Abstract

The advent of the Web in the mid-1990s followed by its fast adoption in a relatively short time, posed significant challenges to classical information retrieval methods developed in the 1970s and the 1980s. The major challenges include that the Web is massive, dynamic, and distributed. The two main types of tasks that are carried on the Web are searching and mining. Searching is locating information given an information need, and mining is extracting information and/or knowledge from a corpus. The metrics for success when carrying these tasks on the Web include precision, recall (completeness), freshness, and efficiency.

INTRODUCTION

Information retrieval is the area of computer science concerned with the representation, storage, organization, and access to documents.

Documents, in this definition, are understood in a broad sense, and include Web pages and other contents available on the Web. The Web is a unique medium for information dissemination, characterized by low entry barriers, low publishing costs, high communication speeds, and a vast distribution network.

Most methods for information retrieval were developed in the 1970s and 1980s for relatively small and coherent collections, such as the ones found in traditional libraries. The Web poses significant challenges to these methods, being massive, dynamic, and distributed.[1]

Web information retrieval (Web IR) or Web search, differs significantly from traditional information retrieval. The two main differences are the scale and nature of the collections being processed. Web search includes topics such as Web crawling, indexing and querying, adversarial Web IR issues, and Web distributed systems and evaluation metrics. Another relevant topic is Web data mining, which includes the analysis of the content, structure, and usage of the Web. In the following, we focus on these two topics: Web search and Web data mining. Our coverage of details and bibliography is by no means complete, and the interested reader is referred to Baeza-Yates and Ribeiro-Neto[2] and Chakrabarti.[3]

WEB SEARCH

Web search is the main application of Web IR, and a very successful one. From the user’s point of view, a short query consisting of a few keywords is written in a search box, and the search engine displays in return a short list, typically of 10–20 Web pages that are considered relevant to the query issued and expected to be high-quality documents.

The two main goals in a search are precision and recall, and they are, to a certain extent, competing goals.

Precision is defined as the fraction of relevant results contained in the result set, or in a part of the result set. For instance, if 3 out of 10 results for a query are relevant, the precision is 30%.

Recall is defined as the fraction of relevant results in a set, compared with the total number of pages on the Web that would be relevant for this query. Of course, the total number of pages on the Web relevant for a particular query is an unknown quantity, but for popular query terms it can be estimated using sampling techniques.

An information retrieval system can have high recall at the expense of precision, simply by returning more results, and high precision at the expense of recall, by removing results for which the algorithm is unsure about their relevance. The design of effective algorithms for search seeks a balance among these two extremes, and in the Web the focus is on precision as recall cannot be measured, only estimated.

In the case of Web search there is a third goal that is freshness. The Web changes continuously and the copy of the Web that the search engine has can become stale very quickly. The three goals: precision, recall, and freshness are sometimes mutually exclusive and introduce three-way trade-offs,[4] as depicted in Fig. 1. These trade-offs create the possibility of several niche markets apart from general Web search, including: vertical search, over a particular subset of pages; archive search, over several snapshots of
the Web; and news search, over Web sites that change with very high frequency.

An additional consideration in search engine design is efficiency. Large Web search engines have to deal with a large volume of queries and search huge data collections, so even large amounts of computational resources can be insufficient. Successful algorithms for Web search avoid consuming too many resources per query or per document.

From the point of view of the search engine, Web search occurs in two main phases. The first phase is offline, with a certain periodicity or by permanent incremental maintenance. It includes crawling the Web to download pages and then indexing them to provide fast searches. The second phase is done online, and corresponds to the process of querying and ranking, which consists of building a ranked list of results using the index for a particular query. These phases are depicted in Fig. 2 and explained in more detail in the rest of this section.

**Web Crawling**

A Web crawler is a system that automatically downloads pages from the Web following a set of predefined rules. A Web crawler receives as input a starting set of URLs that constitutes a “seed set,” and a set of rules to follow. The crawler first downloads the pages from the seed set, extracts the links found in such pages, and then follows those links recursively while certain criteria are met.

Crawling the Web is a required step for many Web IR applications.

Aside from Web search, Web crawlers are multipurpose systems that can be used for a variety of tasks, including finding and reporting “broken links” or other coding errors, and computing statistics about the Web.

The most important design constraint of Web crawlers is that they must avoid disrupting the Web servers they interact with. While downloading Web pages, the crawler is using the resources of others, and thus it must keep its resource consumption as low as possible. Web crawler designers and operators must take every possible step to control the frequency of visits to sites and keep them to a minimum. Also, the authors of Web sites have to ultimately decide which part, if any, of their sites can be visited by crawlers. This is done by using the robots exclusion protocol.[5]

After downloading the pages, they have to be processed to be used by the search engine or other application. HTML is the main language for coding documents on the Web, but there are many other formats present, including PDF, plain text, plus the document formats used by popular text-processing software such as Microsoft Word or OpenOffice. These formats have to be converted to a single representation before they can be used.

The importance of freshness is another aspect of the crawler’s operation. The Web is very dynamic, and it changes continuously; this means that by the time the crawler has finished collecting a set of pages, many of the pages it has downloaded have already changed.[6] Crawling the Web, to a certain extent, resembles watching the sky at night:[2] the light we see from the stars has often taken thousands of years to reach our eyes. Moreover, the light from different stars has taken different amounts of time, so what we see is not a snapshot of the sky at any given moment, present or past. It is a combination of images from different times. The same happens with the collection of Web pages crawled by a search engine.

The Web pages that are not directly accessible by following links, but require the user to enter a query in an online form (e.g., enter an author’s name to retrieve bibliographic data), constitute the “hidden Web.”[47] Searching this content is challenging for search engines. In most cases, large information providers generate “crawler-friendly” pages for better indexing by search engines, but other forms of collaboration may arise in the future, including exposing an interface for querying the local database to the search engine.

**Indexing**

After collecting pages, the next step is to create an index to enable fast searches over the downloaded
Fig. 3 Example of an inverted index.

The most used logical view for this task is the “bag of words” model, in which each document is represented as a multiset containing all its keywords, disregarding the order in which they appear.

Tokenization is the process by which a text is separated into words. This is trivial in Western languages, but harder to do in other languages such as Chinese.

Stopwords are functional words that do not convey meaning by themselves, such as articles and prepositions. The removal of stopwords reduces the amount of data processing and the size of the index, and also improves the retrieval accuracy of information retrieval systems.

Stemming is the extraction of the morphological root of a word. This allows us to search for “housing” and retrieve results that include “house” or “houses.”

After the text normalization operations have been applied, most search systems build an index, a data structure designing to accelerate the process of retrieving documents containing a given query. The most prevalent type of such structure is an inverted index. In Fig. 3, an example of an inverted index for a collection of five toy documents (each of them having two words) is shown.

An inverted index is composed of two parts: a vocabulary, containing all the terms in the collection, and a posting list, which contains references to the documents (s) in which each word of the vocabulary appears. An inverted index is a powerful tool for the search engine, enabling very fast response times.

In the example of Fig. 3, if we search for “global AND climate” in the inverted index, the task is basically to intersect the set of pages containing “global” \{1, 2, 5\} with the set of pages containing “climate” \{2, 3, 4\}, obtaining as a result the set \{2\}. If these lists are sorted, their intersection can be computed very quickly. This is how a basic inverted index works. There are many techniques for providing faster search or reducing the space occupied by the index. For example, if phrase or proximity search is needed, the exact positions where the term appears in a document must be also encoded in the posting list. The interested reader is referred to Baeza-Yates and Ribeiro-Neto for an overview of indexing techniques.

Another aspect is that large search engines achieve high response times by means of parallelization. In this case, the index has to be divided in some way, and each piece of the index has to be given to a different physical computer. There are two main strategies for this partitioning. One is to give each machine a set of documents, the other one is to give each machine a set of terms.

### Querying and Ranking

Most search engines receive queries expressed as a set of keywords. Scalable question answering systems, in which users express their information need by means of a question, have remained elusive to researchers in particular because many natural language processing algorithms still require a prohibitive amount of computational power for Web-scale collections.

Typical queries are very short, between two and three keywords each. After receiving a query, the search engine uses its inverted index (or indexes) to build a page with results that is shown to users. To a certain extent, the problem of finding a set of pages that are related to the query is the “easy” part, given that for most broad queries there are thousands or millions of documents that are potentially appropriate. The most difficult challenge is to find among those documents, a small subset of the best 10 or 20. This is the problem of ranking.

Ranking has two main aspects: relevance and quality. The dimension of relevance indicates how related is the retrieved document to the user intention. The dimension of quality indicates how good is the document by itself. Search engines try to produce results for a given query that are both relevant for the query and have high quality. One of the main techniques to do fast ranking is to use partial evaluation techniques, such that only the top ranked answers are computed, and the rest of the answer is computed incrementally as the user demands it.

### Relevance

Given that the search engine cannot understand the meaning of the queries nor of the documents, it must resort to statistical methods to compare queries to documents. These statistical methods allow the search engine
to provide an estimation on how similar the query is to each document retrieved, which is used as an approximation of how relevant is the document for the query.

The vector space model\[8\] is the most used framework for measuring text similarity. It represents each document as a vector in a high-dimensional space, in which each dimension is a term, and the magnitude of each component of the vector is proportional to the frequency of the corresponding term, and inversely proportional to the document frequency of the term in the collection.

Differences in document size have to be taken into account for the similarity measure between documents, so the angle between documents is used instead of, for instance, the Euclidean distance between them. For instance, the angle between the documents “global warming” and “warming warming global global” is zero (so the documents are equivalent according to this metric), the angle between the documents “global warming” and “global climate” is 45° (under a simple weighting scheme), and the angle between the documents “global warming” and “climate change” is 90°. For normalization purposes, the cosine of such angle is the standard way of expressing this similarity metric.

Information retrieval systems usually do not apply the vector space model naively, as it has significant weaknesses. By itself the vector space model does not take relationships among terms into account. For instance, strictly speaking the cosine similarity between the “global warming” and “climate change” is zero, and the cosine similarity between “global warming” and “strawberry ice cream” is also zero; but clearly the first pair of concepts have a closer relationship than the second pair. Two methods that can be applied to overcome this problem are query expansion and latent semantic indexing.

Query expansion consists in adding related words to the queries, and the same technique can be applied to documents. For instance, this could convert automatically “global warming” into “global world warming climate” and “climate change” into “climate warm cold change global.” The specific words that are added can be obtained from different sources, including co-occurrence in the collection. In the case of the Web, there are rich sources of information to obtain words related to a document. The main one is anchor text, that is, the text contained in the links pointing to the present document. This is a very important feature in the ranking computed by most modern search engines. A second source of information is social book-marking sites that allow users to associate tags to documents.

Latent semantic indexing\[11\] consists in projecting the vectors representing queries (and documents) into a different, and usually smaller, space. This technique is based on principal component analysis and attempts to group automatically terms into the main “concepts” representing multiple weighted terms.

Quality

Search engines are designed to extract a set of features from the documents they index, and use those features to assert what is the quality of a given document. Quality is hard to define and of course hard to estimate using statistical measures. However, certain textual features from documents, including content length, frequencies of some words, features about the paragraphs, etc. tend to be correlated with human assessments about document quality.\[12\]

Apart from the content of the pages themselves, on the Web, a rich source of information for inferring quality can be extracted from links. Links on the Web tend to connect topically related pages,\[13\] and they often imply that the target document has an acceptable or high level of quality. Thus, they can be used for finding high-quality items in the same way as academic citations can partially characterize the importance of a paper. The same considerations as for academic citations apply: not all of the links imply endorsement;\[14\] some pages attract many citations for other reasons aside from quality, and citation counts can be inflated by self-citations or citations that point to errors; among other problems.

There are two classic link analysis algorithms to obtain quality metrics for Web pages: PageRank and HITS. For a survey of their variants, and other methods, see Borodin et al.\[15\]

The PageRank algorithm\[16\] defines the importance of a page in a recursive manner: “a page with high PageRank is a page referenced by many pages with high PageRank.” Despite the definition being recursive, it is possible to compute PageRank scores using results from Markov chain theory. In brief, the wanderings of a “random surfer” are simulated, in which a person browses the Web by following links at random. The PageRank score of a page is roughly proportional to the amount of expected visits the random surfer will do to each page.

The hyperlink-induced topic search (HITS) algorithm\[17\] is another method for ranking Web pages. It starts by building a set of pages related to a topic by querying a search engine, and then expands this set by using incoming and outgoing links, by crawling the Web or by querying a search engine again. Next, two scores for each page are computed: a hub score and an authority score.

As shown in Fig. 4, a page with a high hub score is a page that links to many pages with a high authority score. A page with a high authority score is a page linked by many pages with a high hub score. Again,
despite the apparent circularity of the definition, both hub and authority scores can be computed efficiently by an iterative computation.

Another source of information for ranking pages on the Web is usage data. A page that is visited frequently and/or for long periods by users may be more interesting than a page that is not. This information can be obtained by the search engine by providing a client-side add-on such as a toolbar, or by instrumenting the search engine result pages to capture click information.

**Ranking manipulation**

Visits from search engines are an important source of traffic for many Web sites. Given that in the case of commercial ventures on the Web, traffic is strongly correlated with sales volume, there is a significant economic incentive for obtaining high rankings on search engines. These incentives may lead Web page authors to use deceptive techniques for achieving high rankings. These deceptive techniques are known on a whole as search engine spam.

There are many types of search engine spam: inserting many keywords on Web pages, linking nepotistically among pages, providing different contents to the search engine than to users (also called “cloaking”), among others; for a survey of these methods, Gyöngyi.[19]

Search engine spam has been an important issue for search engines for a number of years, and it is not likely to be solved in the near future. Web spam damages search engines’ reputation as it exploits and weakens the trust relationship between users and search engines.[19] Spamming has become so prevalent that without countermeasures to identify and remove spam, the quality of search engine results would be very poor.[20]

**WEB MINING**

Web mining is the application of data mining techniques to find patterns on data downloaded from the Web. Based on the main source of data they use, these techniques can be broadly classified as Web content mining, Web link mining, and Web usage mining.

**Content Mining**

Web content mining is the extraction of knowledge from the textual content of Web pages. The main challenge here is that HTML, while designed initially to be a language for logical formatting, is actually used as a language for physical formatting. Logical formatting describes document structure, such as paragraphs and headings, while physical formatting describes visual attributes like font sizes, colors, and spacings. With logical formatting, it would be easier to extract information than with the present physical formatting.

In general, the Web sites that are rich in information are built using “dynamic pages” that are generated on demand, in response to a user click or query. These pages are created by querying a local database, formatting the results as HTML, and then displaying such results to the user.

For example, let us consider a Web site about movies being shown in theaters. This Web site may present the movies on a tabular form with the titles, ratings, and show times, for instance. A Web search engine or other information provider interested on doing Web information extraction must read this table and reconstruct the original schema based on it. For example, it must find out that the first column contains the movie title, the second column the rating, and the third column the show times. This is easy for a human but it is hard to do it automatically. Most of the time, some information is lost, as depicted in Fig. 5.
Information extraction systems use clues from the page’s formatting and structure, domain knowledge, and training examples, among other sources of information, to map HTML fragments to tuples in relations. They can also use methods for detecting the page template and isolating navigational areas that do not contribute content. The systems that do this task are informally known as “content scrapers” and they can be quite accurate, especially when restricted to particular domains. For a survey of information extraction methods, Kayed and Shaalan.\[21]\n
Other aspects of content mining besides information extraction are content classification, sentiment analysis, and duplicated pages detection.

Content classification in general looks at statistics obtained from the Web pages to classify their contents. In many cases, this is done to find out what is the topic the contents are about. In other cases, content classification is used to extract document properties such as the genre of the document, or whether it expresses more opinions or more facts, or to evaluate how well-written a document is. In all cases, a statistical description of the document is created, and then a machine learning algorithm takes that description and a set of training labels to construct a model able to separate automatically the classes.\[22,23]\n
Sentiment analysis, including “intention mining,” is the task of finding what is the sentiment or intention of the author of a document. Specifically, it can be used to determine if a certain fragment is expressing a negative or positive opinion. This is very important given the large amount of product and service reviews available on Web pages, blogs, or forums. These reviews are typically very short, usually no more than a few paragraphs. The techniques of sentiment classification include the analysis of the frequency of certain terms\[24]\ with the aid of parts-of-speech taggers or other natural language processing tools.

Finally, there is a significant amount of duplicate content on the Web. According to Broder et al.,\[25]\ roughly one-third of the pages on the Web are duplicates or near duplicates of another page, and present studies have confirmed this trend. Finding near-duplicate content\[26]\ is important for efficiency reasons, to avoid downloading and indexing many times the same pages. It is also important to filter out plagiarism, so that the original page gets ranked high, and not the copies.

**Link Mining**

The overall structure of the Web differs significantly from the one exhibited by random networks. The most salient difference is that, while on a random network most of the nodes have a degree (number of connections) close to the average, in networks such as the Web, the distribution of the degree is very skewed. The networks that have this property are called scale-free networks.

Fig. 6 depicts a random network and a scale-free network with the same number of nodes and edges. In a scale-free network, a few nodes attract more of the in-links. This can be explained by “rich-get-richer” processes\[27]\ in which having many links gives a better chance of attracting new links, increasing the disparity in the number of connections over time.

At a macroscopic level, looking at the properties of the network as a whole, we can describe the Web in terms of the strongly connected components on it. A strongly connected component is a part of a graph in which all pairs of nodes can reach each other (in both directions) by following links. The Web exhibits a very large strongly connected component (CORE), other components that are reachable to/from it by directed links (IN and OUT, respectively), and pages that cannot be reached at all from the CORE, which are called ISLANDS. Minor components such as TENDRILS and TUNNEL can also be identified. This description is called the bow-tie structure of the Web\[28]\ given its shape, depicted in Fig. 7.

PageRank and HITS could be considered simple link mining techniques. More elaborated link analysis can be used for finding similar pages, communities, or detection of Web spam based on links.
Usage Mining

Usage data on the Web is abundant and valuable. Web site administrators can capture usage data by enabling logging on their Web servers, and they can enrich such data by instrumenting their internal links. There are several free software packages available that can do sophisticated analysis of access logs and can discover, for instance, typical browsing paths. This is of particular importance for retailers and other e-commerce Web sites that can use this information to drive the design of their Web sites, improving the user experience and/or increasing their sales volume.

Search engines have access to the queries written by the users, and the pages they selected after seeing the list of results (and the pages they did not select). Data from user search sessions can be used to increase the relevance of the results. Interesting relationships can be inferred by looking at users, queries, and pages. We can observe, for instance, that similar users tend to issue similar queries, that similar pages show up as results for related queries, and so on.

Usage data is increasingly valuable for search engines. Privacy issues arise in the confluence of the legal and technical aspects associated to this data collection, and both users and search engine have incentives for maintaining and enforcing the secrecy of this data.

CONCLUSIONS AND PRESENT TRENDS

As we have seen, Web retrieval methods differ from standard information retrieval methods, and can adapt to the large-scale, open, and distributed nature of the Web. For the future, two topics that are attracting a significant research effort are the mobile Web and the semantic Web.

The Mobile Web is the Web that is accessible and used through portable devices. Today, the capabilities of most mobile cell phones are well beyond just making phone calls. Many include Web-browsing software, and a growing fraction of the activity on the Web is carried through these devices, including browsing, searching, and even producing content (e.g., in the case of cell phones equipped with cameras). A challenge here is to provide users of portable devices with an experience that takes into account their geographical location and their present activity.

The Semantic Web is a vision of the future of the Web, in which the Web contents can be read and understood by both humans and software agents. This will enable information integration and sharing without losing information. Several technologies enable the semantic Web, ranging from simple markup languages as the Extensible Markup Language to other languages that describe relationships among objects, classes, and properties. On top of these layers, applications will be able to analyze and, later, to reason about the contents and to extract knowledge from them.

REFERENCES


BIBLIOGRAPHY
