

## Introduction to Information Retrieval and Web Search<sup>1</sup>

In most of the chapters in this book so far, we have discussed techniques for modeling, designing, querying, transaction processing of, and managing *structured data*. In Section 12.1 we discussed the difference between structured, semistructured, and unstructured data. Information retrieval deals mainly with *unstructured data*, and the techniques for indexing, searching, and retrieving information from large collections of unstructured documents. In this chapter we will provide an introduction to information retrieval. This is a very broad topic, so we will focus on the similarities and differences between information retrieval and database technologies, and on the indexing techniques that form the basis of many information retrieval systems.

This chapter is organized as follows. In Section 27.1 we introduce information retrieval (IR) concepts and discuss how IR differs from traditional databases. Section 27.2 is devoted to a discussion of retrieval models, which form the basis for IR search. Section 27.3 covers different types of queries in IR systems. Section 27.4 discusses text preprocessing, and Section 27.5 provides an overview of IR indexing, which is at the heart of any IR system. In Section 27.6 we describe the various evaluation metrics for IR systems performance. Section 27.7 details Web analysis and its relationship to information retrieval, and Section 27.8 briefly introduces the current trends in IR. Section 27.9 summarizes the chapter. For a limited overview of IR, we suggest that students read Sections 27.1 through 27.6.

---

<sup>1</sup>This chapter is coauthored with Saurav Sahay of the Georgia Institute of Technology.

## 27.1 Information Retrieval (IR) Concepts

**Information retrieval** is the process of retrieving documents from a collection in response to a query (or a search request) by a user. This section provides an overview of information retrieval (IR) concepts. In Section 27.1.1, we introduce information retrieval in general and then discuss the different kinds and levels of search that IR encompasses. In Section 27.1.2, we compare IR and database technologies. Section 27.1.3 gives a brief history of IR. We then present the different modes of user interaction with IR systems in Section 27.1.4. In Section 27.1.5, we describe the typical IR process with a detailed set of tasks and then with a simplified process flow, and end with a brief discussion of digital libraries and the Web.

### 27.1.1 Introduction to Information Retrieval

We first review the distinction between structured and unstructured data (see Section 12.1) to see how information retrieval differs from structured data management. Consider a relation (or table) called HOUSES with the attributes:

HOUSES(Lot#, Address, Square\_footage, Listed\_price)

This is an example of *structured data*. We can compare this relation with home-buying contract documents, which are examples of *unstructured data*. These types of documents can vary from city to city, and even county to county, within a given state in the United States. Typically, a contract document in a particular state will have a standard list of clauses described in paragraphs within sections of the document, with some predetermined (fixed) text and some variable areas whose content is to be supplied by the specific buyer and seller. Other variable information would include interest rate for financing, down-payment amount, closing dates, and so on. The documents could also possibly include some pictures taken during a home inspection. The information content in such documents can be considered *unstructured data* that can be stored in a variety of possible arrangements and formats. By **unstructured information**, we generally mean information that does not have a well-defined formal model and corresponding formal language for representation and reasoning, but rather is based on understanding of natural language.

With the advent of the World Wide Web (or Web, for short), the volume of unstructured information stored in messages and documents that contain textual and multimedia information has exploded. These documents are stored in a variety of standard formats, including HTML, XML (see Chapter 12), and several audio and video formatting standards. Information retrieval deals with the problems of storing, indexing, and retrieving (searching) such information to satisfy the needs of users. The problems that IR deals with are exacerbated by the fact that the number of Web pages and the number of social interaction events is already in the billions, and is growing at a phenomenal rate. All forms of unstructured data described above are being added at the rates of millions per day, expanding the searchable space on the Web at rapidly increasing rates.

Historically, **information retrieval** is “the discipline that deals with the structure, analysis, organization, storage, searching, and retrieval of information” as defined by Gerald Salton, an IR pioneer.<sup>2</sup> We can enhance the definition slightly to say that it applies in the context of unstructured documents to satisfy a user’s information needs. This field has existed even longer than the database field, and was originally concerned with retrieval of cataloged information in libraries based on titles, authors, topics, and keywords. In academic programs, the field of IR has long been a part of Library and Information Science programs. Information in the context of IR does not require machine-understandable structures, such as in relational database systems. Examples of such information include written texts, abstracts, documents, books, Web pages, e-mails, instant messages, and collections from digital libraries. Therefore, all loosely represented (unstructured) or semistructured information is also part of the IR discipline.

We introduced XML modeling and retrieval in Chapter 12 and discussed advanced data types, including spatial, temporal, and multimedia data, in Chapter 26. RDBMS vendors are providing modules to support many of these data types, as well as XML data, in the newer versions of their products, sometimes referred to as *extended RDBMSs*, or *object-relational database management systems* (ORDBMSs, see Chapter 11). The challenge of dealing with unstructured data is largely an information retrieval problem, although database researchers have been applying database indexing and search techniques to some of these problems.

IR systems go beyond database systems in that they do not limit the user to a specific query language, nor do they expect the user to know the structure (schema) or content of a particular database. IR systems use a user’s information need expressed as a **free-form search request** (sometimes called a **keyword search query**, or just **query**) for interpretation by the system. Whereas the IR field historically dealt with cataloging, processing, and accessing text in the form of documents for decades, in today’s world the use of Web search engines is becoming the dominant way to find information. The traditional problems of text indexing and making collections of documents searchable have been transformed by making the Web itself into a quickly accessible repository of human knowledge.

An IR system can be characterized at different levels: by types of *users*, types of *data*, and the types of the *information need*, along with the size and scale of the information repository it addresses. Different IR systems are designed to address specific problems that require a combination of different characteristics. These characteristics can be briefly described as follows:

**Types of Users.** The user may be an *expert user* (for example, a curator or a librarian), who is searching for specific information that is clear in his/her mind and forms relevant queries for the task, or a *layperson user* with a generic information need. The latter cannot create highly relevant queries for search (for

---

<sup>2</sup>See Salton’s 1968 book entitled *Automatic Information Organization and Retrieval*.

example, students trying to find information about a new topic, researchers trying to assimilate different points of view about a historical issue, a scientist verifying a claim by another scientist, or a person trying to shop for clothing).

**Types of Data.** Search systems can be tailored to specific types of data. For example, the problem of retrieving information about a specific topic may be handled more efficiently by customized search systems that are built to collect and retrieve only information related to that specific topic. The information repository could be hierarchically organized based on a concept or topic hierarchy. These topical *domain-specific* or *vertical IR systems* are not as large as or as diverse as the generic World Wide Web, which contains information on all kinds of topics. Given that these domain-specific collections exist and may have been acquired through a specific process, they can be exploited much more efficiently by a specialized system.

**Types of Information Need.** In the context of Web search, users' information needs may be defined as navigational, informational, or transactional.<sup>3</sup> **Navigational search** refers to finding a particular piece of information (such as the Georgia Tech University Website) that a user needs quickly. The purpose of **informational search** is to find current information about a topic (such as research activities in the college of computing at Georgia Tech—this is the classic IR system task). The goal of **transactional search** is to reach a site where further interaction happens (such as joining a social network, product shopping, online reservations, accessing databases, and so on).

**Levels of Scale.** In the words of Nobel Laureate Herbert Simon,

*What information consumes is rather obvious: it consumes the attention of its recipients. Hence a wealth of information creates a poverty of attention, and a need to allocate that attention efficiently among the overabundance of information sources that might consume it.*<sup>4</sup>

This overabundance of information sources in effect creates a high noise-to-signal ratio in IR systems. Especially on the Web, where billions of pages are indexed, IR interfaces are built with efficient scalable algorithms for distributed searching, indexing, caching, merging, and fault tolerance. IR search engines can be limited in level to more specific collections of documents. **Enterprise search systems** offer IR solutions for searching different entities in an enterprise's **intranet**, which consists of the network of computers within that enterprise. The searchable entities include e-mails, corporate documents, manuals, charts, and presentations, as well as reports related to people, meetings, and projects. They still typically deal with hundreds of millions of entities in large global enterprises. On a smaller scale, there are personal information systems such as those on desktops and laptops, called **desktop search engines** (for example, Google Desktop), for retrieving files, folders, and different kinds of entities stored on the computer. There are peer-to-peer systems, such as

<sup>3</sup>See Broder (2002) for details.

<sup>4</sup>From Simon (1971), "Designing Organizations for an Information-Rich World."

BitTorrent, which allows sharing of music in the form of audio files, as well as specialized search engines for audio, such as Lycos and Yahoo! audio search.

## 27.1.2 Databases and IR Systems: A Comparison

Within the computer science discipline, databases and IR systems are closely related fields. Databases deal with structured information retrieval through well-defined formal languages for representation and manipulation based on the theoretically founded data models. Efficient algorithms have been developed for operators that allow rapid execution of complex queries. IR, on the other hand, deals with unstructured search with possibly vague query or search semantics and without a well-defined logical schematic representation. Some of the key differences between databases and IR systems are listed in Table 27.1.

Whereas databases have fixed schemas defined in some data model such as the relational model, an IR system has no fixed data model; it views data or documents according to some scheme, such as the vector space model, to aid in query processing (see Section 27.2). Databases using the relational model employ SQL for queries and transactions. The queries are mapped into relational algebra operations and search algorithms (see Chapter 19) and return a new relation (table) as the query result, providing an exact answer to the query for the current state of the database. In IR systems, there is no fixed language for defining the structure (schema) of the document or for operating on the document—queries tend to be a set of query terms (keywords) or a free-form natural language phrase. An IR query result is a list of document ids, or some pieces of text or multimedia objects (images, videos, and so on), or a list of links to Web pages.

The result of a database query is an exact answer; if no matching records (tuples) are found in the relation, the result is empty (null). On the other hand, the answer to a user request in an IR query represents the IR system's best attempt at retrieving the

**Table 27.1** A Comparison of Databases and IR Systems

Databases	IR Systems
<ul style="list-style-type: none"> <li>■ Structured data</li> <li>■ Schema driven</li> <li>■ Relational (or object, hierarchical, and network) model is predominant</li> <li>■ Structured query model</li> <li>■ Rich metadata operations</li> <li>■ Query returns data</li> <li>■ Results are based on exact matching (always correct)</li> </ul>	<ul style="list-style-type: none"> <li>■ Unstructured data</li> <li>■ No fixed schema; various data models (e.g., vector space model)</li> <li>■ Free-form query models</li> <li>■ Rich data operations</li> <li>■ Search request returns list or pointers to documents</li> <li>■ Results are based on approximate matching and measures of effectiveness (may be imprecise and ranked)</li> </ul>

information most relevant to that query. Whereas database systems maintain a large amount of metadata and allow their use in query optimization, the operations in IR systems rely on the data values themselves and their occurrence frequencies. Complex statistical analysis is sometimes performed to determine the *relevance* of each document or parts of a document to the user request.

### 27.1.3 A Brief History of IR

Information retrieval has been a common task since the times of ancient civilizations, which devised ways to organize, store, and catalog documents and records. Media such as papyrus scrolls and stone tablets were used to record documented information in ancient times. These efforts allowed knowledge to be retained and transferred among generations. With the emergence of public libraries and the printing press, large-scale methods for producing, collecting, archiving, and distributing documents and books evolved. As computers and automatic storage systems emerged, the need to apply these methods to computerized systems arose. Several techniques emerged in the 1950s, such as the seminal work of H. P. Luhn,<sup>5</sup> who proposed using words and their frequency counts as indexing units for documents, and using measures of word overlap between queries and documents as the retrieval criterion. It was soon realized that storing large amounts of text was not difficult. The harder task was to search for and retrieve that information selectively for users with specific information needs. Methods that explored word distribution statistics gave rise to the choice of keywords based on their distribution properties<sup>6</sup> and keyword-based weighting schemes.

The earlier experiments with document retrieval systems such as SMART<sup>7</sup> in the 1960s adopted the *inverted file organization* based on keywords and their weights as the method of indexing (see Section 27.5). Serial (or sequential) organization proved inadequate if queries required fast, near real-time response times. Proper organization of these files became an important area of study; document classification and clustering schemes ensued. The scale of retrieval experiments remained a challenge due to lack of availability of large text collections. This soon changed with the World Wide Web. Also, the Text Retrieval Conference (TREC) was launched by NIST (National Institute of Standards and Technology) in 1992 as a part of the TIPSTER program<sup>8</sup> with the goal of providing a platform for evaluating information retrieval methodologies and facilitating technology transfer to develop IR products.

A **search engine** is a practical application of information retrieval to large-scale document collections. With significant advances in computers and communications technologies, people today have interactive access to enormous amounts of user-generated distributed content on the Web. This has spurred the rapid growth

---

<sup>5</sup>See Luhn (1957) "A statistical approach to mechanized encoding and searching of literary information."

<sup>6</sup>See Salton, Yang, and Yu (1975).

<sup>7</sup>For details, see Buckley et al. (1993).

<sup>8</sup>For details, see Harman (1992).

in search engine technology, where search engines are trying to discover different kinds of real-time content found on the Web. The part of a search engine responsible for discovering, analyzing, and indexing these new documents is known as a **crawler**. Other types of search engines exist for specific domains of knowledge. For example, the biomedical literature search database was started in the 1970s and is now supported by the PubMed search engine,<sup>9</sup> which gives access to over 20 million abstracts.

While continuous progress is being made to tailor search results to the needs of an end user, the challenge remains in providing high-quality, pertinent, and timely information that is precisely aligned to the information needs of individual users.

### 27.1.4 Modes of Interaction in IR Systems

In the beginning of Section 27.1, we defined information retrieval as the process of retrieving documents from a collection in response to a query (or a search request) by a user. Typically the collection is made up of documents containing unstructured data. Other kinds of documents include images, audio recordings, video strips, and maps. Data may be scattered nonuniformly in these documents with no definitive structure. A **query** is a set of **terms** (also referred to as **keywords**) used by the searcher to specify an information need (for example, the terms ‘databases’ and ‘operating systems’ may be regarded as a query to a computer science bibliographic database). An informational request or a search query may also be a natural language phrase or a question (for example, “What is the currency of China?” or “Find Italian restaurants in Sarasota, Florida.”).

There are two main modes of interaction with IR systems—retrieval and browsing—which, although similar in goal, are accomplished through different interaction tasks. **Retrieval** is concerned with the extraction of relevant information from a repository of documents through an IR query, while **browsing** signifies the activity of a user visiting or navigating through similar or related documents based on the user’s assessment of relevance. During browsing, a user’s information need may not be defined *a priori* and is flexible. Consider the following browsing scenario: A user specifies ‘Atlanta’ as a keyword. The information retrieval system retrieves links to relevant result documents containing various aspects of Atlanta for the user. The user comes across the term ‘Georgia Tech’ in one of the returned documents, and uses some access technique (such as clicking on the phrase ‘Georgia Tech’ in a document, which has a built-in link) and visits documents about Georgia Tech in the same or a different Website (repository). There the user finds an entry for ‘Athletics’ that leads the user to information about various athletic programs at Georgia Tech. Eventually, the user ends his search at the Fall schedule for the Yellow Jackets football team, which he finds to be of great interest. This user activity is known as browsing. **Hyperlinks** are used to interconnect Web pages and are mainly used for browsing. **Anchor texts** are text phrases within documents used to label hyperlinks and are very relevant to browsing.

<sup>9</sup>See [www.ncbi.nlm.nih.gov/pubmed/](http://www.ncbi.nlm.nih.gov/pubmed/).

**Web search** combines both aspects—browsing and retrieval—and is one of the main applications of information retrieval today. Web pages are analogous to documents. Web search engines maintain an indexed repository of Web pages, usually using the technique of inverted indexing (see Section 27.5). They retrieve the most relevant Web pages for the user in response to the user’s search request with a possible ranking in descending order of relevance. The **rank of a Webpage** in a retrieved set is the measure of its relevance to the query that generated the result set.

### 27.1.5 Generic IR Pipeline

As we mentioned earlier, documents are made up of unstructured natural language text composed of character strings from English and other languages. Common examples of documents include newswire services (such as AP or Reuters), corporate manuals and reports, government notices, Web page articles, blogs, tweets, books, and journal papers. There are two main approaches to IR: statistical and semantic.

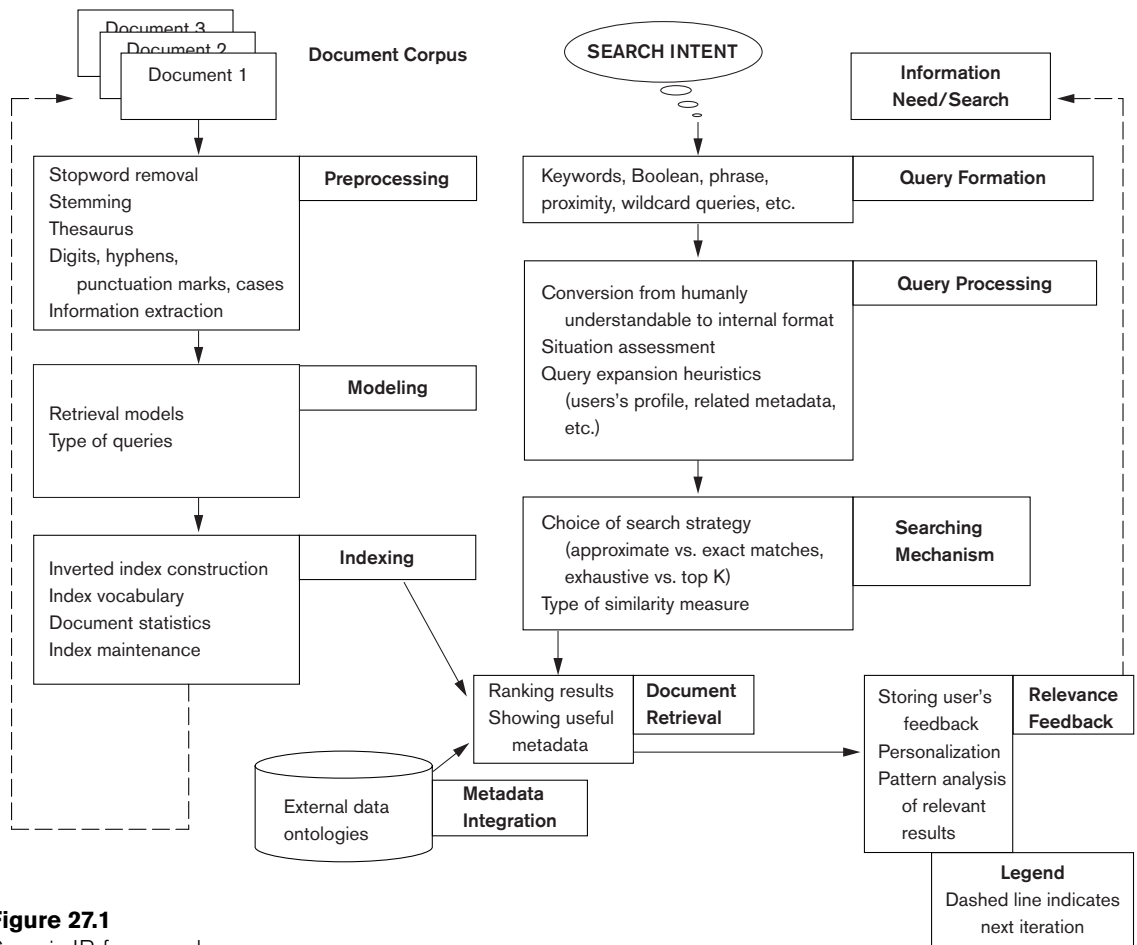
In a **statistical approach**, documents are analyzed and broken down into chunks of text (words, phrases, or  $n$ -grams, which are all subsequences of length  $n$  characters in a text or document) and each word or phrase is counted, weighted, and measured for relevance or importance. These words and their properties are then compared with the query terms for potential degree of match to produce a ranked list of resulting documents that contain the words. Statistical approaches are further classified based on the method employed. The three main statistical approaches are Boolean, vector space, and probabilistic (see Section 27.2).

**Semantic approaches** to IR use knowledge-based techniques of retrieval that broadly rely on the syntactic, lexical, sentential, discourse-based, and pragmatic levels of knowledge understanding. In practice, semantic approaches also apply some form of statistical analysis to improve the retrieval process.

Figure 27.1 shows the various stages involved in an IR processing system. The steps shown on the left in Figure 27.1 are typically offline processes, which prepare a set of documents for efficient retrieval; these are document preprocessing, document modeling, and indexing. The steps involved in query formation, query processing, searching mechanism, document retrieval, and relevance feedback are shown on the right in Figure 27.1. In each box, we highlight the important concepts and issues. The rest of this chapter describes some of the concepts involved in the various tasks within the IR process shown in Figure 27.1.

Figure 27.2 shows a simplified IR processing pipeline. In order to perform retrieval on documents, the documents are first represented in a form suitable for retrieval. The significant terms and their properties are extracted from the documents and are represented in a document index where the words/terms and their properties are stored in a matrix that contains these terms and the references to the documents that contain them. This index is then converted into an inverted index (see Figure 27.4) of a word/term vs. document matrix. Given the query words, the documents



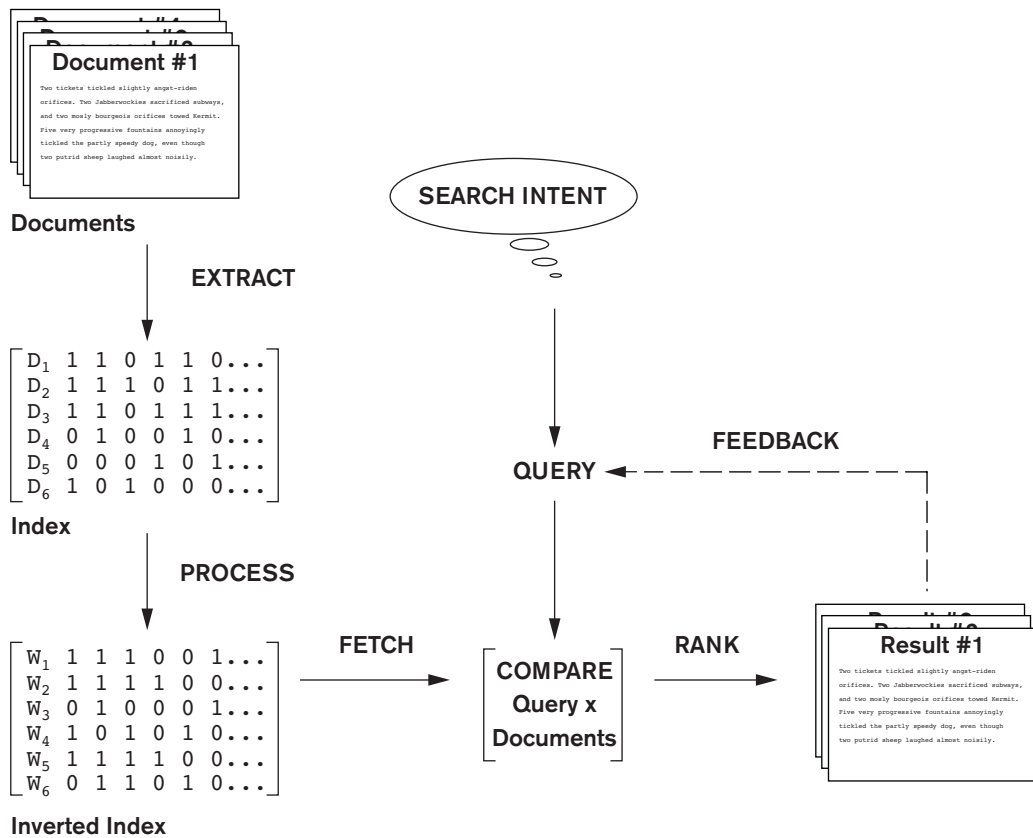


**Figure 27.1**  
Generic IR framework.

containing these words—and the document properties, such as date of creation, author, and type of document—are fetched from the inverted index and compared with the query. This comparison results in a ranked list shown to the user. The user can then provide feedback on the results that triggers implicit or explicit query expansion to fetch results that are more relevant for the user. Most IR systems allow for an interactive search where the query and the results are successively refined.

## 27.2 Retrieval Models

In this section we briefly describe the important models of IR. These are the three main statistical models—Boolean, vector space, and probabilistic—and the semantic model.



**Figure 27.2**  
Simplified IR process pipeline.

### 27.2.1 Boolean Model

In this model, documents are represented as a set of *terms*. Queries are formulated as a combination of terms using the standard Boolean logic set-theoretic operators such as AND, OR and NOT. Retrieval and relevance are considered as binary concepts in this model, so the retrieved elements are an “exact match” retrieval of relevant documents. There is no notion of ranking of resulting documents. All retrieved documents are considered equally important—a major simplification that does not consider frequencies of document terms or their proximity to other terms compared against the query terms.

Boolean retrieval models lack sophisticated ranking algorithms and are among the earliest and simplest information retrieval models. These models make it easy to associate metadata information and write queries that match the contents of the

documents as well as other properties of documents, such as date of creation, author, and type of document.

## 27.2.2 Vector Space Model

The vector space model provides a framework in which term weighting, ranking of retrieved documents, and relevance feedback are possible. Documents are represented as *features* and *weights* of term features in an  $n$ -dimensional vector space of terms. **Features** are a subset of the terms in a *set of documents* that are deemed most relevant to an IR search for this particular set of documents. The process of selecting these important terms (features) and their properties as a sparse (limited) list out of the very large number of available terms (the vocabulary can contain hundreds of thousands of terms) is independent of the model specification. The query is also specified as a terms vector (vector of features), and this is compared to the document vectors for similarity/relevance assessment.

The similarity assessment function that compares two vectors is not inherent to the model—different similarity functions can be used. However, the cosine of the angle between the query and document vector is a commonly used function for similarity assessment. As the angle between the vectors decreases, the cosine of the angle approaches one, meaning that the similarity of the query with a document vector increases. Terms (features) are weighted proportional to their frequency counts to reflect the importance of terms in the calculation of relevance measure. This is different from the Boolean model, which does not take into account the frequency of words in the document for relevance match.

In the vector model, the *document term weight*  $w_{ij}$  (for term  $i$  in document  $j$ ) is represented based on some variation of the TF (term frequency) or TF-IDF (term frequency-inverse document frequency) scheme (as we will describe below). **TF-IDF** is a statistical weight measure that is used to evaluate the importance of a document word in a collection of documents. The following formula is typically used:

$$\text{cosine}(d_j, q) = \frac{\langle d_j, q \rangle}{\|d_j\| \times \|q\|} = \frac{\sum_{i=1}^{|V|} w_{ij} \times w_{iq}}{\sqrt{\sum_{i=1}^{|V|} w_{ij}^2} \times \sqrt{\sum_{i=1}^{|V|} w_{iq}^2}}$$

In the formula given above, we use the following symbols:

- $d_j$  is the document vector.
- $q$  is the query vector.
- $w_{ij}$  is the weight of term  $i$  in document  $j$ .
- $w_{iq}$  is the weight of term  $i$  in query vector  $q$ .
- $|V|$  is the number of dimensions in the vector that is the total number of important keywords (or features).

TF-IDF uses the product of normalized frequency of a term  $i$  ( $TF_{ij}$ ) in document  $D_j$  and the inverse document frequency of the term  $i$  ( $IDF_i$ ) to weight a term in a

document. The idea is that terms that capture the essence of a document occur frequently in the document (that is, their TF is high), but if such a term were to be a good term that discriminates the document from others, it must occur in only a few documents in the general population (that is, its IDF should be high as well).

IDF values can be easily computed for a fixed collection of documents. In case of Web search engines, taking a representative sample of documents approximates IDF computation. The following formulas can be used:

$$TF_{ij} = f_{ij} / \sum_{i=1 \text{ to } |V|} f_{ij}$$

$$IDF_i = \log(N / n_i)$$

In these formulas, the meaning of the symbols is:

- $TF_{ij}$  is the normalized term frequency of term  $i$  in document  $D_j$ .
- $f_{ij}$  is the number of occurrences of term  $i$  in document  $D_j$ .
- $IDF_i$  is the inverse document frequency weight for term  $i$ .
- $N$  is the number of documents in the collection.
- $n_i$  is the number of documents in which term  $i$  occurs.

Note that if a term  $i$  occurs in all documents, then  $n_i = N$  and hence  $IDF_i = \log(1)$  becomes zero, nullifying its importance and creating a situation where division by zero can occur. The weight of term  $i$  in document  $j$ ,  $w_{ij}$  is computed based on its TF-IDF value in some techniques. To prevent division by zero, it is common to add a 1 to the denominator in the formulae such as the cosine formula above.

Sometimes, the relevance of the document with respect to a query ( $\text{rel}(D_j, Q)$ ) is directly measured as the sum of the TF-IDF values of the terms in the Query  $Q$ :

$$\text{rel}(D_j, Q) = \sum_{i \in Q} TF_{ij} \times IDF_i$$

The normalization factor (similar to the denominator of the cosine formula) is incorporated into the TF-IDF formula itself, thereby measuring relevance of a document to the query by the computation of the dot product of the query and document vectors.

The Rocchio<sup>10</sup> algorithm is a well-known relevance feedback algorithm based on the vector space model that modifies the initial query vector and its weights in response to user-identified relevant documents. It expands the original query vector  $q$  to a new vector  $q_e$  as follows:

$$q_e = \alpha q + \frac{\beta}{|D_r|} \sum_{d_r \in D_r} d_r - \frac{\gamma}{|D_{ir}|} \sum_{d_{ir} \in D_{ir}} d_{ir},$$

---

<sup>10</sup>See Rocchio (1971).

Here,  $D_r$  and  $D_{nr}$  are relevant and nonrelevant document sets and  $\alpha$ ,  $\beta$ , and  $\gamma$  are parameters of the equation. The values of these parameters determine how the feedback affects the original query, and these may be determined after a number of trial-and-error experiments.

### 27.2.3 Probabilistic Model

The similarity measures in the vector space model are somewhat ad hoc. For example, the model assumes that those documents closer to the query in cosine space are more relevant to the query vector. In the probabilistic model, a more concrete and definitive approach is taken: ranking documents by their estimated probability of relevance with respect to the query and the document. This is the basis of the *Probability Ranking Principle* developed by Robertson:<sup>11</sup>

In the probabilistic framework, the IR system has to decide whether the documents belong to the **relevant set** or the **nonrelevant set** for a query. To make this decision, it is assumed that a predefined relevant set and nonrelevant set exist for the query, and the task is to calculate the probability that the document belongs to the relevant set and compare that with the probability that the document belongs to the nonrelevant set.

Given the document representation  $D$  of a document, estimating the relevance  $R$  and nonrelevance  $NR$  of that document involves computation of conditional probability  $P(R|D)$  and  $P(NR|D)$ . These conditional probabilities can be calculated using Bayes' Rule:<sup>12</sup>

$$P(R|D) = P(D|R) \times P(R)/P(D)$$

$$P(NR|D) = P(D|NR) \times P(NR)/P(D)$$

A document  $D$  is classified as relevant if  $P(R|D) > P(NR|D)$ . Discarding the constant  $P(D)$ , this is equivalent to saying that a document is relevant if:

$$P(D|R) \times P(R) > P(D|NR) \times P(NR)$$

The likelihood ratio  $P(D|R)/P(D|NR)$  is used as a score to determine the likelihood of the document with representation  $D$  belonging to the relevant set.

The *term independence* or *Naïve Bayes* assumption is used to estimate  $P(D|R)$  using computation of  $P(t_i|R)$  for term  $t_i$ . The likelihood ratios  $P(D|R)/P(D|NR)$  of documents are used as a proxy for ranking based on the assumption that highly ranked documents will have a high likelihood of belonging to the relevant set.<sup>13</sup>

<sup>11</sup>For a description of the Cheshire II system, see Robertson (1997).

<sup>12</sup>Bayes' theorem is a standard technique for measuring likelihood; see Howson and Urbach (1993), for example.

<sup>13</sup>Readers should refer to Croft et al. (2009) pages 246–247 for a detailed description.

With some reasonable assumptions and estimates about the probabilistic model along with extensions for incorporating query term weights and document term weights in the model, a probabilistic ranking algorithm called **BM25** (Best Match 25) is quite popular. This weighting scheme has evolved from several versions of the **Okapi**<sup>14</sup> system.

The Okapi weight for Document  $d_j$  and query  $q$  is computed by the formula below. Additional notations are as follows:

- $t_i$  is a term.
- $f_{ij}$  is the raw frequency count of term  $t_i$  in document  $d_j$ .
- $f_{iq}$  is the raw frequency count of term  $t_i$  in query  $q$ .
- $N$  is the total number of documents in the collection.
- $df_i$  is the number of documents that contain the term  $t_i$ .
- $dl_j$  is the document length (in bytes) of  $d_j$ .
- $avdl$  is the average document length of the collection.

The Okapi relevance score of a document  $d_j$  for a query  $q$  is given by the equation below, where  $k_1$  (between 1.0–2.0),  $b$  (usually 0.75), and  $k_2$  (between 1–1000) are parameters:

$$\text{okapi}(d_j, q) = \sum_{t_i \in q, d_j} \ln \frac{N - df_i + 0.5}{df_i + 0.5} \times \frac{(k_1 + 1)f_{ij}}{k_1 \left( 1 - b + b \frac{dl_j}{avdl} \right) + f_{ij}} \times \frac{(k_2 + 1)f_{iq}}{k_2 + f_{iq}},$$

## 27.2.4 Semantic Model

However sophisticated the above statistical models become, they can miss many relevant documents because those models do not capture the complete meaning or information need conveyed by a user's query. In semantic models, the process of matching documents to a given query is based on concept level and semantic matching instead of index term (keyword) matching. This allows retrieval of relevant documents that share meaningful associations with other documents in the query result, even when these associations are not inherently observed or statistically captured.

Semantic approaches include different levels of analysis, such as morphological, syntactic, and semantic analysis, to retrieve documents more effectively. In **morphological analysis**, roots and affixes are analyzed to determine the parts of speech (nouns, verbs, adjectives, and so on) of the words. Following morphological analysis, **syntactic analysis** follows to parse and analyze complete phrases in documents. Finally, the semantic methods have to resolve word ambiguities and/or generate relevant synonyms based on the **semantic relationships** between levels of structural entities in documents (words, paragraphs, pages, or entire documents).

<sup>14</sup>City University of London Okapi System by Robertson, Walker, and Hancock-Beaulieu (1995).

The development of a sophisticated semantic system requires complex knowledge bases of semantic information as well as retrieval heuristics. These systems often require techniques from artificial intelligence and expert systems. Knowledge bases like Cyc<sup>15</sup> and WordNet<sup>16</sup> have been developed for use in *knowledge-based IR systems* based on semantic models. The Cyc knowledge base, for example, is a representation of a vast quantity of commonsense knowledge about assertions (over 2.5 million facts and rules) interrelating more than 155,000 concepts for reasoning about the objects and events of everyday life. WordNet is an extensive thesaurus (over 115,000 concepts) that is very popular and is used by many systems and is under continuous development (see Section 27.4.3).

## 27.3 Types of Queries in IR Systems

Different keywords are associated with the document set during the process of indexing. These keywords generally consist of words, phrases, and other characterizations of documents such as date created, author names, and type of document. They are used by an IR system to build an inverted index (see Section 27.5), which is then consulted during the search. The queries formulated by users are compared to the set of index keywords. Most IR systems also allow the use of Boolean and other operators to build a complex query. The query language with these operators enriches the expressiveness of a user's information need.

### 27.3.1 Keyword Queries

Keyword-based queries are the simplest and most commonly used forms of IR queries: the user just enters keyword combinations to retrieve documents. The query keyword terms are implicitly connected by a logical AND operator. A query such as 'database concepts' retrieves documents that contain both the words 'database' and 'concepts' at the top of the retrieved results. In addition, most systems also retrieve documents that contain only 'database' or only 'concepts' in their text. Some systems remove most commonly occurring words (such as *a*, *the*, *of*, and so on, called **stopwords**) as a preprocessing step before sending the filtered query keywords to the IR engine. Most IR systems do not pay attention to the ordering of these words in the query. All retrieval models provide support for keyword queries.

### 27.3.2 Boolean Queries

Some IR systems allow using the AND, OR, NOT, ( ), +, and – Boolean operators in combinations of keyword formulations. AND requires that both terms be found. OR lets either term be found. NOT means any record containing the second term will be excluded. '( )' means the Boolean operators can be nested using parentheses. '+' is equivalent to AND, requiring the term; the '+' should be placed directly in front

---

<sup>15</sup>See Lenat (1995).

<sup>16</sup>See Miller (1990) for a detailed description of WordNet.

of the search term. ‘-’ is equivalent to AND NOT and means to exclude the term; the ‘~’ should be placed directly in front of the search term not wanted. Complex Boolean queries can be built out of these operators and their combinations, and they are evaluated according to the classical rules of Boolean algebra. No ranking is possible, because a document either satisfies such a query (is “relevant”) or does not satisfy it (is “nonrelevant”). A document is retrieved for a Boolean query if the query is logically true as an exact match in the document. Users generally do not use combinations of these complex Boolean operators, and IR systems support a restricted version of these set operators. Boolean retrieval models can directly support different Boolean operator implementations for these kinds of queries.

### 27.3.3 Phrase Queries

When documents are represented using an inverted keyword index for searching, the relative order of the terms in the document is lost. In order to perform exact phrase retrieval, these phrases should be encoded in the inverted index or implemented differently (with relative positions of word occurrences in documents). A phrase query consists of a sequence of words that makes up a phrase. The phrase is generally enclosed within double quotes. Each retrieved document must contain at least one instance of the exact phrase. Phrase searching is a more restricted and specific version of proximity searching that we mention below. For example, a phrase searching query could be ‘conceptual database design’. If phrases are indexed by the retrieval model, any retrieval model can be used for these query types. A phrase thesaurus may also be used in semantic models for fast dictionary searching for phrases.

### 27.3.4 Proximity Queries

Proximity search refers to a search that accounts for how close within a record multiple terms should be to each other. The most commonly used proximity search option is a phrase search that requires terms to be in the exact order. Other proximity operators can specify how close terms should be to each other. Some will also specify the order of the search terms. Each search engine can define proximity operators differently, and the search engines use various operator names such as NEAR, ADJ(adjacent), or AFTER. In some cases, a sequence of single words is given, together with a maximum allowed distance between them. Vector space models that also maintain information about positions and offsets of tokens (words) have robust implementations for this query type. However, providing support for complex proximity operators becomes computationally expensive because it requires the time-consuming preprocessing of documents, and is thus suitable for smaller document collections rather than for the Web.

### 27.3.5 Wildcard Queries

Wildcard searching is generally meant to support regular expressions and pattern matching-based searching in text. In IR systems, certain kinds of wildcard search support may be implemented—usually words with any trailing characters (for



example, ‘data\*’ would retrieve *data*, *database*, *datapoint*, *dataset*, and so on). Providing support for wildcard searches in IR systems involves preprocessing overhead and is not considered worth the cost by many Web search engines today. Retrieval models do not directly provide support for this query type.

### 27.3.6 Natural Language Queries

There are a few natural language search engines that aim to understand the structure and meaning of queries written in natural language text, generally as a question or narrative. This is an active area of research that employs techniques like shallow semantic parsing of text, or query reformulations based on natural language understanding. The system tries to formulate answers for such queries from retrieved results. Some search systems are starting to provide natural language interfaces to provide answers to specific types of questions, such as definition and factoid questions, which ask for definitions of technical terms or common facts that can be retrieved from specialized databases. Such questions are usually easier to answer because there are strong linguistic patterns giving clues to specific types of sentences—for example, ‘defined as’ or ‘refers to’. Semantic models can provide support for this query type.

## 27.4 Text Preprocessing

In this section we review the commonly used text preprocessing techniques that are part of the text processing task in Figure 27.1.

### 27.4.1 Stopword Removal

**Stopwords** are very commonly used words in a language that play a major role in the formation of a sentence but which seldom contribute to the meaning of that sentence. Words that are expected to occur in 80 percent or more of the documents in a collection are typically referred to as *stopwords*, and they are rendered potentially useless. Because of the commonness and function of these words, they do not contribute much to the relevance of a document for a query search. Examples include words such as *the*, *of*, *to*, *a*, *and*, *in*, *said*, *for*, *that*, *was*, *on*, *he*, *is*, *with*, *at*, *by*, and *it*. These words are presented here with decreasing frequency of occurrence from a large corpus of documents called **AP89**.<sup>17</sup> The first six of these words account for 20 percent of all words in the listing, and the most frequent 50 words account for 40 percent of all text.

Removal of stopwords from a document must be performed before indexing. Articles, prepositions, conjunctions, and some pronouns are generally classified as stopwords. Queries must also be preprocessed for stopword removal before the actual retrieval process. Removal of stopwords results in elimination of possible spurious indexes, thereby reducing the size of an index structure by about 40

<sup>17</sup>For details, see Croft et al. (2009), pages 75–90.

percent or more. However, doing so could impact the recall if the stopword is an integral part of a query (for example, a search for the phrase ‘To be or not to be,’ where removal of stopwords makes the query inappropriate, as all the words in the phrase are stopwords). Many search engines do not employ query stopword removal for this reason.

### 27.4.2 Stemming

A **stem** of a word is defined as the word obtained after trimming the suffix and prefix of an original word. For example, ‘comput’ is the stem word for *computer*, *computing*, and *computation*. These suffixes and prefixes are very common in the English language for supporting the notion of verbs, tenses, and plural forms. **Stemming** reduces the different forms of the word formed by inflection (due to plurals or tenses) and derivation to a common stem.

A stemming algorithm can be applied to reduce any word to its stem. In English, the most famous stemming algorithm is Martin Porter’s stemming algorithm. The Porter stemmer<sup>18</sup> is a simplified version of Lovin’s technique that uses a reduced set of about 60 rules (from 260 suffix patterns in Lovin’s technique) and organizes them into sets; conflicts within one subset of rules are resolved before going on to the next. Using stemming for preprocessing data results in a decrease in the size of the indexing structure and an increase in recall, possibly at the cost of precision.

### 27.4.3 Utilizing a Thesaurus

A **thesaurus** comprises a precompiled list of important concepts and the main word that describes each concept for a particular domain of knowledge. For each concept in this list, a set of synonyms and related words is also compiled.<sup>19</sup> Thus, a synonym can be converted to its matching concept during preprocessing. This preprocessing step assists in providing a standard vocabulary for indexing and searching. Usage of a thesaurus, also known as a *collection of synonyms*, has a substantial impact on the recall of information systems. This process can be complicated because many words have different meanings in different contexts.

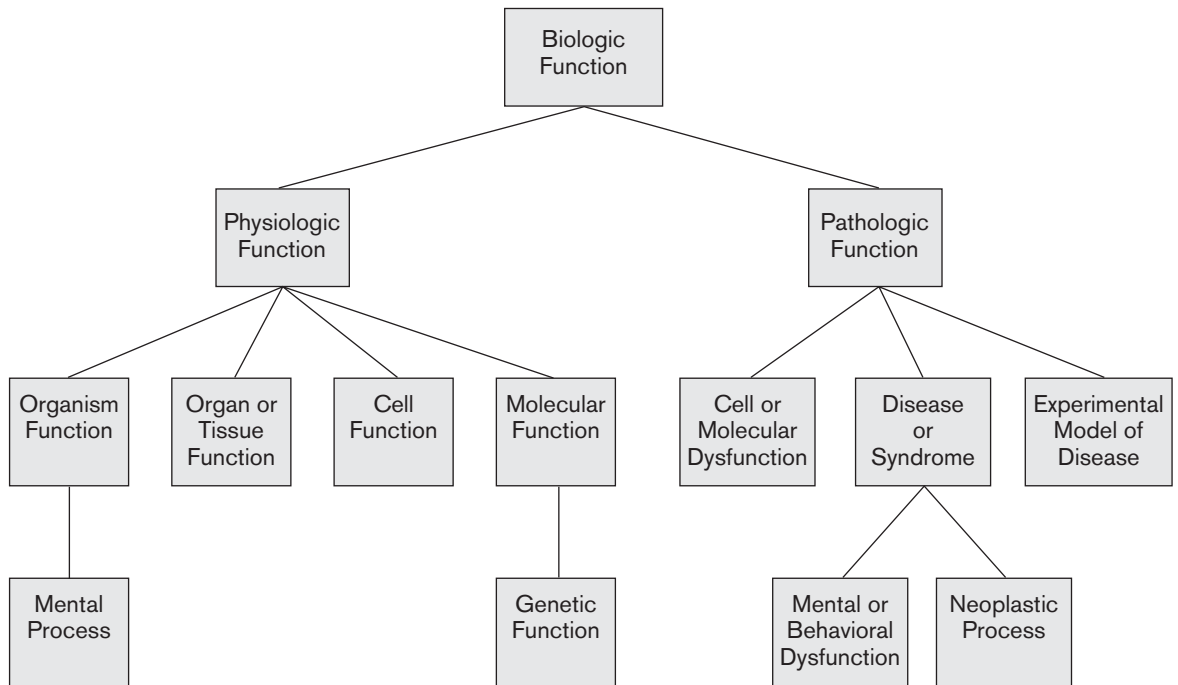
UMLS<sup>20</sup> is a large biomedical thesaurus of millions of concepts (called the *Metathesaurus*) and a semantic network of meta concepts and relationships that organize the Metathesaurus (see Figure 27.3). The concepts are assigned labels from the semantic network. This thesaurus of concepts contains synonyms of medical terms, hierarchies of broader and narrower terms, and other relationships among words and concepts that make it a very extensive resource for information retrieval of documents in the medical domain. Figure 27.3 illustrates part of the UMLS Semantic Network.

---

<sup>18</sup>See Porter (1980).

<sup>19</sup>See Baeza-Yates and Ribeiro-Neto (1999).

<sup>20</sup>Unified Medical Language System from the National Library of Medicine.

**Figure 27.3**

A Portion of the UMLS Semantic Network: "Biologic Function" Hierarchy

Source: UMLS Reference Manual, National Library of Medicine.

**WordNet**<sup>21</sup> is a manually constructed thesaurus that groups words into strict synonym sets called *synsets*. These synsets are divided into noun, verb, adjective, and adverb categories. Within each category, these synsets are linked together by appropriate relationships such as class/subclass or "is-a" relationships for nouns.

WordNet is based on the idea of using a controlled vocabulary for indexing, thereby eliminating redundancies. It is also useful in providing assistance to users with locating terms for proper query formulation.

#### 27.4.4 Other Preprocessing Steps: Digits, Hyphens, Punctuation Marks, Cases

Digits, dates, phone numbers, e-mail addresses, URLs, and other standard types of text may or may not be removed during preprocessing. Web search engines, however, index them in order to use this type of information in the document

<sup>21</sup>See Fellbaum (1998) for a detailed description of WordNet.

metadata to improve precision and recall (see Section 27.6 for detailed definitions of *precision* and *recall*).

Hyphens and punctuation marks may be handled in different ways. Either the entire phrase with the hyphens/punctuation marks may be used, or they may be eliminated. In some systems, the character representing the hyphen/punctuation mark may be removed, or may be replaced with a space. Different information retrieval systems follow different rules of processing. Handling hyphens automatically can be complex: it can either be done as a classification problem, or more commonly by some heuristic rules.

Most information retrieval systems perform case-insensitive search, converting all the letters of the text to uppercase or lowercase. It is also worth noting that many of these text preprocessing steps are language specific, such as involving accents and diacritics and the idiosyncrasies that are associated with a particular language.

### 27.4.5 Information Extraction

**Information extraction** (IE) is a generic term used for extracting structured content from text. Text analytic tasks such as identifying noun phrases, facts, events, people, places, and relationships are examples of IE tasks. These tasks are also called *named entity recognition tasks* and use rule-based approaches with either a thesaurus, regular expressions and grammars, or probabilistic approaches. For IR and search applications, IE technologies are mostly used to identify contextually relevant features that involve text analysis, matching, and categorization for improving the relevance of search systems. Language technologies using part-of-speech tagging are applied to semantically annotate the documents with extracted features to aid search relevance.

## 27.5 Inverted Indexing

The simplest way to search for occurrences of query terms in text collections can be performed by sequentially scanning the text. This kind of online searching is only appropriate when text collections are quite small. Most information retrieval systems process the text collections to create indexes and operate upon the inverted index data structure (refer to the indexing task in Figure 27.1). An inverted index structure comprises vocabulary and document information. **Vocabulary** is a set of distinct query terms in the document set. Each term in a vocabulary set has an associated collection of information about the documents that contain the term, such as document id, occurrence count, and offsets within the document where the term occurs. The simplest form of vocabulary terms consists of words or individual tokens of the documents. In some cases, these vocabulary terms also consist of phrases, n-grams, entities, links, names, dates, or manually assigned descriptor terms from documents and/or Web pages. For each term in the vocabulary, the corresponding document ids, occurrence locations of the term in each document, number of occurrences of the term in each document, and other relevant information may be stored in the document information section.

Weights are assigned to document terms to represent an estimate of the usefulness of the given term as a descriptor for distinguishing the given document from other documents in the same collection. A term may be a better descriptor of one document than of another by the weighting process (see Section 27.2).

An **inverted index** of a document collection is a data structure that attaches distinct terms with a list of all documents that contains the term. The process of inverted index construction involves the extraction and processing steps shown in Figure 27.2. Acquired text is first preprocessed and the documents are represented with the vocabulary terms. Documents' statistics are collected in document lookup tables. Statistics generally include counts of vocabulary terms in individual documents as well as different collections, their positions of occurrence within the documents, and the lengths of the documents. The vocabulary terms are weighted at indexing time according to different criteria for collections. For example, in some cases terms in the titles of the documents may be weighted more heavily than terms that occur in other parts of the documents.

One of the most popular weighting schemes is the TF-IDF (term frequency-inverse document frequency) metric that we described in Section 27.2. For a given term this weighting scheme distinguishes to some extent the documents in which the term occurs more often from those in which the term occurs very little or never. These weights are normalized to account for varying document lengths, further ensuring that longer documents with proportionately more occurrences of a word are not favored for retrieval over shorter documents with proportionately fewer occurrences. These processed document-term streams (matrices) are then inverted into term-document streams (matrices) for further IR steps.

Figure 27.4 shows an illustration of term-document-position vectors for the four illustrative terms—*example*, *inverted*, *index*, and *market*—which refer to the three documents and the position where they occur in those documents.

The different steps involved in inverted index construction can be summarized as follows:

1. Break the documents into vocabulary terms by tokenizing, cleansing, stopword removal, stemming, and/or use of an additional thesaurus as vocabulary.
2. Collect document statistics and store the statistics in a document lookup table.
3. Invert the document-term stream into a term-document stream along with additional information such as term frequencies, term positions, and term weights.

Searching for relevant documents from the inverted index, given a set of query terms, is generally a three-step process.

1. **Vocabulary search.** If the query comprises multiple terms, they are separated and treated as independent terms. Each term is searched in the vocabulary. Various data structures, like variations of B<sup>+</sup>-tree or hashing, may be

**Document 1**

This example shows an example of an inverted index.

**Document 2**

Inverted index is a data structure for associating terms to documents.

**Document 2**

Stock market index is used for capturing the sentiments of the financial market.

ID	Term	Document: position
1.	example	1:2, 1:5
2.	inverted	1:8, 2:1
3.	index	1:9, 2:2, 3:3
4.	market	3:2, 3:13

**Figure 27.4**  
Example of an inverted index.

used to optimize the search process. Query terms may also be ordered in lexicographic order to improve space efficiency.

2. **Document information retrieval.** The document information for each term is retrieved.
3. **Manipulation of retrieved information.** The document information vector for each term obtained in step 2 is now processed further to incorporate various forms of query logic. Various kinds of queries like prefix, range, context, and proximity queries are processed in this step to construct the final result based on the document collections returned in step 2.

## 27.6 Evaluation Measures of Search Relevance





Without proper evaluation techniques, one cannot compare and measure the relevance of different retrieval models and IR systems in order to make improvements.

Evaluation techniques of IR systems measure the *topical relevance* and *user relevance*. **Topical relevance** measures the extent to which the topic of a result matches the topic of the query. Mapping one's information need with "perfect" queries is a cognitive task, and many users are not able to effectively form queries that would retrieve results more suited to their information need. Also, since a major chunk of user queries are informational in nature, there is no fixed set of right answers to show to the user. **User relevance** is a term used to describe the "goodness" of a retrieved result with regard to the user's information need. User relevance includes other implicit factors, such as user perception, context, timeliness, the user's environment, and current task needs. Evaluating user relevance may also involve subjective analysis and study of user retrieval tasks to capture some of the properties of implicit factors involved in accounting for users' bias for judging performance.

In Web information retrieval, no binary classification decision is made on whether a document is relevant or nonrelevant to a query (whereas the Boolean (or binary) retrieval model uses this scheme, as we discussed in Section 27.2.1). Instead, a ranking of the documents is produced for the user. Therefore, some evaluation measures focus on comparing different rankings produced by IR systems. We discuss some of these measures next.

### 27.6.1 Recall and Precision

Recall and precision metrics are based on the binary relevance assumption (whether each document is relevant or nonrelevant to the query). **Recall** is defined as the number of relevant documents retrieved by a search divided by the total number of existing relevant documents. **Precision** is defined as the number of relevant documents retrieved by a search divided by the total number of documents retrieved by that search. Figure 27.5 is a pictorial representation of the terms *retrieved vs. relevant* and shows how search results relate to four different sets of documents.

		Relevant?	
		Yes	No
Retrieved?	Yes	 Hits TP	 False Alarms FP
	No	Misses FN 	Correct Rejections TN 

**Figure 27.5**  
Retrieved vs. relevant search results.

The notation for Figure 27.5 is as follows:

- TP: true positive
- FP: false positive
- FN: false negative
- TN: true negative

The terms *true positive*, *false positive*, *false negative*, and *true negative* are generally used in any type of classification tasks to compare the given classification of an item with the desired correct classification. Using the term *hits* for the documents that truly or “correctly” match the user request, we can define:

$$\text{Recall} = |\text{Hits}|/|\text{Relevant}|$$

$$\text{Precision} = |\text{Hits}|/|\text{Retrieved}|$$

Recall and precision can also be defined in a ranked retrieval setting. The Recall at rank position  $i$  for document  $d_i^q$  (denoted by  $r(i)$ ) ( $d_i^q$  is the retrieved document at position  $i$  for query  $q$ ) is the fraction of relevant documents from  $d_1^q$  to  $d_i^q$  in the result set for the query. Let the set of relevant documents from  $d_1^q$  to  $d_i^q$  in that set be  $S_i$  with cardinality  $|S_i|$ . Let  $(|D_q|)$  be the size of relevant documents for the query. In this case,  $|S_i| \leq |D_q|$ . Then:

$$\text{Recall } r(i) = |S_i|/|D_q|$$

The Precision at rank position  $i$  or document  $d_i^q$  (denoted by  $p(i)$ ) is the fraction of documents from  $d_1^q$  to  $d_i^q$  in the result set that are relevant:

$$\text{Precision } p(i) = |S_i|/i$$

Table 27.2 illustrates the  $p(i)$ ,  $r(i)$ , and average precision (discussed in the next section) metrics. It can be seen that recall can be increased by presenting more results to the user, but this approach runs the risk of decreasing the precision. In the

**Table 27.2** Precision and Recall for Ranked Retrieval

Doc. No.	Rank Position $i$	Relevant	Precision( $i$ )	Recall( $i$ )
10	1	Yes	1/1 = 100%	1/10 = 10%
2	2	Yes	2/2 = 100%	2/10 = 20%
3	3	Yes	3/3 = 100%	3/10 = 30%
5	4	No	3/4 = 75%	3/10 = 30%
17	5	No	3/5 = 60%	3/10 = 30%
34	6	No	3/6 = 50%	3/10 = 30%
215	7	Yes	4/7 = 57.1%	4/10 = 40%
33	8	Yes	5/8 = 62.5%	5/10 = 50%
45	9	No	5/9 = 55.5%	5/10 = 50%
16	10	Yes	6/10 = 60%	6/10 = 60%



example, the number of relevant documents for some query = 10. The rank position and the relevance of an individual document are shown. The precision and recall value can be computed at each position within the ranked list as shown in the last two columns.

## 27.6.2 Average Precision

Average precision is computed based on the precision at each relevant document in the ranking. This measure is useful for computing a single precision value to compare different retrieval algorithms on a query  $q$ .

$$P_{\text{avg}} = \sum_{d^i \in D_q} P(i) / |D_q|$$

Consider the sample precision values of relevant documents in Table 27.2. The average precision ( $P_{\text{avg}}$  value) for the example in Table 27.2 is  $P(1) + P(2) + P(3) + P(7) + P(8) + P(10)/6 = 79.93$  percent (only relevant documents are considered in this calculation). Many good algorithms tend to have high top-k average precision for small values of k, with correspondingly low values of recall.

## 27.6.3 Recall/Precision Curve

A recall/precision curve can be drawn based on the recall and precision values at each rank position, where the x-axis is the recall and the y-axis is the precision. Instead of using the precision and recall at each rank position, the curve is commonly plotted using recall levels  $r(i)$  at 0 percent, 10 percent, 20 percent...100 percent. The curve usually has a negative slope, reflecting the inverse relationship between precision and recall.

## 27.6.4 F-Score

F-score ( $F$ ) is the harmonic mean of the precision ( $p$ ) and recall ( $r$ ) values. High precision is achieved almost always at the expense of recall and vice versa. It is a matter of the application's context whether to tune the system for high precision or high recall. F-score is a single measure that combines precision and recall to compare different result sets:

$$F = \frac{2pr}{p+r}$$

One of the properties of harmonic mean is that the harmonic mean of two numbers tends to be closer to the smaller of the two. Thus  $F$  is automatically biased toward the smaller of the precision and recall values. Therefore, for a high F-score, both precision and recall must be high.

$$F = \frac{2}{\frac{1}{p} + \frac{1}{r}}$$

## 27.7 Web Search and Analysis<sup>22</sup>

The emergence of the Web has brought millions of users to search for information, which is stored in a very large number of active sites. To make this information accessible, search engines such as Google and Yahoo! have to crawl and index these sites and document collections in their index databases. Moreover, search engines have to regularly update their indexes given the dynamic nature of the Web as new Web sites are created and current ones are updated or deleted. Since there are many millions of pages available on the Web on different topics, search engines have to apply many sophisticated techniques such as link analysis to identify the importance of pages.

There are other types of search engines besides the ones that regularly crawl the Web and create automatic indexes: these are human-powered, vertical search engines or metasearch engines. These search engines are developed with the help of computer-assisted systems to aid the curators with the process of assigning indexes. They consist of manually created specialized Web directories that are hierarchically organized indexes to guide user navigation to different resources on the Web. **Vertical search engines** are customized topic-specific search engines that crawl and index a specific collection of documents on the Web and provide search results from that specific collection. **Metasearch engines** are built on top of search engines: they query different search engines simultaneously and aggregate and provide search results from these sources.

Another source of searchable Web documents is digital libraries. **Digital libraries** can be broadly defined as collections of electronic resources and services for the delivery of materials in a variety of formats. These collections may include a university's library catalog, catalogs from a group of participating universities as in the State of Florida University System, or a compilation of multiple external resources on the World Wide Web such as Google Scholar or the IEEE/ACM index. These interfaces provide universal access to different types of content—such as books, articles, audio, and video—situated in different database systems and remote repositories. Similar to real libraries, these digital collections are maintained via a catalog and organized in categories for online reference. Digital libraries “include personal, distributed, and centralized collections such as online public access catalogs (OPACs) and bibliographic databases, distributed document databases, scholarly and professional discussion lists and electronic journals, other online databases, forums, and bulletin boards.”<sup>23</sup>

### 27.7.1 Web Analysis and Its Relationship to Information Retrieval

In addition to browsing and searching the Web, another important activity closely related to information retrieval is to *analyze* or *mine* information on the Web for

<sup>22</sup>The contributions of Pranesh P. Ranganathan and Hari P. Kumar to this section is appreciated.

<sup>23</sup>Covi and Kling (1996), page 672.

new information of interest. (We discuss mining of data from files and databases in Chapter 28.) Application of data analysis techniques for discovery and analysis of useful information from the Web is known as **Web analysis**. Over the past few years the World Wide Web has emerged as an important repository of information for many day-to-day applications for individual consumers, as well as a significant platform for e-commerce and for social networking. These properties make it an interesting target for data analysis applications. The Web mining and analysis field is an integration of a wide range of fields spanning information retrieval, text analysis, natural language processing, data mining, machine learning, and statistical analysis.

The goals of Web analysis are to improve and personalize search results relevance and to identify trends that may be of value to various businesses and organizations. We elaborate on these goals next.

- **Finding relevant information.** People usually search for specific information on the Web by entering keywords in a search engine or browsing information portals and using services. Search services are constrained by search relevance problems since they have to map and approximate the information need of millions of users as an *a priori* task. Low *precision* (see Section 27.6) ensues due to results that are nonrelevant to the user. In the case of the Web, high *recall* (see section 27.6) is impossible to determine due to the inability to index all the pages on the Web. Also, measuring recall does not make sense since the user is concerned with only the top few documents. The most relevant feedback for the user is typically from only the top few results.
- **Personalization of the information.** Different people have different content and presentation preferences. By collecting personal information and then generating user-specific dynamic Web pages, the pages are personalized for the user. The customization tools used in various Web-based applications and services, such as click-through monitoring, eyeball tracking, explicit or implicit user profile learning, and dynamic service composition using Web APIs, are used for service adaptation and personalization. A personalization engine typically has algorithms that make use of the user's personalization information—collected by various tools—to generate user-specific search results.
- **Finding information of commercial value.** This problem deals with finding interesting patterns in users' interests, behaviors, and their use of products and services, which may be of commercial value. For example, businesses such as the automobile industry, clothing, shoes, and cosmetics may improve their services by identifying patterns such as usage trends and user preferences using various Web analysis techniques.

Based on the above goals, we can classify Web analysis into three categories: **Web content analysis**, which deals with extracting useful information/knowledge from Web page contents; **Web structure analysis**, which discovers knowledge from hyperlinks representing the structure of the Web; and **Web usage analysis**, which mines user access patterns from usage logs that record the activity of every user.

### 27.7.2 Searching the Web

The World Wide Web is a huge corpus of information, but locating resources that are both high quality and relevant to the needs of the user is very difficult. The set of Web pages taken as a whole has almost no unifying structure, with variability in authoring style and content, thereby making it more difficult to precisely locate needed information. Index-based search engines have been one of the prime tools by which users search for information on the Web. Web search engines **crawl** the Web and create an index to the Web for searching purposes. When a user specifies his need for information by supplying keywords, these Web search engines query their repository of indexes and produce links or URLs with abbreviated content as search results. There may be thousands of pages relevant to a particular query. A problem arises when only a few most relevant results are to be returned to the user. The discussion we had about querying and relevance-based ranking in IR systems in Sections 27.2 and 27.3 is applicable to Web search engines. These ranking algorithms explore the link structure of the Web.

Web pages, unlike standard text collections, contain connections to other Web pages or documents (via the use of hyperlinks), allowing users to browse from page to page. A **hyperlink** has two components: a **destination page** and an **anchor text** describing the link. For example, a person can link to the Yahoo! Website on his Web page with anchor text such as “My favorite Website.” Anchor texts can be thought of as being implicit endorsements. They provide very important latent human annotation. A person linking to other Web pages from his Web page is assumed to have some relation to those Web pages. Web search engines aim to distill results per their relevance and authority. There are many redundant hyperlinks, like the links to the homepage on every Web page of the Web site. Such hyperlinks must be eliminated from the search results by the search engines.

A **hub** is a Web page or a Website that links to a collection of prominent sites (authorities) on a common topic. A good **authority** is a page that is pointed to by many good hubs, while a good hub is a page that points to many good authorities. These ideas are used by the HITS ranking algorithm, which is described in Section 27.7.3. It is often found that authoritative pages are not very self-descriptive, and authorities on broad topics seldom link directly to one another. These properties of hyperlinks are being actively used to improve Web search engine result ranking and organize the results as hubs and authorities. We briefly discuss a couple of ranking algorithms below.

### 27.7.3 Analyzing the Link Structure of Web Pages

The goal of **Web structure analysis** is to generate structural summary about the Website and Web pages. It focuses on the inner structure of documents and deals with the link structure using hyperlinks at the interdocument level. The structure and content of Web pages are often combined for information retrieval by Web search engines. Given a collection of interconnected Web documents, interesting and informative facts describing their connectivity in the Web subset can be discovered. Web structure analysis is also used to reveal the structure of Web pages, which

helps with navigation and makes it possible to compare/integrate Web page schemes. This aspect of Web structure analysis facilitates Web document classification and clustering on the basis of structure.

**The PageRank Ranking Algorithm.** As discussed earlier, ranking algorithms are used to order search results based on relevance and authority. Google uses the well-known **PageRank** algorithm,<sup>24</sup> which is based on the “importance” of each page. Every Web page has a number of forward links (out-edges) and backlinks (in-edges). It is very difficult to determine all the backlinks of a Web page, while it is relatively straightforward to determine its forward links. According to the PageRank algorithm, highly linked pages are more important (have greater authority) than pages with fewer links. However, not all backlinks are important. A backlink to a page from a credible source is more important than a link from some arbitrary page. Thus a page has a high rank if the sum of the ranks of its backlinks is high. PageRank was an attempt to see how good an approximation to the “importance” of a page can be obtained from the link structure.

The computation of page ranking follows an iterative approach. PageRank of a Web page is calculated as a sum of the PageRanks of all its backlinks. PageRank treats the Web like a *Markov model*. An imaginary Web surfer visits an infinite string of pages by clicking randomly. The PageRank of a page is an estimate of how often the surfer winds up at a particular page. PageRank is a measure of query-independent importance of a page/node. For example, let  $P(X)$  be the PageRank of any page  $X$  and  $C(X)$  be the number of outgoing links from page  $X$ , and let  $d$  be the damping factor in the range  $0 < d < 1$ . Usually  $d$  is set to 0.85. Then PageRank for a page  $A$  can be calculated as:

$$P(A) = (1 - d) + d (P(T_1)/C(T_1) + \dots + P(T_n)/C(T_n))$$

Here  $T_1, T_2, \dots, T_n$  are the pages that point to Page  $A$  (that is, are citations to page  $A$ ). PageRank forms a probability distribution over Web pages, so the sum of all Web pages’ PageRanks is one.

**The HITS Ranking Algorithm.** The HITS<sup>25</sup> algorithm proposed by Jon Kleinberg is another type of ranking algorithm exploiting the link structure of the Web. The algorithm presumes that a good hub is a document that points to many hubs, and a good authority is a document that is pointed at by many other authorities. The algorithm contains two main steps: a sampling component and a weight-propagation component. The sampling component constructs a focused collection  $S$  of pages with the following properties:

1.  $S$  is relatively small.
2.  $S$  is rich in relevant pages.
3.  $S$  contains most (or a majority) of the strongest authorities.

<sup>24</sup>The PageRank algorithm was proposed by Lawrence Page (1998) and Sergey Brin, founders of Google. For more information, see <http://en.wikipedia.org/wiki/PageRank>.

<sup>25</sup>See Kleinberg (1999).

The weight component recursively calculates the hub and authority values for each document as follows:

1. Initialize hub and authority values for all pages in  $S$  by setting them to 1.
2. While (hub and authority values do not converge):
  - a. For each page in  $S$ , calculate authority value = Sum of hub values of all pages *pointing to* the current page.
  - b. For each page in  $S$ , calculate hub value = Sum of authority values of all pages *pointed at* by the current page.
  - c. Normalize hub and authority values such that sum of all hub values in  $S$  equals 1 and the sum of all authority values in  $S$  equals 1.

### 27.7.4 Web Content Analysis

As mentioned earlier, **Web content analysis** refers to the process of discovering useful information from Web content/data/documents. The **Web content data** consists of unstructured data such as free text from electronically stored documents, semi-structured data typically found as HTML documents with embedded image data, and more structured data such as tabular data, and pages in HTML, XML, or other markup languages generated as output from databases. More generally, the term *Web content* refers to any real data in the Web page that is intended for the user accessing that page. This usually consists of but is not limited to text and graphics.

We will first discuss some preliminary Web content analysis tasks and then look at the traditional analysis tasks of Web page classification and clustering later.

**Structured Data Extraction.** Structured data on the Web is often very important as it represents essential information, such as a structured table showing the airline flight schedule between two cities. There are several approaches to structured data extraction. One includes writing a **wrapper**, or a program that looks for different structural characteristics of the information on the page and extracts the right content. Another approach is to manually write an extraction program for each Website based on observed format patterns of the site, which is very labor intensive and time consuming. It does not scale to a large number of sites. A third approach is **wrapper induction** or **wrapper learning**, where the user first manually labels a set of training set pages, and the learning system generates rules—based on the learning pages—that are applied to extract target items from other Web pages. A fourth approach is the automatic approach, which aims to find patterns/grammars from the Web pages and then uses **wrapper generation** to produce a wrapper to extract data automatically.

**Web Information Integration.** The Web is immense and has millions of documents, authored by many different persons and organizations. Because of this, Web pages that contain similar information may have different syntax and different words that describe the same concepts. This creates the need for integrating

information from diverse Web pages. Two popular approaches for Web information integration are:

1. **Web query interface integration**, to enable querying multiple Web databases that are not visible in external interfaces and are hidden in the “deep Web.” The **deep Web**<sup>26</sup> consists of those pages that do not exist until they are created dynamically as the result of a specific database search, which produces some of the information in the page (see Chapter 14). Since traditional search engine crawlers cannot probe and collect information from such pages, the deep Web has heretofore been hidden from crawlers.
2. **Schema matching**, such as integrating directories and catalogs to come up with a global schema for applications. An example of such an application would be to combine a personal health record of an individual by matching and collecting data from various sources dynamically by cross-linking health records from multiple systems.

These approaches remain an area of active research and a detailed discussion of them is beyond the scope of this book. Consult the Selected Bibliography at the end of this chapter for further details.

**Ontology-Based Information Integration.** This task involves using ontologies to effectively combine information from multiple heterogeneous sources. Ontologies—formal models of representation with explicitly defined concepts and named relationships linking them—are used to address the issues of semantic heterogeneity in data sources. Different classes of approaches are used for information integration using ontologies.

- **Single ontology approaches** use one global ontology that provides a shared vocabulary for the specification of the semantics. They work if all information sources to be integrated provide nearly the same view on a domain of knowledge. For example, UMLS (described in Section 27.4.3) can serve as a common ontology for biomedical applications.
- In a **multiple ontology approach**, each information source is described by its own ontology. In principle, the “source ontology” can be a combination of several other ontologies but it cannot be assumed that the different “source ontologies” share the same vocabulary. Dealing with multiple, partially overlapping, and potentially conflicting ontologies is a very difficult problem faced by many applications, including those in bioinformatics and other complex area of knowledge.
- **Hybrid ontology approaches** are similar to multiple ontology approaches: the semantics of each source is described by its own ontology. But in order to make the source ontologies comparable to each other, they are built upon one global shared vocabulary. The shared vocabulary contains basic terms (the primitives) of a domain of knowledge. Because each term of source

---

<sup>26</sup>The deep Web as defined by Bergman (2001).

ontology is based on the primitives, the terms become more easily comparable than in multiple ontology approaches. The advantage of a hybrid approach is that new sources can be easily added without the need to modify the mappings or the shared vocabulary. In multiple and hybrid approaches, several research issues, such as ontology mapping, alignment, and merging, need to be addressed.

**Building Concept Hierarchies.** One common way of organizing search results is via a linear ranked list of documents. But for some users and applications, a better way to display results would be to create groupings of related documents in the search result. One way of organizing documents in a search result, and for organizing information in general, is by creating a **concept hierarchy**. The documents in a search result are organized into groups in a hierarchical fashion. Other related techniques to organize documents are through **classification** and **clustering** (see Chapter 28). Clustering creates groups of documents, where the documents in each group share many common concepts.

**Segmenting Web Pages and Detecting Noise.** There are many superfluous parts in a Web document, such as advertisements and navigation panels. The information and text in these superfluous parts should be eliminated as noise before classifying the documents based on their content. Hence, before applying classification or clustering algorithms to a set of documents, the areas or blocks of the documents that contain noise should be removed.

## 27.7.5 Approaches to Web Content Analysis

The two main approaches to Web content analysis are (1) agent based (IR view) and (2) database based (DB view).

**The agent-based approach** involves the development of sophisticated artificial intelligence systems that can act autonomously or semi-autonomously on behalf of a particular user, to discover and process Web-based information. Generally, the agent-based Web analysis systems can be placed into the following three categories:

- **Intelligent Web agents** are software agents that search for relevant information using characteristics of a particular application domain (and possibly a user profile) to organize and interpret the discovered information. For example, an intelligent agent that retrieves product information from a variety of vendor sites using only general information about the product domain.
- **Information Filtering/Categorization** is another technique that utilizes Web agents for categorizing Web documents. These Web agents use methods from information retrieval, and semantic information based on the links among various documents to organize documents into a concept hierarchy.
- **Personalized Web agents** are another type of Web agents that utilize the personal preferences of users to organize search results, or to discover information and documents that could be of value for a particular user. User



preferences could be learned from previous user choices, or from other individuals who are considered to have similar preferences to the user.

**The database-based approach** aims to infer the structure of the Website or to transform a Web site to organize it as a database so that better information management and querying on the Web become possible. This approach of Web content analysis primarily tries to model the data on the Web and integrate it so that more sophisticated queries than keyword-based search can be performed. These could be achieved by finding the schema of Web documents, building a Web document warehouse, a Web knowledge base, or a virtual database. The database-based approach may use a model such as the Object Exchange Model (OEM)<sup>27</sup> that represents semi-structured data by a labeled graph. The data in the OEM is viewed as a graph, with objects as the vertices and labels on the edges. Each object is identified by an object identifier and a value that is either atomic—such as integer, string, GIF image, or HTML document—or complex in the form of a set of object references.

The main focus of the database-based approach has been with the use of multilevel databases and Web query systems. A **multilevel database** at its lowest level is a database containing primitive semistructured information stored in various Web repositories, such as hypertext documents. At the higher levels, metadata or generalizations are extracted from lower levels and organized in structured collections such as relational or object-oriented databases. In a **Web query system**, information about the content and structure of Web documents is extracted and organized using database-like techniques. Query languages similar to SQL can then be used to search and query Web documents. They combine structural queries, based on the organization of hypertext documents, and content-based queries.

### 27.7.6 Web Usage Analysis

**Web usage analysis** is the application of data analysis techniques to discover usage patterns from Web data, in order to understand and better serve the needs of Web-based applications. This activity does not directly contribute to information retrieval; but it is important to improve or enhance the users' search experience.

**Web usage data** describes the pattern of usage of Web pages, such as IP addresses, page references, and the date and time of accesses for a user, user group, or an application. Web usage analysis typically consists of three main phases: preprocessing, pattern discovery, and pattern analysis.

1. **Preprocessing.** Preprocessing converts the information collected about usage statistics and patterns into a form that can be utilized by the pattern discovery methods. We use the term “page view” to refer to pages viewed or visited by a user. There are several different types of preprocessing techniques available:
  - **Usage preprocessing** analyzes the available collected data about usage patterns of users, applications, and groups of users. Because this data is often incomplete, the process is difficult. Data cleaning techniques are necessary to

<sup>27</sup>See Kosala and Blockeel (2000).

eliminate the impact of irrelevant items in the analysis result. Frequently, usage data is identified by an IP address, and consists of clicking streams that are collected at the server. Better data is available if a usage tracking process is installed at the client site.

- **Content preprocessing** is the process of converting text, image, scripts and other content into a form that can be used by the usage analysis. Often, this consists of performing content analysis such as classification or clustering. The clustering or classification techniques can group usage information for similar types of Web pages, so that usage patterns can be discovered for specific classes of Web pages that describe particular topics. Page views can also be classified according to their intended use, such as for sales or for discovery or for other uses.
- **Structure preprocessing:** The structure preprocessing can be done by parsing and reformatting the information about hyperlinks and structure between viewed pages. One difficulty is that the site structure may be dynamic and may have to be constructed for each server session.

## 2. Pattern Discovery

The techniques that are used in pattern discovery are based on methods from the fields of statistics, machine learning, pattern recognition, data analysis, data mining, and other similar areas. These techniques are adapted so they take into consideration the specific knowledge and characteristics for Web Analysis. For example, in association rule discovery (See Section 28.2), the notion of a transaction for market-basket analysis considers the items to be unordered. But the order of accessing of Web pages is important, and so it should be considered in Web usage analysis. Hence, pattern discovery involves mining sequences of page views. In general, using Web usage data, the following types of data mining activities may be performed for pattern discovery.

- **Statistical analysis.** Statistical techniques are the most common method to extract knowledge about visitors to a Website. By analyzing the session log, it is possible to apply statistical measures such as mean, median, and frequency count to parameters such as pages viewed, viewing time per page, length of navigation paths between pages, and other parameters that are relevant to Web usage analysis.
- **Association rules.** In the context of Web usage analysis, association rules refer to sets of pages that are accessed together with a support value exceeding some specified threshold. (See Section 28.2 on association rules.) These pages may not be directly connected to one another via hyperlinks. For example, association rule discovery may reveal a correlation between users who visited a page containing electronic products to those who visit a page about sporting equipment.
- **Clustering.** In the Web usage domain, there are two kinds of interesting clusters to be discovered: usage clusters and page clusters. **Clustering of users** tends to establish groups of users exhibiting similar browsing patterns.

Such knowledge is especially useful for inferring user demographics in order to perform market segmentation in E-commerce applications or provide personalized Web content to the users. **Clustering of pages** is based on the content of the pages, and pages with similar contents are grouped together. This type of clustering can be utilized in Internet search engines, and in tools that provide assistance to Web browsing.

- **Classification.** In the Web domain, one goal is to develop a profile of users belonging to a particular class or category. This requires extraction and selection of features that best describe the properties of a given class or category of users. As an example, an interesting pattern that may be discovered would be: 60% of users who placed an online order in /Product/Books are in the 18-25 age group and live in rented apartments.
- **Sequential patterns.** These kinds of patterns identify sequences of Web accesses, which may be used to predict the next set of Web pages to be accessed by a certain class of users. These patterns can be used by marketers to produce targeted advertisements on Web pages. Another type of sequential pattern pertains to which items are typically purchased following the purchase of a particular item. For example, after purchasing a computer, a printer is often purchased
- **Dependency modeling.** Dependency modeling aims to determine and model significant dependencies among the various variables in the Web domain. As an example, one may be interested to build a model representing the different stages a visitor undergoes while shopping in an online store based on the actions chosen (e.g., from a casual visitor to a serious potential buyer).

### 3. Pattern Analysis

The final step is to filter out those rules or patterns that are considered to be not of interest from the discovered patterns. The particular analysis methodology based on the application. One common technique for pattern analysis is to use a query language such as SQL to detect various patterns and relationships. Another technique involves loading of usage data into a data warehouse with ETL tools and performing OLAP operations to view it along multiple dimensions (see Section 29.3). It is common to use visualization techniques, such as graphing patterns or assigning colors to different values, to highlight patterns or trends in the data.

## 27.7.7 Practical Applications of Web Analysis

**Web Analytics.** The goal of **web analytics** is to understand and optimize the performance of Web usage. This requires collecting, analyzing, and performance monitoring of Internet usage data. On-site Web analytics measures the performance of a Website in a commercial context. This data is typically compared against key performance indicators to measure effectiveness or performance of the Website as a whole, and can be used to improve a Website or improve the marketing strategies.

**Web Spamming.** It has become increasingly important for companies and individuals to have their Websites/Web pages appear in the top search results. To achieve this, it is essential to understand search engine ranking algorithms and to present the information in one's page in such a way that the page is ranked high when the respective keywords are queried. There is a thin line separating legitimate page optimization for business purposes and spamming. **Web Spamming** is thus defined as a deliberate activity to promote one's page by manipulating the results returned by the search engines. Web analysis may be used to detect such pages and discard them from search results.

**Web Security.** Web analysis can be used to find interesting usage patterns of Websites. If any flaw in a Website has been exploited, it can be inferred using Web analysis thereby allowing the design of more robust Websites. For example, the backdoor or information leak of Web servers can be detected by using Web analysis techniques on some abnormal Web application log data. Security analysis techniques such as intrusion detection and denial of service attacks are based on Web access pattern analysis.

**Web Crawlers.** **Web crawlers** are programs that visit Web pages and create copies of all the visited pages so they can be processed by a search engine for indexing the downloaded pages to provide fast searches. Another use of crawlers is to automatically check and maintain the Websites. For example, the HTML code and the links in a Website can be checked and validated by the crawler. Another unfortunate use of crawlers is to collect e-mail addresses from Web pages, so they can be used for spam e-mails later.

## 27.8 Trends in Information Retrieval

In this section we review a few concepts that are being considered in more recent research work in information retrieval.

### 27.8.1 Faceted Search

Faceted Search is a technique that allows for integrated search and navigation experience by allowing users to explore by filtering available information. This search technique is used often in ecommerce Websites and applications enabling users to navigate a multi-dimensional information space. Facets are generally used for handling three or more dimensions of classification. This allows the **faceted classification scheme** to classify an object in various ways based on different taxonomical criteria. For example, a Web page may be classified in various ways: by content (airlines, music, news, ...); by use (sales, information, registration, ...); by location; by language used (HTML, XML, ...) and in other ways or facets. Hence, the object can be classified in multiple ways based on multiple taxonomies.

A **facet** defines properties or characteristics of a class of objects. The properties should be mutually exclusive and exhaustive. For example, a collection of art objects might be classified using an artist facet (name of artist), an era facet (when the art

was created), a type facet (painting, sculpture, mural, ...), a country of origin facet, a media facet (oil, watercolor, stone, metal, mixed media, ...), a collection facet (where the art resides), and so on.

Faceted search uses faceted classification that enables a user to navigate information along multiple paths corresponding to different orderings of the facets. This contrasts with traditional taxonomies in which the hierarchy of categories is fixed and unchanging. University of California, Berkeley's Flamenco project<sup>28</sup> is one of the earlier examples of a faceted search system.

## 27.8.2 Social Search

The traditional view of Web navigation and browsing assumes that a single user is searching for information. This view contrasts with previous research by library scientists who studied users' information seeking habits. This research demonstrated that additional individuals may be valuable information resources during information search by a single user. More recently, research indicates that there is often direct user cooperation during Web-based information search. Some studies report that significant segments of the user population are engaged in explicit collaboration on joint search tasks on the Web. Active collaboration by multiple parties also occur in certain cases (for example, enterprise settings); at other times, and perhaps for a majority of searches, users often interact with others remotely, asynchronously, and even involuntarily and implicitly.

Socially enabled online information search (social search) is a new phenomenon facilitated by recent Web technologies. **Collaborative social search** involves different ways for active involvement in search-related activities such as co-located search, remote collaboration on search tasks, use of social network for search, use of expertise networks, involving social data mining or collective intelligence to improve the search process and even social interactions to facilitate information seeking and sense making. This social search activity may be done synchronously, asynchronously, co-located or in remote shared workspaces. Social psychologists have experimentally validated that the act of social discussions has facilitated cognitive performance. People in social groups can provide solutions (answers to questions), pointers to databases or to other people (meta-knowledge), validation and legitimization of ideas, and can serve as memory aids and help with problem reformulation. **Guided participation** is a process in which people co-construct knowledge in concert with peers in their community. Information seeking is mostly a solitary activity on the Web today. Some recent work on collaborative search reports several interesting findings and the potential of this technology for better information access.

## 27.8.3 Conversational Search

**Conversational Search** (CS) is an interactive and collaborative information finding interaction. The participants engage in a conversation and perform a social search activity that is aided by intelligent agents. The collaborative search activity helps the

<sup>28</sup>Yee (2003) describes faceted metadata for image search.

agent learn about conversations with interactions and feedback from participants. It uses the semantic retrieval model with natural language understanding to provide the users with faster and relevant search results. It moves search from being a solitary activity to being a more participatory activity for the user. The search agent performs multiple tasks of finding relevant information and connecting the users together; participants provide feedback to the agent during the conversations that allows the agent to perform better.

## 27.9 Summary

In this chapter we covered an important area called information retrieval (IR) that is closely related to databases. With the advent of the Web, unstructured data with text, images, audio, and video is proliferating at phenomenal rates. While database management systems have a very good handle on structured data, the unstructured data containing a variety of data types is being stored mainly on ad hoc information repositories on the Web that are available for consumption primarily via IR systems. Google, Yahoo, and similar search engines are IR systems that make the advances in this field readily available for the average end-user, giving them a richer search experience with continuous improvement.

We started by defining the basic terminology of IR, presented the query and browsing modes of interaction in IR systems, and provided a comparison of the IR and database technologies. We presented schematics of the IR process at a detailed and an overview level, and then discussed digital libraries, which are repositories of targeted content on the Web for academic institutions as well as professional communities, and gave a brief history of IR.

We presented the various retrieval models including Boolean, vector space, probabilistic, and semantic models. They allow for a measurement of whether a document is relevant to a user query and provide similarity measurement heuristics. We then discussed various evaluation metrics such as recall and precision and F-score to measure the goodness of the results of IR queries. Then we presented different types of queries—besides keyword-based queries, which dominate, there are other types including Boolean, phrase, proximity, natural language, and others for which explicit support needs to be provided by the retrieval model. Text preprocessing is important in IR systems, and various activities like stopword removal, stemming, and the use of thesauruses were discussed. We then discussed the construction and use of inverted indexes, which are at the core of IR systems and contribute to factors involving search efficiency. Relevance feedback was briefly addressed—it is important to modify and improve the retrieval of pertinent information for the user through his interaction and engagement in the search process.

We did a somewhat detailed introduction to analysis of the Web as it relates to information retrieval. We divided this treatment into the analysis of content, structure, and usage of the Web. Web search was discussed, including an analysis of the Web link structure, followed by an introduction to algorithms for ranking the results from a Web search such as PageRank and HITS. Finally, we briefly discussed

current trends, including faceted search, social search, and conversational search. This is an introductory treatment of a vast field and the reader is referred to specialized textbooks on information retrieval and search engines.

## Review Questions

- 27.1. What is structured data and unstructured data? Give an example of each from your experience with data that you may have used.
- 27.2. Give a general definition of information retrieval (IR). What does information retrieval involve when we consider information on the Web?
- 27.3. Discuss the types of data and the types of users in today's information retrieval systems.
- 27.4. What is meant by navigational, informational, and transformational search?
- 27.5. What are the two main modes of interaction with an IR system? Describe with examples.
- 27.6. Explain the main differences between database and IR systems mentioned in Table 27.1.
- 27.7. Describe the main components of the IR system as shown in Figure 27.1.
- 27.8. What are digital libraries? What types of data are typically found in them?
- 27.9. Name some digital libraries that you have accessed. What do they contain and how far back does the data go?
- 27.10. Give a brief history of IR and mention the landmark developments.
- 27.11. What is the Boolean model of IR? What are its limitations?
- 27.12. What is the vector space model of IR? How does a vector get constructed to represent a document?
- 27.13. Define the TF-IDF scheme of determining the weight of a keyword in a document. What is the necessity of including IDF in the weight of a term?
- 27.14. What are probabilistic and semantic models of IR?
- 27.15. Define *recall* and *precision* in IR systems.
- 27.16. Give the definition of *precision* and *recall* in a ranked list of results at position  $i$ .
- 27.17. How is F-score defined as a metric of information retrieval? In what way does it account for both precision and recall?
- 27.18. What are the different types of queries in an IR system? Describe each with an example.
- 27.19. What are the approaches to processing phrase and proximity queries?

- 27.20. Describe the detailed IR process shown in Figure 27.2.
- 27.21. What is stopword removal and stemming? Why are these processes necessary for better information retrieval?
- 27.22. What is a thesaurus? How is it beneficial to IR?
- 27.23. What is information extraction? What are the different types of information extraction from structured text?
- 27.24. What are vocabularies in IR systems? What role do they play in the indexing of documents?
- 27.25. Take five documents with about three sentences each with some related content. Construct an inverted index of all important stems (keywords) from these documents.
- 27.26. Describe the process of constructing the result of a search request using an inverted index.
- 27.27. Define *relevance feedback*.
- 27.28. Describe the three types of Web analyses discussed in this chapter.
- 27.29. List the important tasks mentioned that are involved in analyzing Web content. Describe each in a couple of sentences.
- 27.30. What are the three categories of agent-based Web content analyses mentioned in this chapter?
- 27.31. What is the database-based approach to analyzing Web content? What are Web query systems?
- 27.32. What algorithms are popular in ranking or determining the importance of Web pages? Which algorithm was proposed by the founders of Google?
- 27.33. What is the basic idea behind the PageRank algorithm?
- 27.34. What are hubs and authority pages? How does the HITS algorithm use these concepts?
- 27.35. What can you learn from Web usage analysis? What data does it generate?
- 27.36. What mining operations are commonly performed on Web usage data? Give an example of each.
- 27.37. What are the applications of Web usage mining?
- 27.38. What is search relevance? How is it determined?
- 27.39. Define *faceted search*. Make up a set of facets for a database containing all types of buildings. For example, two facets could be “building value or price” and “building type (residential, office, warehouse, factory, and so on)”.
- 27.40. What is social search? What does collaborative social search involve?
- 27.41. Define and explain *conversational search*.



## Selected Bibliography

Information retrieval and search technologies are active areas of research and development in industry and academia. There are many IR textbooks that provide detailed discussion on the materials that we have briefly introduced in this chapter. A recent book entitled *Search Engines: Information Retrieval in Practice* by Croft, Metzler, and Strohman (2009) gives a practical overview of search engine concepts and principles. *Introduction to Information Retrieval* by Manning, Raghavan, and Schütze (2008) is an authoritative book on information retrieval. Another introductory textbook in IR is *Modern Information Retrieval* by Ricardo Baeza-Yates and Berthier Ribeiro-Neto (1999), which provides detailed coverage of various aspects of IR technology. Gerald Salton's (1968) and van Rijsbergen's (1979) classic books on information retrieval provide excellent descriptions of the foundational research done in the IR field until the late 1960s. Salton also introduced the vector space model as a model of IR. Manning and Schütze (1999) provide a good summary of natural language technologies and text preprocessing. "Interactive Information Retrieval in Digital Environments" by Xie (2008) provides a good human-centered approach to information retrieval. The book *Managing Gigabytes* by Witten, Moffat, and Bell (1999) provides detailed discussions for indexing techniques. The TREC book by Voorhees and Harman (2005) provides a description of test collection and evaluation procedures in the context of TREC competitions.

Broder (2002) classifies Web queries into three distinct classes—navigational, informational, and transactional—and presents a detailed taxonomy of Web search. Covi and Kling (1996) give a broad definition for digital libraries in their paper and discuss organizational dimensions of effective digital library use. Luhn (1957) did some seminal work in IR at IBM in the 1950s on autoindexing and business intelligence that received a lot of attention at that time. The SMART system (Salton et al. (1993)), developed at Cornell, was one of the earliest advanced IR systems that used fully automatic term indexing, hierarchical clustering, and document ranking by degree of similarity to the query. The SMART system represented documents and queries as weighted term vectors according to the vector space model. Porter (1980) is credited with the weak and strong stemming algorithms that have become standards. Robertson (1997) developed a sophisticated weighting scheme in the City University of London Okapi system that became very popular in TREC competitions. Lenat (1995) started the Cyc project in the 1980s for incorporating formal logic and knowledge bases in information processing systems. Efforts toward creating the WordNet thesaurus continued in the 1990s, and are still ongoing. WordNet concepts and principles are described in the book by Fellbaum (1998). Rocchio (1971) describes the relevance feedback algorithm, which is described in Salton's (1971) book on *The SMART Retrieval System—Experiments in Automatic Document Processing*.

Abiteboul, Buneman, and Suciú (1999) provide an extensive discussion of data on the Web in their book that emphasizes semistructured data. Atzeni and Mendelzon (2000) wrote an editorial in the VLDB journal on databases and the Web. Atzeni et al. (2002) propose models and transformations for Web-based data. Abiteboul et al. (1997) propose the Lord query language for managing semistructured data.

Chakrabarti (2002) is an excellent book on knowledge discovery from the Web. The book by Liu (2006) consists of several parts, each providing a comprehensive overview of the concepts involved with Web data analysis and its applications. Excellent survey articles on Web analysis include Kosala and Blockeel (2000) and Liu et al. (2004). Etzioni (1996) provides a good starting point for understanding Web mining and describes the tasks and issues related with the World Wide Web. An excellent overview of the research issues, techniques, and development efforts associated with Web content and usage analysis is presented by Cooley et al. (1997). Cooley (2003) focuses on mining Web usage patterns through the use of Web structure. Spiliopoulou (2000) describes Web usage analysis in detail. Web mining based on page structure is described in Madria et al. (1999) and Chakraborti et al. (1999). Algorithms to compute the rank of a Web page are given by Page et al. (1999), who describe the famous PageRank algorithm, and Kleinberg (1998), who presents the HITS algorithm.