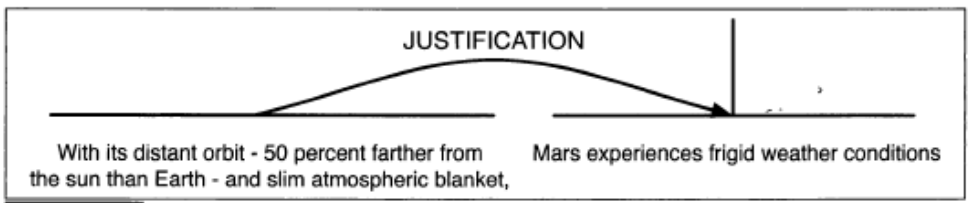

Figure 23.15 The justification relation between two discourse units, a satellite (on the left) and a nucleus (on the right).

We can use this intuition for summarization by first applying a discourse parser of the type discussed in Chapter 21 to compute the coherence relations between each discourse unit. Once a sentence has been parsed into a coherence relation graph or parse tree, we can use the intuition that the nuclear units are important for summarization by recursively extracting the salient units of a text.

Consider the coherence parse tree in Fig. 23.16. The salience of each node in the tree can be defined recursively as follows:

- Base case: The salient unit of a leaf node is the leaf node itself.
- Recursive case: The salient units of an intermediate node are the union of the salient units of its immediate *nuclear* children.

By this definition, discourse unit (2) is the most salient unit of the entire text (since the root node spanning units 1–8 has the node spanning units 1–6 as its nucleus, and unit 2 is the nucleus of the node spanning units 1–6).

If we rank each discourse unit by the height of the nodes of which it is the nucleus, we can assign a partial ordering of salience to units; the algorithm of Marcu (1995) assigns the following partial ordering to this discourse:

$$2 > 8 > 3 > 1,4,5,7 > 6 \tag{23.27}$$

See Marcu (1995, 2000b) for the details of exactly how this partial order is computed, and Teufel and Moens (2002) for another method for using rhetorical structure in summarization.

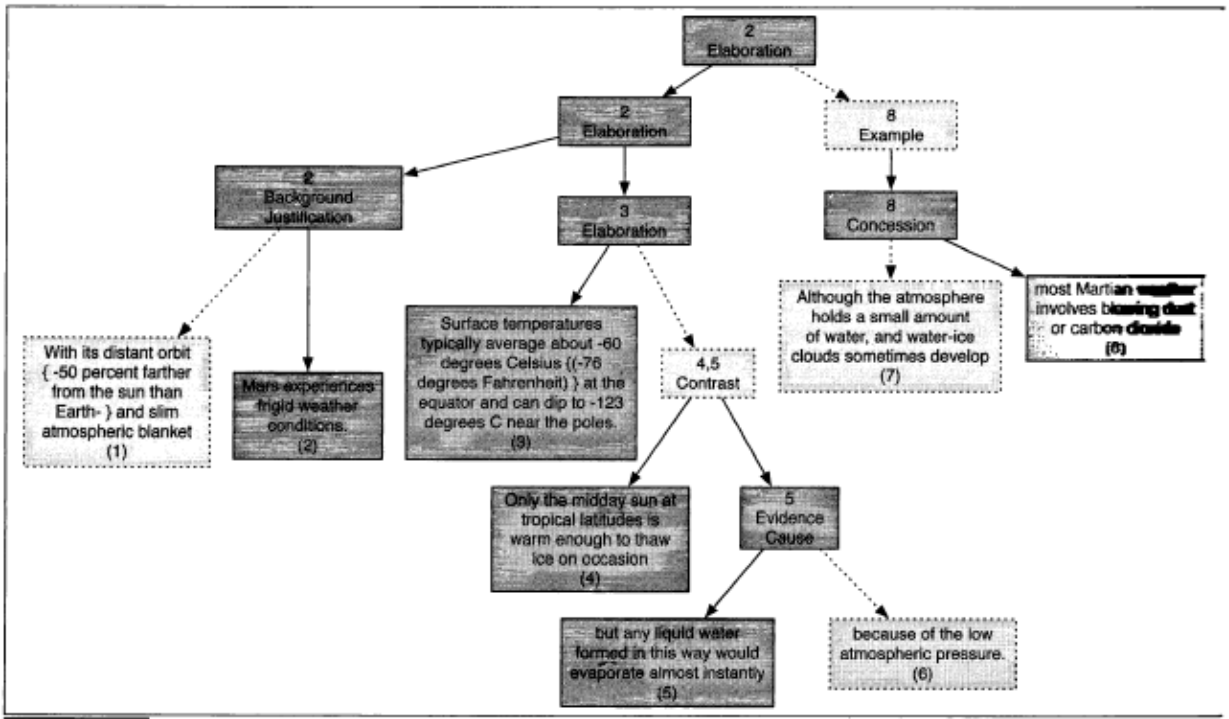

Figure 23.16 The discourse tree for the text on page 792. Boldface links connect nodes to their nuclei children; dotted lines to the satellite children. After Marcu (2000a).

23.4.3 Supervised Content Selection

While the use of topic signatures for unsupervised content selection is an extremely effective method, a topic signature is only a single cue for finding extract-worthy sentences. Many other cues exist, including the alternative saliency methods discussed above like centrality and PageRank methods, as well as other cues like the position of the sentence in the document (sentences at the very beginning or end of the document tend to be more important), the length of each sentence, and so on. We'd like a method that can weigh and combine all of these cues.

The most principled method for weighing and combining evidence is supervised machine learning. For supervised machine learning, we need a training set of documents paired with human-created summary extracts, such as the Ziff-Davis corpus (Marcu, 1999). Since these are *extracts*, each sentence in the summary is, by definition, taken from the document. That means we can assign a label to every sentence in the document; **1** if it appears in the extract, **0** if it doesn't. To build our classifier, then, we just need to choose features to extract that are predictive of being a good sentence to appear in a summary. Some of the features commonly used in sentence classification are shown in Fig. 23.17.

position	The position of the sentence in the document. For example, Hovy and Lin (1999) found that the single most extract-worthy sentence in most newspaper articles is the title sentence. In the Ziff-Davis corpus they examined, the next most informative was the first sentence of paragraph 2 (P1S1), followed by the first sentence of paragraph 3 (P3S1); thus the list of ordinal sentence positions starting from the most informative was: T1, P2S1, P3S1, P4S1, P1S1, P2S2,... Position, like almost all summarization features, is heavily genre dependent. In *Wall Street Journal articles*, they found the most important information appeared in the following sentences: T1, P1S1, P1S2,...
cue phrases	Sentences containing phrases like *in summary*, *in conclusion*, or *this paper* are more likely to be extract worthy. These cue phrases are very dependent on the genre. For example, in British House of Lords legal summaries, the phrase *it seems to me that* is a useful cue phrase (Hachey and Grover, 2005).
word informativeness	Sentences that contain more terms from the **topic signature**, as described in the previous section, are more extract worthy.
sentence length	Very short sentences are rarely appropriate for extracting. We usually capture this fact by using a binary feature based on a cutoff (true if the sentence has more than, say, five words).
cohesion	Recall from Chapter 21 that a **lexical chain** is a series of related words that occurs throughout a discourse. Sentences that contain more terms from a lexical chain are often extract worthy because they are indicative of a continuing topic (Barzilay and Elhadad, 1997). This kind of cohesion can also be computed by graph-based methods (Mani and Bloedorn, 1999). The PageRank graph-based measures of sentence centrality discussed above can also be viewed as a coherence metric (Erkan and Radev, 2004).

Figure 23.17 Some features commonly used in supervised classifiers for determining whether a document sentence should be extracted into a summary.

Each sentence in our training document thus has a label (0 if the sentence is not in the training summary for that document, 1 if it is) and a set of extracted feature values like those in Fig. 23.17. We can then train our classifier to estimate these labels for unseen data; for example, a probabilistic classifier like naive Bayes or MaxEnt would be computing the probability that a particular sentence s is extract worthy given a set of features $f_1 ... f_n$; we can then just extract any sentences for which this probability is greater than 0.5:

$$P(\text{extract-worthy}(s)|f_1, f_2, f_3, ..., f_n) \tag{23.28}$$

There is one problem with the algorithm as we've described it: it requires that we have for each document a training summary that consists solely of extracted sentences. If we could weaken this restriction, we could apply the algorithm to a much wider variety of summary-document pairs, such as conference papers or journal articles and their abstracts. Luckily, it turns out that when humans write summaries, even with the goal of writing abstractive summaries, they very often use phrases and sentences from the document to compose the summary. But they don't use *only* extracted sentences; they often combine two sentences into one, change some of the words in the sentences, or write completely new abstractive sentences. Here is an example of an extracted sentence from a human summary that, although modified in the final human summary, was clearly a document sentence that should be labeled as extract worthy:

(23.29) **Human summary**: This paper identifies the desirable features of an ideal multisensor gas monitor and lists the different models currently available.

(23.30) **Original document sentence**: The present part lists the desirable features and the different models of portable, multisensor gas monitors currently available.

Thus, an important preliminary stage is to *align* each training document with its summary, with the goal of finding which sentences in the document were (completely or mostly) included in the summary. A simple algorithm for **alignment** is to find the source document and abstract sentences with the longest common subsequences of non-stopwords; alternatively, minimum edit distance can be computed or more sophisticated knowledge sources, such as WordNet, can be used. Recent work has focused on more complex alignment algorithms such as the use of HMMs (Jing, 2002; Daumé III and Marcu, 2005, inter alia).

Alignment

Given such alignment algorithms, supervised methods for content selection can make use of parallel corpora of documents and human abstractive summaries, such as academic papers with their abstracts (Teufel and Moens, 2002).

23.4.4 Sentence Simplification

Once a set of sentences has been extracted and ordered, the final step is **sentence realization**. One component of sentence realization is **sentence compression** or **sentence simplification**. The following examples, taken by Jing (2000) from a human summary,

Sentence compression
Sentence simplification

show that the human summarizer chose to eliminate some of the adjective modifiers and subordinate clauses when expressing the extracted sentence in the summary.

(23.31) **Original sentence:** ~~When it arrives sometime new year in new TV sets,~~ the V-chip will give parents a ~~new and potentially revolutionary~~ device to block out programs they don't want their children to see.

(23.32) **Simplified sentence by humans:** The V-chip will give parents a device to block out programs they don't want their children to see.

The simplest algorithms for sentence simplification use rules to select parts of the sentence to prune or keep, often by running a parser or partial parser over the sentences. Some representative rules from Zajic et al. (2007), Conroy et al. (2006), and Vanderwende et al. (2007) remove the following:

appositives	Rajam, ~~28, an artist who was living at the time in Philadelphia,~~ found the inspiration in the back of city magazines.
attribution clauses	Rebels agreed to talks with government officials, ~~international observers said Tuesday.~~
PPs without named entities	The commercial fishing restrictions in Washington will not be lifted [SBAR unless the salmon population 329 increases [PP ~~to a sustainable number~~]
initial adverbials	"For example", "On the other hand", "As a matter of fact", "At this point"

More sophisticated models of sentence compression are based on supervised machine learning, in which a parallel corpus of documents together with their human summaries is used to compute the probability that particular words or parse nodes will be pruned. See the Historical Notes section at the end of the chapter for pointers to this extensive recent literature.

23.5 Multi-Document Summarization

Multi-document summarization

When we apply summarization techniques to groups of documents rather than to a single document, we call the goal **multi-document summarization**. Multi-document summarization is particularly appropriate for Web-based applications, for example, for building summaries of a particular event in the news by combining information from different news stories, or finding answers to complex questions by including components from extracted from multiple documents.

While multi-document summarization is far from a solved problem, even the current technology can be useful for information-finding tasks. McKeown et al. (2005), for example, gave documents to human participants along with a human summary and an automatically generated summary or no summary, and had the participants perform time-restricted, fact-gathering tasks. The participants had to answer three related questions about an event in the news; subjects who read the automatic summaries gave higher-quality answers to the questions.

Multi-document summarization algorithms are based on the same three steps we've seen before. In most cases, we start with a cluster of documents that we'd like to summarize; we must then perform **content selection**, **information ordering**, and **sentence realization**, as described in the next three sections and sketched in Fig. 23.18.

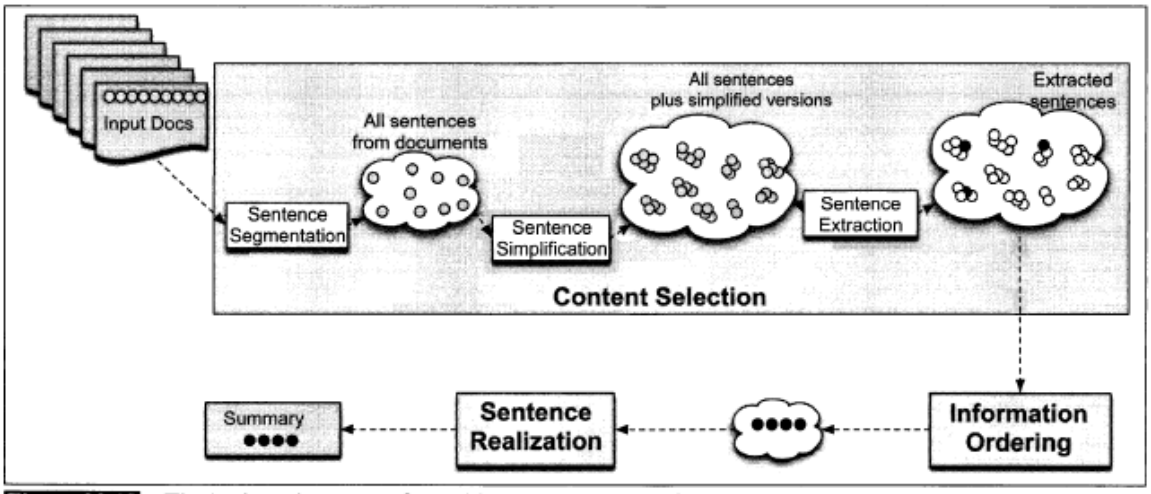

Figure 23.18 The basic architecture of a multi-document summarizer.

23.5.1 Content Selection in Multi-Document Summarization

In single-document summarization we used both supervised and unsupervised methods for content selection. For multiple-document summarization, supervised training sets are less available, and we focus more on unsupervised methods.

The major difference between the tasks of single-document and multiple-document summarization is the greater amount of **redundancy** when we start with multiple documents. A group of documents can have significant overlap in words, phrases, and concepts, in addition to information that might be unique to each article. While we want each sentence in the summary to be about the topic, we don't want the summary to consist of a set of identical sentences.

For this reason, algorithms for multi-document summarization focus on ways to avoid redundancy when selecting sentences for the summary. When adding a new sentence to a list of extracted sentences, we need some way to make sure the sentence doesn't overlap too much with the already extracted sentences.

A simple method of avoiding redundancy is to explicitly include a redundancy factor in the scoring for choosing a sentence to extract. The redundancy factor is based on the similarity between a candidate sentence and the sentences that have already been extracted into the summary; a sentence is penalized if it is too similar to the summary. For example, the **MMR** or **Maximal Marginal Relevance** scoring system (Carbonell and Goldstein, 1998; Goldstein et al., 2000) includes the following penalty term for representing the similarity between a sentence s and the set of sentences already extracted for the summary *Summary*, where λ is a weight that can be tuned and Sim is some similarity function:

MMR
Maximal Marginal Relevance

$$\text{MMR penalization factor}(s) = \lambda max_{s_i \in Summary} \text{Sim}(s, s_i) \tag{23.33}$$

An alternative to this MMR-based method is to instead apply a clustering algorithm to all the sentences in the documents to be summarized to produce a number of clusters of related sentences and then to select a single (centroid) sentence from each cluster for entry into the summary.

By adding MMR or clustering methods for avoiding redundancy, we can also do sentence simplification or compression at the content-selection stage rather than at the sentence-realization stage. A common way to fit simplification into the architecture is to run various sentence simplification rules (Section 23.4.4) on each sentence in the input corpus. The result will be multiple versions of the input sentence, each version with different amounts of simplification. For example, the following sentence

> Former Democratic National Committee finance director Richard Sullivan faced more pointed questioning from Republicans during his second day on the witness stand in the Senate's fund-raising investigation.

might produce different shortened versions:

- Richard Sullivan faced pointed questioning.
- Richard Sullivan faced pointed questioning from Republicans.
- Richard Sullivan faced pointed questioning from Republicans during day on stand in Senate fund-raising investigation.
- Richard Sullivan faced pointed questioning from Republicans in Senate fund-raising investigation.

This expanded corpus is now used as the input to content extraction. Redundancy methods such as clustering or MMR will choose only the (optimally long) single version of each original sentence.

23.5.2 Information Ordering in Multi-Document Summarization

The second stage of an extractive summarizer is the ordering or structuring of information, when we must decide how to concatenate the extracted sentences into a coherent order. Recall that in single-document summarization, we can just use the original article ordering for these sentences. This isn't appropriate for most multiple-document applications, although we can certainly apply it if many or all of the extracted sentences happen to come from a single article.

Chronological ordering

For sentences extracted from news stories, one technique is to use the dates associated with the story, a strategy known as **chronological ordering**. It turns out that pure chronological ordering can produce summaries that lack cohesion; this problem can be addressed by the ordering of slightly larger chunks of sentences rather than single sentences; see Barzilay et al. (2002).

Perhaps the most important factor for information ordering, however, is **coherence**. Recall from Chapter 21 the various devices that contribute to the coherence of a discourse. One is having sensible coherence relations between the sentences; thus, we could prefer orderings in summaries that result in sensible coherence relations between the sentences. Another aspect of coherence has to do with cohesion and lexical chains; we could, for example, prefer orderings that have more local cohesion. A final aspect

of coherence is coreference; a coherence discourse is one in which entities are mentioned in coherent patterns. We could prefer orderings with coherent entity-mention patterns.

All of these kinds of coherence have been used for information ordering. For example, we can use *lexical cohesion* as an ordering heuristic by ordering each sentence next to sentences containing similar words. We can do this by defining the standard tf-idf cosine distance between each pair of sentences and choosing the overall ordering that minimizes the average distance between neighboring sentences (Conroy et al., 2006), or we can build models of predictable word sequences across sentences (Soricut and Marcu, 2006).

Coreference-based coherence algorithms have also made use of the intuitions of **Centering**. Recall that the Centering algorithm was based on the idea that each discourse segment has a salient entity, the *focus*. Centering theory proposed that certain syntactic realizations of the focus (i.e., as subject or object) and certain transitions between these realizations (e.g., if the same entity is the subject of adjacent sentences) created a more coherent discourse. Thus, we can prefer orderings in which the transition between entity mentions is a preferred one.

For example, in the entity-based information approach of Barzilay and Lapata (2005, 2008), a training set of summaries is parsed and labeled for coreference. The resulting sequence of entity realizations can be automatically extracted and represented into an **entity grid**. Figure 23.19 shows a simplified version of a parsed summary and the extracted grid. A probabilistic model of particular entity transitions (i.e., $\{S,O,X,-\}$) can then be trained from the entity grid. For example, the transitions $\{X,O,S,S\}$ for the headword *Microsoft* exemplify the fact that new entities in a discourse are often introduced first in oblique or object position and then only later appear in subject position. See Barzilay and Lapata (2008) for details.

Entity grid

A general way to view all of these methods is as assigning a coherence score to a sequence of sentences through a local coherence score between pairs or sequences of sentences; a single general transition score between sentences could then combine lexical coherence and entity-based coherence. Once we have such a scoring function, choosing an ordering that optimizes all these local pairwise distances is known to be quite difficult. The task of finding the optimal ordering of a set of sentences given a set of pairwise distances between the sentences is equivalent to very hard problems like Cyclic Ordering and the Traveling Salesman Problem.[2] Sentence ordering is thus equivalent to the difficult class of problems known as **NP-complete**. While these problems are difficult to solve exactly, a number of good approximation methods for solving NP-complete problems have been applied to the information ordering task. See Althaus et al. (2004), Knight (1999a), Cohen et al. (1999), and Brew (1992) for the relevant proofs and approximation techniques.

In the models described above, the information-ordering task is completely separate from content extraction. An alternative approach is to learn the two tasks jointly, resulting in a model that both selects sentences and orders them. For example, in the HMM model of Barzilay and Lee (2004), the hidden states correspond to document

[2] The Traveling Salesman Problem: given a set of cities and the pairwise distances between them, find the shortest path that visits each city exactly once and then returns to the start city.

		Department	Trial	Microsoft	Markets	Products	Brands	Case	Netscape	Software	Tactics
1	[The Justice Department]$_S$ is conducting an [anti-trust trial]$_O$ against [Microsoft Corp.]$_X$	1	S	O	X	-	-	-	-	-	-
2	[Microsoft]$_O$ is accused of trying to forcefully buy into [markets]$_X$ where [its own products]$_S$ are not competitive enough to unseat [established brands]$_O$	2	-	-	O	X	S	O	-	-	-
3	[The case]$_S$ resolves around [evidence]$_O$ of [Microsoft]$_S$ aggressively pressuring [Netscape]$_O$ into merging [browser software]$_O$	3	-	-	S	O	-	-	S	O	O
4	[Microsoft]$_S$ claims [its tactics]$_S$ are commonplace and good economically.	4	-	-	S	-	-	-	-	-	O

Figure 23.19 A summary (showing entities in subject (S), object (O), or oblique (X) position), and the entity grid that is extracted from it. Adapted from Barzilay and Lapata (2005).

content topics and the observations to sentences. For example for newspaper articles on earthquakes, the hidden states (topics) might be *strength of earthquake*, *location*, *rescue efforts*, and *casualties*. Barzilay and Lee apply clustering and HMM induction to induce these hidden states and the transitions between them. For example, here are three sentences from the *location* cluster they induce:

(23.34) The Athens seismological institute said the temblor's epicenter was located 380 kilometers (238 miles) south of the capital.

(23.35) Seismologists in Pakistan's Northwest Frontier Province said the temblor's epicenter was about 250 kilometers (155 miles) north of the provincial capital Peshawar.

(23.36) The temblor was centered 60 kilometers (35 miles) northwest of the provincial capital of Kunming, about 2,200 kilometers (1,300 miles) southwest of Beijing, a bureau seismologist said.

The learned structure of the HMM then implicitly represents information-ordering facts like *mention "casualties" prior to "rescue" efforts* through the HMM transition probabilities.

In summary, we've seen information ordering based on **chronological order**, based on **coherence**, and an ordering that is learned automatically from the data. In the next section on query-focused summarization, we introduce a final method in which information ordering can be specified according to an ordering template that is predefined for different query types.

Sentence Realization

While discourse coherence can be factored in during sentence ordering, the resulting sentences may still have coherence problems. For example, as we saw in Chapter 21, when a referent appears multiple times in a coreference chain in a discourse, the longer or more descriptive noun phrases occur before shorter, reduced, or pronominal forms. But the ordering we choose for the extracted sentences may not respect this coherence preference.

For example, the boldfaced names in the original summary in Fig. 23.20 appear in an incoherent order; the full name **U.S. President George W. Bush** occurs only after the shortened form **Bush** has been introduced.

One possible way to address this problem in the sentence-realization stage is to apply a coreference resolution algorithm to the output, extracting names and applying some simple cleanup rewrite rules like the following:

(23.37) Use the **full name** at the first mention, and just the **last name** at subsequent mentions.

(23.38) Use a **modified** form for the first mention, but remove appositives or premodifiers from any subsequent mentions.

The rewritten summary in Fig. 23.20 shows how such rules would apply; in general, such methods would depend on high-accuracy coreference resolution.

Original summary:

Presidential advisers do not blame **O'Neill**, but they've long recognized that a shakeup of the economic team would help indicate **Bush** was doing everything he could to improve matters. **U.S. President George W. Bush** pushed out **Treasury Secretary Paul O'Neill** and top economic adviser Lawrence Lindsey on Friday, launching the first shakeup of his administration to tackle the ailing economy before the 2004 election campaign.

Rewritten summary:

Presidential advisers do not blame **Treasury Secretary Paul O'Neill**, but they've long recognized that a shakeup of the economic team would help indicate **U.S. President George W. Bush** was doing everything he could to improve matters. **Bush** pushed out **O'Neill** and White House economic adviser Lawrence Lindsey on Friday, launching the first shakeup of his administration to tackle the ailing economy before the 2004 election campaign.

Figure 23.20 Rewriting references, from Nenkova and McKeown (2003).

Sentence fusion

Recent research has also focused on a finer granularity for realization than the extracted sentence by using **sentence fusion** algorithms to combine phrases from different sentences. The sentence fusion algorithm of Barzilay and McKeown (2005) parses each sentence, uses multiple-sequence alignment of the parses to find areas of common information, builds a fusion lattice with overlapping information, and creates a new fused sentence by linearizing a string of words from the lattice.

23.6 Focused Summarization and Question Answering

As noted at the beginning of this chapter, most interesting questions are not factoid questions. User needs require longer, more informative answers than a single phrase can provide. For example, while a DEFINITION question might be answered by a short phrase like "*Autism is* **a developmental disorder**" or "*A caldera is* **a volcanic crater**", a user might want more information, as in the following definition of *water spinach*:

> *Water spinach* (ipomoea aquatica) is a semi-aquatic leafy green plant characterized by long hollow stems and spear-shaped or heart-shaped leaves which is widely grown throughout Asia as a leaf vegetable. The leaves and stems are often eaten stir-fried as greens with salt or salty sauces, or in soups. Other common names include *morning glory vegetable*, *kangkong*

> (Malay), *rau muong* (Vietnamese), *ong choi* (Cantonese), and *kong xin cai* (Mandarin). It is not related to spinach, but is closely related to sweet potato and convolvulus.

Complex questions can also be asked in domains like medicine, such as this question about a particular drug intervention:

(23.39) In children with an acute febrile illness, what is the efficacy of single-medication therapy with acetaminophen or ibuprofen in reducing fever?

For this medical question, we'd like to be able to extract an answer of the following type, perhaps giving the document ID(s) that the extract came from and some estimate of our confidence in the result:

> Ibuprofen provided greater temperature decrement and longer duration of antipyresis than acetaminophen when the two drugs were administered in approximately equal doses. (PubMedID: 1621668, Evidence Strength: A)

Questions can be even more complex, such as this one from the Document Understanding Conference annual summarization competition:

(23.40) Where have poachers endangered wildlife, what wildlife has been endangered and what steps have been taken to prevent poaching?

Where a factoid answer might be found in a single phrase in a single document or Web page, these kinds of complex questions are likely to require much longer answers that are synthesized from many documents or pages.

For this reason, summarization techniques are often used to build answers to these kinds of complex questions. But unlike the summarization algorithms introduced above, the summaries produced for complex question answering must be relevant to some user question. When a document is summarized for the purpose of answering some user query or information need, we call the goal **query-focused summarization** or sometimes just **focused summarization**. (The terms **topic-based summarization** and **user-focused summarization** are also used.) A query-focused summary is thus really a kind of longer, non-factoid answer to a user question or information need.

Query-focused summarization

Snippet

One kind of query-focused summary is a **snippet**, the kind that Web search engines like Google return to the user to describe each retrieved document. Snippets are query-focused summaries of a single document. But since for complex queries we will want to aggregate information from multiple documents, we'll need to summarize multiple documents.

Indeed, the simplest way to do query-focused summarization is to slightly modify the algorithms for multiple document summarization that we introduced in the previous section to make use of the query. For example, when ranking sentences from all the returned documents in the content-selection phase, we can require that any extracted sentence must contain at least one word overlapping with the query. Or we can just add the cosine distance from the query as one of the relevance features in sentence extraction. We can characterize such a method of query-focused summarization as a bottom-up, domain-independent method.

An alternative way to do query-focused summarization is to make additional use of top-down or information-extraction techniques, building specific content-selection

algorithms for different types of complex questions. Thus, we could specifically build a query-focused summarizer for the kinds of advanced questions introduced above, like definition questions, biography questions, and certain medical questions. In each case, we use our top-down expectations for what makes a good definition, biography, or medical answer to guide what kinds of sentences we extract.

Genus
Species

For example, a **definition** of a term often includes information about the term's **genus** and **species**. The genus is the hypernym or superordinate of the word; thus, a sentence like *The Hajj is a type of ritual* is a genus sentence. The species gives important additional properties of the term that differentiate the term from other hyponyms of the genus; an example is *"The annual hajj begins in the twelfth month of the Islamic year"*. Other kinds of information that can occur in a definition include **synonyms**, **etymology**, **subtypes**, and so on.

To build extractive answers for definition questions, we'll need to make sure we extract sentences with the genus information, the species information, and other generally informative sentences. Similarly, a good **biography** of a person contains information such as the person's **birth/death**, **fame factor**, **education**, **nationality** and so on; we'll need to extract sentences with each of these kinds of information. A medical answer that summarizes the results of a study on applying a drug to a medical problem would need to contain information like the **problem** (the medical condition), the **intervention** (the drug or procedure), and the **outcome** (the result of the study). Figure 23.21 shows some example predicates for definition, biography, and medical intervention questions.

Definition	
genus	`The Hajj is a type of ritual`
species	`the annual hajj begins in the twelfth month of the Islamic year`
synonym	`The Hajj, or Pilgrimage to Mecca, is the central duty of Islam`
subtype	`Qiran, Tamattu', and Ifrad are three different types of Hajj`
Biography	
dates	`was assassinated on April 4, 1968`
nationality	`was born in Atlanta, Georgia`
education	`entered Boston University as a doctoral student`
Drug efficacy	
population	`37 otherwise healthy children aged 2 to 12 years`
problem	`acute, intercurrent, febrile illness`
intervention	`acetaminophen (10 mg/kg)`
outcome	`ibuprofen provided greater temperature decrement and longer duration of antipyresis than acetaminophen when the two drugs were administered in approximately equal doses`

Figure 23.21 Examples of some different types of information that must be extracted in order to produce answers to certain kinds of complex questions.

In each case, we use the **information-extraction** methods of Chapter 22 to find specific sentences for genus and species (for definitions), or dates, nationality, and education (for biographies), or problems, interventions, and outcomes (for medical questions). We can then use standard domain-independent, content-selection algorithms to find other good sentences to add on to these.

A typical architecture consists of the four steps shown in Fig. 23.22 from the definition-extraction system of Blair-Goldensohn et al. (2004). The input is a definition question T, the number N of documents to retrieve, and the length L of the answer (in sentences).

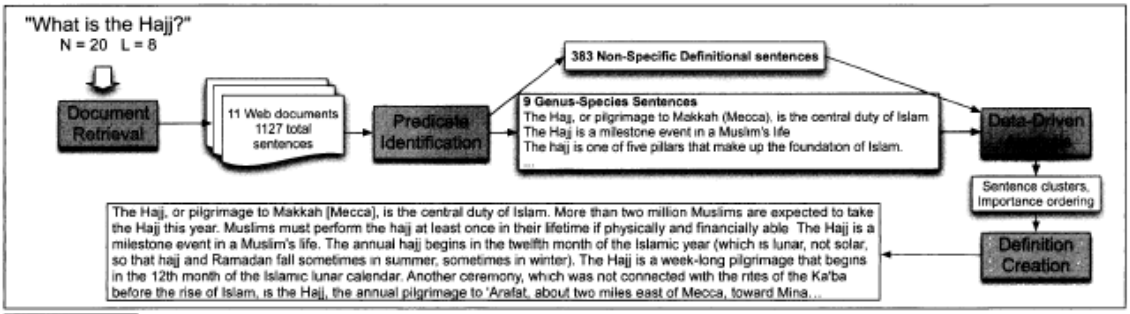

Figure 23.22 Architecture of a query-focused summarizer for definition questions (Blair-Goldensohn et al., 2004).

The first step in any IE-based, complex question-answering system is information retrieval. In this case, a handwritten set of patterns is used to extract the term to be defined from the query T (*Hajj*) and to generate a series of queries that are sent to an IR engine. Similarly, in a biography system it would be the name that would be extracted and passed to the IR engine. The returned documents are broken up into sentences.

In the second stage, we apply classifiers to label each sentence with an appropriate set of classes for the domain. For definition questions, Blair-Goldensohn et al. (2004) used a set of four classes: **genus**, **species**, **other definitional**, or **other**. The third class, **other definitional**, is used to select other sentences that might be added into the summary. These classifiers can be based on any of the information extraction techniques introduced in Chapter 22, including hand-written rules or supervised machine learning.

In the third stage, we can use the methods described in the section on generic (non-query-focused), multiple-document summarization content selection to add additional sentences to our answer that might not fall into a specific information-extraction type. For example, for definition questions, all the sentences that are classified as *other definitional* are examined and a set of relevant sentences is selected from them. This selection can be done by the centroid method, in which we form a tf-idf vector for each sentence, find the centroid of all the vectors, and then choose the K sentences closest to the centroid. Alternatively, we can use a method for avoiding redundancy, like clustering the vectors and choosing the best sentence from each cluster.

Because query-focused summarizers of this type are domain specific, we can use domain-specific methods such as fixed hand-built templates for information ordering as well. For biography questions we might use a template like the following:

(23.41) <NAME> is <WHY FAMOUS>. She was born on <BIRTHDATE> in <BIRTHLOCATION>. She <EDUCATION>. <DESCRIPTIVE SENTENCE>. <DESCRIPTIVE SENTENCE>.

The various sentences or phrases selected in the content selection phase can then be fit into this template. These templates can also be somewhat more abstract. For

example, for definitions, we could place a genus-species sentence first, followed by remaining sentences ordered by their saliency scores.

23.7 Summarization Evaluation

As is true for other speech and language processing areas like machine translation, there are a wide variety of evaluation metrics for summarization, metrics requiring human annotation as well as completely automatic metrics.[3]

As we have seen for other tasks, we can evaluate a system by **extrinsic** (task-based) or **intrinsic** (task-independent) methods. We described a kind of extrinsic evaluation of multi-document summarization in Section 23.5, in which subjects were asked to perform time-restricted, fact-gathering tasks and were given full documents together with either no summaries, human summaries, or automatically generated summaries to read. The subjects had to answer three related questions about an event in the news. For query-focused, single-document summarization (like the task of generating Web **snippets**), we can measure how different summarization algorithms affect human performance at the task of deciding if a document is relevant/not-relevant to a query by looking solely at the summary.

ROUGE

The most commonly used intrinsic summarization evaluation metric is an automatic method called **ROUGE, Recall-Oriented Understudy for Gisting Evaluation** (Lin and Hovy, 2003; Lin, 2004). ROUGE is inspired by the BLEU metric used for evaluating machine translation output and, like BLEU, automatically scores a machine-generated candidate summary by measuring the amount of N-gram overlap between the candidate and human-generated summaries (the references).

ROUGE-1
ROUGE-2

BLEU is computed by averaging the number of overlapping N-grams of different length between the hypothesis and reference translations. In ROUGE, by contrast, the length of the N-gram is fixed; **ROUGE-1** uses unigram overlap, and **ROUGE-2** uses bigram overlap. We'll define ROUGE-2; the definitions of all the other ROUGE-N metrics follow naturally. ROUGE-2 is a measure of the bigram recall between the candidate summary and the set of human reference summaries:

$$ROUGE2 = \frac{\sum_{S \in \{ReferenceSummaries\}} \sum_{bigram \in S} \text{Count}_{\text{match}}(bigram)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{bigram \in S} \text{Count}(bigram)} \tag{23.42}$$

The function $\text{Count}_{\text{match}}(bigram)$ returns the maximum number of bigrams that co-occur in the candidate summary and the set of reference summaries. ROUGE-1 is the same but counts unigrams instead of bigrams.

[3] We focus here on evaluation of entire summarization algorithms and ignore evaluation of subcomponents such as information ordering, although see, for example, Lapata (2006) on the use of Kendall's τ, a metric of rank correlation, for information ordering.

Note that ROUGE is a recall-oriented measure, whereas BLEU is a precision-oriented measure. This is because the denominator of (23.42) is the total sum of the number of bigrams in the reference summaries. By contrast, in BLEU the denominator is the total sum of the number of N-grams in the candidates. Thus, ROUGE is measuring something like how many of the human reference summary bigrams are covered by the candidate summary, whereas BLEU is measuring something like how many of the candidate translation bigrams occurred in the human reference translations.

ROUGE-L
ROUGE-S
ROUGE-SU
Skip bigram

Variants of ROUGE include **ROUGE-L**, which measures the **longest common subsequence** between the reference and candidate summaries, and **ROUGE-S** and **ROUGE-SU**, which measure the number of **skip bigrams** between the reference and candidate summaries. A skip bigram is a pair of words in their sentence order, but allowing for any number of other words to appear between the pair.

While ROUGE is the most commonly applied automatic baseline, it is not as applicable to summarization as similar metrics like BLEU are to machine translation. This is because human summarizers seem to disagree strongly about which sentences to include in a summary, making even the overlap of humans with each other very low.

Pyramid Method

This difference in which sentences humans choose to extract has motivated human evaluation methods that attempt to focus more on meaning. One metric, the **Pyramid Method**, is a way of measuring how many units of meaning are shared between the candidate and reference summaries. The Pyramid Method also weights the units of meaning by importance; units of meaning that occur in more of the human summaries are weighted more highly. The units of meaning are called **Summary Content Units** (SCU), which are sub-sentential semantic units that roughly correspond to propositions or coherent pieces of propositions.

Summary Content Units

In the Pyramid Method, humans label the Summary Content Units in each reference and candidate summary, and then an overlap measure is computed.

Let's see an example from Nenkova et al. (2007) of how two SCUs are labeled in sentences from six human abstracts. We'll first show sentences from the human summaries indexed by a letter (corresponding to one of the six human summaries) and a number (the position of the sentence in the human summary):

A1. The industrial espionage case involving GM and VW began with the hiring of Jose Ignacio Lopez, an employee of GM subsidiary Adam Opel, by VW as a production director.

B3. However, he left GM for VW under circumstances, which along with ensuing events, were described by a German judge as "potentially the biggest-ever case of industrial espionage".

C6. He left GM for VW *in March 1993*.

D6. The issue stems from the alleged recruitment of GM's eccentric and visionary Basque-born procurement chief Jose Ignacio Lopez de Arriortura and seven of Lopez's business colleagues.

E1. *On March 16, 1993*, with Japanese car import quotas to Europe expiring in two years, renowned cost-cutter, Agnacio Lopez De Arriortura, left his job as head of purchasing at General Motor's Opel, Germany, to become Volkswagen's Purchasing and Production director.

F3. *In March 1993*, Lopez and seven other GM executives moved to VW overnight.

The annotators first identify similar sentences, like those above, and then label SCUs. The underlined and italicized spans of words in the above sentences result in the following two SCUs, each one with a weight corresponding to the number of summaries in which it appears (six for the first SCU, and three for the second):

SCU1 (w=6): *Lopez left GM for VW*
A1. the hiring of Jose Ignacio Lopez, an employee of GM . . . by VW
B3. he left GM for VW
C6. He left GM for VW
D6. recruitment of GMs . . . Jose Ignacio Lopez
E1. Agnacio Lopez De Arriortura, left his job . . . at General Motors Opel . . . to become Volkswagens . . . director
F3. Lopez . . . GM . . . moved to VW

SCU2 (w=3) *Lopez changes employers in March 1993*
C6. in March, 1993
E1. On March 16, 1993
F3. In March 1993

Once the annotation is done, the informativeness of a given summary can be measured as the ratio of the sum of the weights of its SCUs to the weight of an optimal summary with the same number of SCUs. See the Historical Notes section at the end of the chapter for more details and pointers to the literature.

Random sentence baseline
Leading sentence baseline

The standard baselines for evaluating summaries are the **random sentences** baseline and the **leading sentences** baseline. Assuming we are evaluating summaries of length N sentences, the random baseline just chooses N random sentences, and the leading baseline chooses the first N sentences. The leading sentences method is quite a strong baseline and many proposed summarization algorithms fail to beat it.

23.8 Summary

- The dominant models of information retrieval represent the meanings of documents and queries as bags of words.
- The **vector space model** views documents and queries as vectors in a large multidimensional space. The similarity between documents and queries or other documents can be measured by the cosine of the angle between the vectors.
- The main components of a factoid question-answering system are the **question classification** module to determine the named entity type of the answer, a **passage retrieval** module to identify relevant passages, and an answer processing module to extract and format the final answer.
- Factoid question answers can be evaluated with **mean reciprocal rank** (**MRR**).
- Summarization can be **abstractive** or **extractive**; most current algorithms are extractive.
- Three components of **summarization algorithms** are **content selection**, **information ordering**, and **sentence realization**.

- Current single-document summarization algorithms focus mainly on **sentence extraction**, relying on features like **position** in the discourse, **word informativeness**, **cue phrases**, and **sentence length**.
- Multiple-document summarization algorithms often perform **sentence simplification** on document sentences.
- **Redundancy avoidance** is important in multiple-document summarization; it is often implemented by the addition of a redundancy penalty term like **MMR** into sentence extraction.
- **Information-ordering** algorithms in multi-document summarization are often based on maintaining **coherence**.
- **Query-focused summarization** can be done by slight modifications to **generic summarization** algorithms or by using information-extraction methods.

Bibliographical and Historical Notes

Luhn (1957) is generally credited with first advancing the notion of fully automatic indexing of documents based on their contents. Over the years, Salton's SMART project (Salton, 1971) at Cornell developed or evaluated many of the most important notions in information retrieval, including the vector model, term weighting schemes, relevance feedback, and the use of cosine as a similarity metric. The notion of using inverse document frequency in term weighting is due to Sparck Jones (1972). The original notion of relevance feedback is due to Rocchio (1971).

Probabilistic IR

An alternative to the vector model that we have not covered is the **probabilistic model** originally shown to be effective by Robinson and Sparck Jones (1976). See Crestani et al. (1998) and Chapter 11 of Manning et al. (2008) on probabilistic models in information retrieval.

Manning et al. (2008) is a comprehensive modern text on information retrieval. Good but slightly older texts include Baeza-Yates and Ribeiro-Neto (1999) and Frakes and Baeza-Yates (1992); older classic texts include Salton and McGill (1983) and van Rijsbergen (1975). Many of the classic papers in the field can be found in Sparck Jones and Willett (1997). Current work is published in the annual proceedings of the ACM Special Interest Group on Information Retrieval (SIGIR). The U.S. National Institute of Standards and Technology (NIST) has run an annual evaluation project for text information retrieval and extraction called the Text REtrieval Conference (TREC) since the early 1990s; the conference proceedings from TREC contain results from these standardized evaluations. The primary journals in the field are the *Journal of the American Society of Information Sciences*, *ACM Transactions on Information Systems*, *Information Processing and Management*, and *Information Retrieval*.

Question answering was one of the earliest tasks for NLP systems in the 1960s and 1970s (Green et al., 1961; Simmons, 1965; Woods et al., 1972; Lehnert, 1977), but the field lay dormant for a few decades until the need for querying the Web brought the task back into focus. The TREC QA track began in 1999, and a wide variety of factoid and non-factoid systems have been competing in annual evaluations since then.

In addition to the references in this chapter, see Strzalkowski and Harabagiu (2006) for a collection of recent research papers.

Research on text summarization began with the work of Luhn (1958) on extractive methods for the automatic generation of abstracts, focusing on surface features like term frequency, and the later work of Edmunson (1969) incorporating positional features as well. Term-based features were also used in the early application of automatic summarization at Chemical Abstracts Service (Pollock and Zamora, 1975). The 1970s and 1980s saw a number of approaches grounded in AI methodology such as scripts (DeJong, 1982), semantic networks (Reimer and Hahn, 1988), or combinations of AI and statistical methods (Rau et al., 1989).

The work of Kupiec et al. (1995) on training a sentence classifier with supervised machine learning led to many statistical methods for sentence extraction. Around the turn of the century, the growth of the Web led naturally to interest in multi-document summarization and query-focused summarization.

SumBasic *Centroid*

There have been a wide variety of algorithms for the main components of summarizers. The simple, unsupervised, log-linear content-selection algorithm we describe is simplified from the **SumBasic** algorithm of Nenkova and Vanderwende (2005), Vanderwende et al. (2007), and the **centroid** algorithm of Radev et al. (2000) and Radev et al. (2001). A number of algorithms for information ordering have used entity coherence, including Kibble and Power (2000), Lapata (2003), Karamanis and Manurung (2002), Karamanis (2003), Barzilay and Lapata (2005), and Barzilay and Lapata (2008). Algorithms for combining multiple cues for coherence and searching for the optimal ordering include Althaus et al. (2004), based on linear programming, the genetic algorithms of Mellish et al. (1998) and Karamanis and Manurung (2002), and the Soricut and Marcu (2006) algorithm, which uses A* search based on IDL expressions. Karamanis (2007) showed that adding coherence based on rhetorical relations to entity coherence didn't improve sentence ordering. See Lapata (2006, 2003), Karamanis et al. (2004), and Karamanis (2006) on methods for evaluating information ordering.

Sentence compression is an especially active area of research. Early algorithms focused on the use of syntactic knowledge for eliminating less important words or phrases (Grefenstette, 1998; Mani et al., 1999; Jing, 2000). Recent research has focused on using supervised machine learning, in which a parallel corpus of documents together with their human summaries is used to compute the probability that particular words or parse nodes will be pruned. Methods include the use of maximum entropy (Riezler et al., 2003), the noisy channel model and synchronous context-free grammars (Galley and McKeown, 2007; Knight and Marcu, 2000; Turner and Charniak, 2005; Daumé III and Marcu, 2002), Integer Linear Programming (Clarke and Lapata, 2007), and large-margin learning (McDonald, 2006). These methods rely on various features, especially including syntactic or parse knowledge (Jing, 2000; Dorr et al., 2003; Siddharthan et al., 2004; Galley and McKeown, 2007; Zajic et al., 2007; Conroy et al., 2006; Vanderwende et al., 2007), but also including coherence information (Clarke and Lapata, 2007). Alternative recent methods are able to function without these kinds of parallel document/summary corpora (Hori and Furui, 2004; Turner and Charniak, 2005; Clarke and Lapata, 2006).

See Daumé III and Marcu (2006) for a recent Bayesian model of query-focused summarization.

For more information on summarization evaluation, see Nenkova et al. (2007), Passonneau et al. (2005), and Passonneau (2006) for details on the Pyramid Method, van Halteren and Teufel (2003) and Teufel and van Halteren (2004) on related semantic-coverage evaluation methods, and Lin and Demner-Fushman (2005) on the link between evaluations for summarization and question answering. A NIST program starting in 2001, the Document Understanding Conference (DUC), has sponsored an annual evaluation of summarization algorithms. These have included single-document, multiple-document, and query-focused summarization; proceedings from the annual workshop are available online.

Mani and Maybury (1999) is the definitive collection of classic papers on summarization. Sparck Jones (2007) is a good recent survey, and Mani (2001) is the standard textbook.

Paraphrase detection

The task of **paraphrase detection** is an important task related to improving recall in question answering and avoiding redundancy in summarization, and it is also relevant for tasks like textual entailment. See Lin and Pantel (2001), Barzilay and Lee (2003), Pang et al. (2003), Dolan et al. (2004), Quirk et al. (2004) for representative papers on techniques for detecting paraphrases.

Text categorization

Another task related to information retrieval and summarization is the **text categorization** task, which is to assign a new document to one of a pre-existing set of document classes. The standard approach is to use supervised machine learning to train classifiers on a set of documents that have been labeled with the correct class. A most important application of text categorization is for **spam detection**.

Spam detection

Exercises

23.1 Pose the following queries to your favorite Web search engine.

Who did the Vice President kill?
Who killed the former Treasury Secretary?

Do an error-analysis on the returned snippets and pages. What are the sources of the errors? How might these errors be addressed by a more intelligent question-answering system?

23.2 Do some error analysis on Web-based question answering. Choose 10 questions and type them all into two different search engines. Analyze the errors. For example, what kinds of questions could neither system answer? Which kinds of questions did one work better on? Was there a type of question that could be answered just from the snippets?

23.3 Read Brill et al. (2002). Implement a simple version of the AskMSR system.

23.4 Apply the system you developed for the last question to a small, closed, set of Web pages of interest. For example, you could use the set of pages that describe the undergraduate degree and course requirements at a university. How does the restriction to a small collection affect the performance of the system?