5
Web Document Modeling

Alessandro Micarelli, Filippo Sciarrone, and Mauro Marinilli

Department of Computer Science and Automation
Artificial Intelligence Laboratory
Roma Tre University
Via della Vasca Navale, 79 - 00146 Rome, Italy
{micarel, sciarro, marinil}@dia.uniroma3.it

Abstract. A very common issue of adaptive Web-Based systems is the modeling of documents. Such documents represent domain-specific information for a number of purposes. Application areas such as Information Search, Focused Crawling and Content Adaptation (among many others) benefit from several techniques and approaches to model documents effectively. For example, a document usually needs preliminary processing in order to obtain the relevant information in an effective and useful format, so as to be automatically processed by the system. The objective of this chapter is to support other chapters, providing a basic overview of the most common and useful techniques and approaches related with document modeling. This chapter describes high-level techniques to model Web documents, such as the Vector Space Model and a number of AI approaches, such as Semantic Networks, Neural Networks and Bayesian Networks. This chapter is not meant to act as a substitute of more comprehensive discussions about the topics presented. Rather, it provides a brief and informal introduction to the main concepts of document modeling, also focusing on the systems that are presented in the rest of the book as concrete examples of the related concepts.

5.1 Introduction

The Web document, in its various representation forms, is the focal point of this chapter, which aims at illustrating the most common techniques employed in Web Document Modeling[1] literature, with particular attention given to those used in Adaptive Web-Based systems. The purpose of this chapter is to offer a support for the other chapters in this volume dealing with the various adaptive systems in greater detail.[2] The present chapter is therefore not to be considered a comprehensive guide to the discussed topics, nor a substitute for more specialized literature. Readers are encouraged to consult the provided references in order to broaden their grasp of the discussed topics or equivalent literature.

[1] In this context, with the term *modeling* we mean the construction of an abstract representation of the document, useful for all applications aimed at processing information automatically.

[2] Fig. 5.1 shows the structure of this chapter together with the links to the other chapters of this book.

It is well known that, owing to the Internet, each one of us can benefit from a large quantity of information, available on-line in several, essentially standard, formats, such as HTML, XML and XHTML text pages and *jpeg* and *tif* graphic ones. More complex formats, particularly multimedia (audio and/or video) usually require a longer search and interaction with the Web, as well as more sophisticated fruition tools installed on user clients, now widely accessible.

The quantity of information available on the World Wide Web is increasing exponentially, and this boom has paved the way for a new era, creating new opportunities in many different fields, such as e-business, e-commerce, e-marketing, e-finance and e-learning, just to mention some of the most interesting ones. *Web Intelligence* [90, 91] and *Wisdom Web* [93, 48, 92] are other examples of new disciplines born from the development of the World Wide Web. All this proceeds from the birth of HTTP, *HyperText Transfer Protocol* [80, 39] and of HTML, *HyperText Markup Language*, a subset of SGML, *Standard Generalized Markup Language* [29], which made the Internet enjoyable for anyone who had a computer with a browser on-board.[3] With time, the number of Web surfers increased, and so did the available Web documents, especially HTML pages. This eventually required the need to gather the information to be supplied in more structured containers, enabling more thorough, personalized searches, directly correlated to the semantics of the document itself, for a more intelligent fruition and to provide more sophisticated search-tools systems.

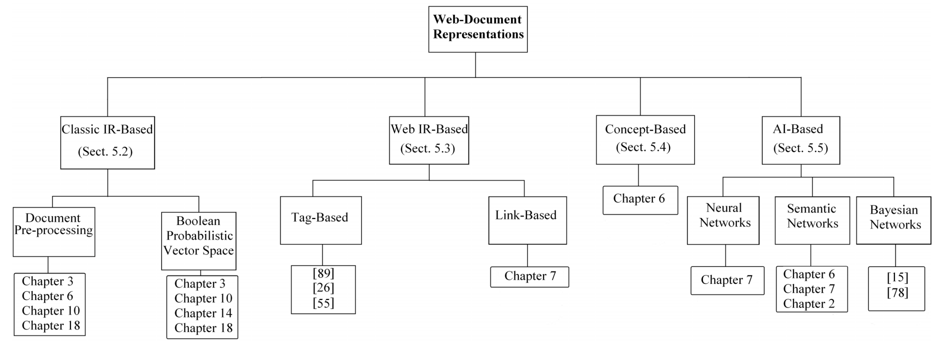

Web-Document Representations
Classic IR-Based (Sect. 5.2)
Web IR-Based (Sect. 5.3)
Concept-Based (Sect. 5.4)
AI-Based (Sect. 5.5)
Document Pre-processing
Boolean Probabilistic Vector Space
Tag-Based
Link-Based
Chapter 6
Neural Networks
Semantic Networks
Bayesian Networks
Chapter 3 Chapter 6 Chapter 10 Chapter 18
Chapter 3 Chapter 10 Chapter 14 Chapter 18
[89] [26] [55]
Chapter 7
Chapter 7
Chapter 6 Chapter 7 Chapter 2
[15] [78]

Fig. 5.1. The structure of this chapter with the references to other chapters of this volume

All the aforementioned techniques are the ones used concretely to represent documents on various Web sites, and may therefore be called *Layout Representations*. For automatic systems that operate on the Web, however, e.g. adaptive Web-Based systems, such documents often need to be *pre-processed*, that is, recasting them in representations different from primitive ones and more apt to undergo a particular elaboration. Think of a personalized search system in which the calculation of the relevance of a

[3] The reader can visit the w3c Web site: http://www.w3c.org for further reading on Web standards and protocols.

particular HTML document, as retrieved by a single user, needs to be automatically processed. Only a suitably recast representation can ensure a thorough evaluation of the document itself. Some of these representation methods are inspired by the classic Information Retrieval (IR) - an area of research conceived when the Internet still had not dawned - that enjoys a long tradition in the modeling and treatment of documents [74, 76]. Nonetheless, in literature, especially since the Web was born, other IR techniques have been proposed, capable of exploiting the hypertext features of the Web. While some proposals focused on the enhancement of classic IR systems, others searched for new retrieval strategies, exploiting the Web's hypertext features, such as page hyperlinks and HTML general tags [14, 46, 55]. The result was the advent of *Web Information Retrieval* (Web-IR), one of the new IR techniques used to retrieve documents from the Web [2, 1]. Finally, there are also modeling techniques that are based on document-concept representations and on the models and methods of Knowledge Representation, typical of Artificial Intelligence.

This chapter is therefore subdivided into four main parts: the first part (Section 5.2) describes the classic IR methods and techniques for representing documents, with particular examples based on real Web pages; the second part (Section 5.3) describes the Web-IR methods and techniques, with particular reference to the typical hypertext features of the majority of Web documents; the third part (Section 5.4) illustrates document representation methods based on concept approaches, where documents are modeled by means of concept sets. The last part (Section 5.5), deals with document modeling methods based on Artificial Intelligence techniques. More precisely, Section 5.2 illustrates the approach based on classic IR techniques to represent documents. In particular, Subsection 5.2.1 illustrates the typical characteristics of the document pre-processing techniques, including the weighting of the various terms that appear within it. Subsection 5.2.2 describes the *Boolean Model* document modeling technique, characterized by its simplicity and easiness of implementation. Subsection 5.2.3 presents the *Probabilistic Model* document modeling technique, which ranks the query-document similarity according to the probability theory. Subsection 5.2.4 deals with the *Vector Space Model*, the most common one in classic IR systems: it models a document as a point in an n-dimensional space, and calculates the document-query similarity following geometric rules. Section 5.3 illustrates the modern Web-IR techniques, based on HTML tags and hyperlinks. Subsection 5.3.1 shows three examples of *Tag-Based* document modeling techniques where an HTML document is modeled taking into account its HTML structure. Finally, in Subsection 5.3.2, we discuss two algorithms, *HITS* and *PageRank*, that model documents exploiting the hypertext structure of the Web. In Section 5.4 we discuss a concept approach to document modeling, called *Latent Semantic Indexing*, where a document is modeled by a set of concepts. Section 5.5, studies in-depth some document representation methods based on Artificial Intelligence techniques. Subsection 5.5.1 illustrates the characteristics of document modeling via Artificial Neural Networks; Subsection 5.5.2 presents a brief description of the characteristics of document modeling based on Semantic Networks ; finally, Subsection 5.5.3 illustrates how Bayesian Networks are used in diverse adaptive systems to model documents. Our conclusions are drawn in Section 5.6.

5.2 Classic IR-Based Document Representations

This section illustrates the classic IR document modeling techniques, namely the document modeling techniques born from the classic works of Salton [74], of Salton and McGill [76] and of Van Rijsbergen [67], suitable for the retrieval of relevant documents from a collection, and proposed when the Internet was only just dawning. Many adaptive systems on the Web employ such IR techniques to represent Web documents (see for example [58, 43, 78, 11, 36]), obtaining good results. Obviously, this description cannot be fully exhaustive, considering the vastness of the topic; rather, it is to be considered a starting point for possible in-depth studies quoted in the relevant bibliography.

Subsection 5.2.1 deals with *Document Pre-processing*, the process used to remove from the original document all information deemed non-relevant for the semantics of the very document, such as HTML tags and *stopwords*, thus leaving relevant terms only. In fact, in a classic IR environment, a document is simply a *bag of words*, i.e., an unstructured set of words or terms appearing in it, each one having an associated relevance weight. This approach does not take into account typical features of human language, such as homonymy and polysemy: for example, the term *mouse* is represented in the same way, both when it indicates an animal and a computer device; the terms *home* and *house* are represented as two uncorrelated terms. Nonetheless, this document modeling approach, which is uniquely based on large-scale statistics, is currently the most widespread, mainly because of its simplicity and of its suitability for the automatic processing of documents (see for example [77]).

The *Boolean Model* is illustrated in Subsection 5.2.2. This document modeling technique, which can be implemented very easily, is based on the Theory of Sets and on Boolean Algebra.

The *Probabilistic Model* is illustrated in Subsection 5.2.3. This type of document modeling follows the probability theory in calculating the relevance of a document following a given query.

Finally, the *Vector Space Model* is illustrated in Subsection 5.2.4. This technique is based on vector space for the representation of queries and documents. However, such a technique requires large-scale statistics calculated on the entire document collection from which the document to be modeled is retrieved. Many IR systems employ this technique to retrieve relevant documents through query-document similarity metrics, expressed in terms of geometric distance.

5.2.1 Document Pre-processing

A document may be published on the Web in several formats: HTML, PDF, DOC, TEXT, etc, but in all cases, the information it contains is to be *enriched* with particular character sequences, not intended for the user but for the computer visualization driver, such as the browser in the case of HTML Web pages. In order to keep the explanation as simple as possible, we shall henceforth deal only with HTML documents, without distorting conclusions whatsoever, unless exclusive details of implementation come about. As a starting example, we use the Web page illustrated in Fig. 5.2: it represents a Web page as the user sees it through his/her browser.[4] We can say this is the *user's point of*

[4] Web site: http://www.springeronline.com.

view of the document. Fig. 5.3, on the other hand, shows a fragment of the same Web page, as the browser *sees* it: we can say that this is the *browser's point of view* of the same document. They are obviously two very different viewpoints of the same thing: the user does not, and must not, see HTML formatting tags for the browser, while the browser must process everything included in the HTML Web page. In order to better comprehend this detail, crucial to get a grasp of the issue, take a look at the following HTML code lines, included in the aforesaid Web page:

```
<link rel=''stylesheet" href=''include/style\_0.css"
type=''text/css"> <script language=''JavaScript"
src=''include/item.js"></script>
<script language=''JavaScript"
src=''include/fw\_menu.js"></script>
<span class=''bodytext">

Benefit from attractive savings on Springer books
by signing up for Springer's free new book e-mail
notification service. New title info, news and
special announcements: with Springer Alerts it
pays to be informed.

</span>
```

In classic IR approach, none of the terms included in the first five lines offer significant information to the user, since they don't refer to document content, but to the layout, whereas the real information content is to be found in the following four lines, i.e., *bodytext* lines. A traditional IR system should therefore somehow extract the four significant lines for the user in order to select the correct terms to represent the Web page. This term-selection process is a part of the document pre-processing task which is broken down into four phases, explained in the following paragraphs.

1. HTML Tag Removal Phase. This phase consists in removing all HTML instructions (i.e., tags) from an HTML page. The simplest approach excludes all terms belonging to HTML tags. In the case in point, only this text would remain:

```
Benefit from attractive savings on Springer books
by signing up for Springer's free new book e-mail
notification service. New title info, news and
special announcements: with Springer Alerts it
pays to be informed.
```

2. Stopwords Removal Phase. Not all terms of a document are necessarily relevant. Some frequently used terms, within the document itself, tend to be removed: these terms are known as *Stopwords* (i.e., "a", "the", "in", "to"; or pronouns: "I", "he", "she", "it"). Stopwords obviously vary according to the language. It is possible to download an

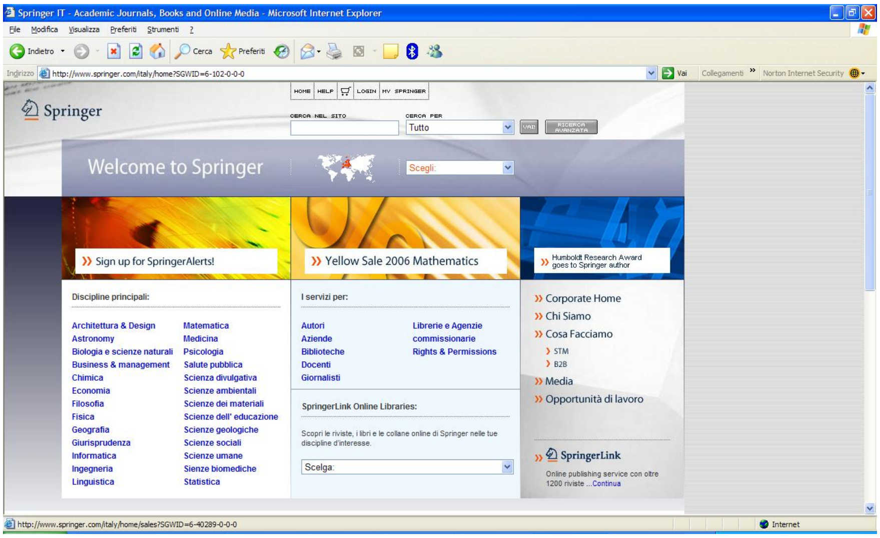
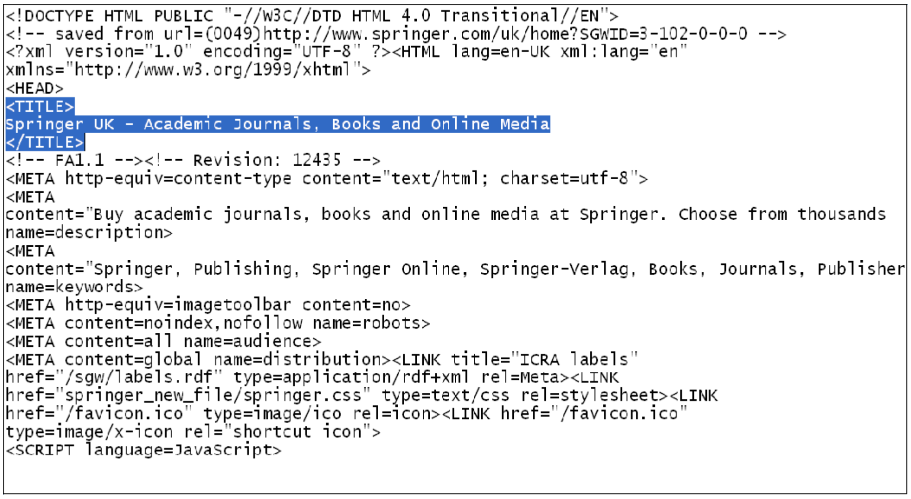

Springer IT - Academic Journals, Books and Online Media - Microsoft Internet Explorer
File Modifica Visualizza Preferiti Strumenti ?
Indietro Cerca Preferiti
Indirizzo http://www.springer.com/italy/home?SGWID=6-102-0-0-0
Vai Collegamenti Norton Internet Security
HOME HELP LOGIN MY SPRINGER
Springer
CERCA NEL SITO
CERCA PER
Tutto
VAI
RICERCA AVANZATA
Welcome to Springer
Scegli:
Sign up for SpringerAlerts!
Yellow Sale 2006 Mathematics
Humboldt Research Award goes to Springer author
Discipline principali:
Architettura & Design
Astronomy
Biologia e scienze naturali
Business & management
Chimica
Economia
Filosofia
Fisica
Geografia
Giurisprudenza
Informatica
Ingegneria
Linguistica
Matematica
Medicina
Psicologia
Salute pubblica
Scienza divulgativa
Scienze ambientali
Scienze dei materiali
Scienze dell' educazione
Scienze geologiche
Scienze sociali
Scienze umane
Sienze biomediche
Statistica
I servizi per:
Autori
Aziende
Biblioteche
Docenti
Giornalisti
Librerie e Agenzie commissionarie
Rights & Permissions
SpringerLink Online Libraries:
Scopri le riviste, i libri e le collane online di Springer nelle tue discipline d'interesse.
Scelga:
Corporate Home
Chi Siamo
Cosa Facciamo
STM
B2B
Media
Opportunità di lavoro
SpringerLink
Online publishing service con oltre 1200 riviste ...Continua
http://www.springer.com/italy/home/sales?SGWID=6-40289-0-0-0
Internet

Fig. 5.2. The HTML example page from the user point of view



```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<!-- saved from url=(0049)http://www.springer.com/uk/home?SGWID=3-102-0-0-0 -->
<?xml version="1.0" encoding="UTF-8" ?><HTML lang=en-UK xml:lang="en"
xmlns="http://www.w3.org/1999/xhtml">
<HEAD>
<TITLE>
Springer UK - Academic Journals, Books and Online Media
</TITLE>
<!-- FA1.1 --><!-- Revision: 12435 -->
<META http-equiv=content-type content="text/html; charset=utf-8">
<META
content="Buy academic journals, books and online media at Springer. Choose from thousands
name=description>
<META
content="Springer, Publishing, Springer Online, Springer-Verlag, Books, Journals, Publisher
name=keywords>
<META http-equiv=imagetoolbar content=no>
<META content=noindex,nofollow name=robots>
<META content=all name=audience>
<META content=global name=distribution><LINK title="ICRA labels"
href="/sgw/labels.rdf" type=application/rdf+xml rel=Meta><LINK
href="springer_new_file/springer.css" type=text/css rel=stylesheet><LINK
href="/favicon.ico" type=image/ico rel=icon><LINK href="/favicon.ico"
type=image/x-icon rel="shortcut icon">
<SCRIPT language=JavaScript>
```

Fig. 5.3. The HTML example page from the browser point of view

example of *stoplist*, i.e., a set of standard stopwords, of about 500 stopwords, from the *http://bll.epnet.com* Web site.[5] By exploiting this list for our text fragment, we obtain:

```
Benefit  attractive savings  Springer books  signing
Springer book e-mail notification service  title info
special announcements Springer Alerts  pays informed
```

[5] http://bll.epnet.com/help/ehost/Stop_Words.htm.

3. Stemming Phase. The goal of this phase is to reduce a term to its morphologic root, in order to recognize morphologic variations of the word itself. For example, the root *comput* is the reduced version of “comput-er”, “comput-ational”, “comput-ation” and “compute”. The morphologic analysis must be specific for every language, and can be extremely complex. The simplest stemming systems just identify and remove suffixes and prefixes. Considering its widespread use, it is worthwhile mentioning Porter’s Stemmer [65]. It is a simple procedure, which cyclically recognizes and removes known suffixes and prefixes without having to use a dictionary. This algorithm can also generate terms which are not language words, as is to be seen in the previous example: “computer”, “computational”, “computation” all become “comput”. Terms which can actually be different are unified, and the algorithm doesn’t recognize morphologic variations. The reader can find more detailed information on this algorithm on the Web site: *http://www.mozart-oz.org*.[6] After having processed our example text fragment by Porter’s stemmer,[7] we get the mapped text shown in Tab. 5.1. In Chapter 10 [59] the reader can find a relevant example of such a technique in the case of a newspaper article.

Table 5.1. The mapping between stemmed and not stemmed terms of the example text after the Porter’s stemming process

Original Terms	Stemmed Terms
benefit	benefit
attractive	attract
savings	save
springer	springer
books	book
signing	sign
springer	springer
book	book
email	email
notification	notif
service	servic
title	titl
info	info
special	special
announcements	announc
alerts	alert
pays	pai
informed	inform

4. Term Weighting Phase. By indicating our example document with d, we obtain a first representation of our original HTML document as a simple set of its terms:

[6] http://www.mozart-oz.org/mogul/doc/lager/porter-stemmer/.

[7] We submitted our text fragment to the Web site: http://maya.cs.depaul.edu/ classes/ds575/porter.html.

$d \equiv \{t_1, t_2, \ldots, t_{16}\}$, i.e., $d \equiv \{$benefit, attract, save, springer, book, sign, email, notif, servic, titl, info, special, announc, alert, pai, inform$\}$. Each term t_k of the document d belonging to a document collection D, is named *feature* or *index term* and its relevance within the document d is measured by means of an associated numeric *weight* w_k. In classic IR, weight w_k is calculated by taking into account the whole collection D of documents from which the document d has been retrieved [76, 5]: the more w_k is high the more term t_k is important for the discerning of the document d. A first simple weight, also used as a starting point to build more sophisticated weights proposed in the literature, is $w = Term\ Frequency\ TF$, namely, the number of times term t appears in the document d and indicated by $TF(t, d)$. This leads to the representation of a document d by a vector of pairs $d \equiv \{(t_1, w_1), (t_2, w_2) \ldots (t_n, w_n)\}$, with $w_k = TF(t_k, d)$. In the case in point, the representation of our example document d becomes that of Tab. 5.2.

Table 5.2. A simple representation of the example document through Term Frequency

Index Terms	**TF(t,d)**
benefit	1
attract	1
save	1
springer	3
book	2
sign	1
email	1
notif	1
servic	1
titl	1
info	1
special	1
announc	1
alert	1
pai	1
inform	1

Generally speaking, we may assert that a weight w is mostly built as a function of the frequency of term t in document d, as expressed by Eq. 5.1.

$$w = f[TF(t, d)] \tag{5.1}$$

For every document of the collection D, all terms t are extracted and the *Index Term Database* (ITD) is built: it consists of all index terms of all documents belonging to the collection D. However, not all the terms in a document d have the same relevance in discerning the document d itself for a correct representation and retrieval. Tab. 5.3 shows the generic *Term-Document Frequency Matrix*, i.e., a matrix A whose elements a_{ij} represent the weights w_{ij} of the generic index terms t_i in the document d_j, for a collection D composed of n documents and with $m = |ITD|$.

Table 5.3. Term-Document Frequency Matrix

ITD	d_1	d_2	...	d_n
t_1	w_{11}	w_{12}	...	w_{1n}
t_2	w_{21}	w_{22}	...	w_{2n}
...	...	...	...	...
t_m	w_{m1}	w_{m2}	...	w_{mn}

A term's weight must ensure the needed discerning power for a correct representation and retrieval of documents containing it; thus, a term's relevance must follow some specific guidelines, namely [68, 76]:

– The more a term t appears in a document d, the more this term can characterize the topic dealt with by the document itself.
– A term t that appears in almost all documents of the collection D does not entail a relevant information content for the characterization of the topic of a particular document.

Hence, Term Weighting involves at least two components: the frequency of a term t within a document d and the frequency of term t within the whole collection D. These general guidelines provided the first calculation methods for index terms weighting, i.e., Salton and Buckley Weighting Schema [74, 75].

It is now possible to resort to more complicated approaches based on function f defined in Eq. 5.1, defining a new variable, called *Document Frequency* $DF(t)$, namely, the number of documents in which term t appears at least once. A high value of $DF(t)$ should reduce the importance of term t, that is, the reduction of its weight w. The following items show the most common calculation methods in literature, and the ones used and shown in other chapters of this book.

- *Boolean Weighting.* It represents the simplest version of the calculation of weights w_i. The calculation formula is as follows:

$$w_i = 1 \Leftrightarrow t_i \in d \tag{5.2}$$

$$w_i = 0 \Leftrightarrow t_i \notin d \tag{5.3}$$

In the case of a generic collection D of documents, we could turn to Tab. 5.4.

Table 5.4. Example of Boolean weights

ITD	d_1	d_2	...	d_n
t_1	0	1	...	0
t_2	1	0	...	1
...	...	...	...	...
t_m	1	1	...	1

A practical example of this weighting method is to be seen in the Subsection on the Boolean Model, Subsection 5.2.2, whereas application examples can be studied further in [58].

- *TFxIDF Weighting*. The weight w_i of the term t_i is calculated in such a way to be proportional to the frequency of the term t_i in the document d, and inversely proportional to the number of documents in the collection D in which t_i appears. Given that $|D|$ is the number of documents in the collection D, the weight w_i is calculated by the following formula:

$$w_i = TF(t_i, d) \log \frac{|D|}{DT(t_i)} \tag{5.4}$$

  being $DT(t_i)$ the number of documents of the collection D that include the term t_i. *TFxIDF* weighting is a very common technique, owing to its simplicity and effectiveness and many IR systems employ it in literature. In the Subsection on Vector Space Model, the reader will find a clear implementation example and a link to other chapters of the book that use it. Besides, several proposals have been made to adapt *TFxIDF* to hypertext links on the Web [41], leading to good results.
  An in-depth example of this weighting technique is shown in Chapter 10 [59]. In Chapter 3 [54] the reader can find an example of the use of TFxIDF in Content-Based Filtering Systems while in Chapter 6 [51] is shown the WATSON system [18] that exploits this weighting method to create the contextual query. Finally, another important application of TFxIDF is illustrated in Chapter 18 [12]: here this weighting technique is used in order to learn user models for news access.

- *Okapi BM25 Weighting*. It is worthwhile illustrating the Okapi weighting scheme [84] employed with good results on the benchmark TREC[8] [70, 69].
  This weighting technique is part of the probabilistic models for the calculation of term relevance within a document. This approach computes a term weight according to the probability of its appearance in a relevant document, and to the probability of it appearing in a non-relevant document in a collection D. A simplified version of the Okapi BM25 formula to assess the relevance of a term t of a document d belonging to a collection D, is the following [41]:

$$w_i = TF(t_i, d) \frac{\log \frac{(|D| - TF(t_i,d) + 0{,}5)}{(|D| + 0{,}5)}}{k_1 \cdot ((1-b) + b \cdot \frac{|d|}{\overline{d}}) + TF(t_i, d)} \tag{5.5}$$

  with $\overline{d}$ being the average length of a document included in the collection D, $|D|$ the number of documents in the collection D, k_1 and b two constants to be determined experimentally.[9]

[8] See http://www.soi.city.ac.uk/∼mg/okapi-pack/old_mg_bak/okapi-pack.html.

[9] The reader can visit http://www.soi.city.ac.uk/organisation/is/research/cisr/ for an in-depth reading of the Okapi project.

- *Entropy weighting.* This approach is based on ideas of the Information Theory. Some studies [31] proved its efficiency and performance compared to other methods. In this case, the weight is given by:

$$w_i = \log(TF(t_i, d) + 1)(1 + \frac{1}{\log(|D|)} \sum_{j=1}^{|D|} [\frac{TF(t_i, d_j)}{DF(t_i)} \log \frac{TF(t_i, d_j)}{DF(t_i)}]) \tag{5.6}$$

where the value of:

$$\frac{1}{\log(|D|)} \sum_{j=1}^{|D|} [\frac{TF(t_i, d_j)}{DF(t_i)} \log \frac{TF(t_i, d_j)}{DF(t_i)}] \tag{5.7}$$

is equal to the *Entropy* of term t_i, which is equal to -1 if t_i is equally distributed in all documents, and equal to 0 if t_i is present in only one document.

- *Genetic Programming Weighting*. Another interesting modern approach to the calculation of weights is the one based on the Genetic Programming Theory [47]. In the work of Cummins and O'Riordan [25] term weighting schemes are automatically determined by genetic evolution and then tested on standard test collections. Finally they are compared to the traditional TFxIDF weighting scheme and to the BM25 weighting scheme using standard IR performance metrics.

For further reading on Term Weighting techniques, the reader can find other methods and representations for example in the works of Park *et al.* [57] where a new and interesting method for ranking documents, based on discrete wavelet transform, is proposed.

5.2.2 Boolean Model

In the Boolean Model, documents are represented by sets of keywords, extracted manually and/or automatically from all the documents of a collection D [76, 67]. It is a very simple model, easy to understand and implement but with some effectiveness problems. This model is based on Set Theory and on Boolean Algebra: it ascribes a binary value to the weight w_i of a term t_i accordingly to its appearance (or non-appearance) in a document $d_k \in D$. In this way, the document d_k is represented by a vector $\overrightarrow{d_k}$:

$$\overrightarrow{d_k} \equiv \{(t_1, w_{1k}), (t_2, w_{2k}), \ldots, (t_m, w_{mk})\} \tag{5.8}$$

where $w_{ik} = 1$ iff $\Leftrightarrow$ $t_i \in d_k$, $w_{ik} = 0$ $otherwise$ and where terms t_i are all the index terms belonging to the ITD of the collection D. In this model, also the query is represented through Boolean expressions such as [5]:

$$q = k_\alpha \wedge (k_\beta \vee k_\gamma) \tag{5.9}$$

where terms k_i could be present or absent in the documents of the collection D.

We reckon it could be interesting, by this stage, for the reader, to apply such a representation model to a concrete example, in order to better illustrate the concepts described in this Subsection. Starting from the example of the HTML page illustrated in Fig.5.2, imagine a collection D consisting of only 3 documents, $D \equiv (d_1, d_2, d_3)$[10] and of an ITD T composed of 16 terms, $ITD \equiv (t_1, t_2, \ldots, t_{16})$, represented in the Boolean Model shown in Tab. 5.5.

Table 5.5. An example of a document collection D composed of 3 documents and of an ITD of 16 index terms in the Boolean Model

ITD	Terms	d_1	d_2	d_3
t_1	benefit	1	0	0
t_2	attract	1	1	0
t_3	save	1	1	0
t_4	springer	1	1	1
t_5	book	1	0	1
t_6	sign	0	0	1
t_7	email	1	1	0
t_8	titl	0	1	1
t_9	info	0	0	1
t_{10}	special	0	0	1
t_{11}	announc	0	0	1
t_{12}	alert	0	0	1
t_{13}	pai	0	1	0
t_{14}	inform	1	0	1
t_{15}	notif	0	1	1
t_{16}	servic	1	0	1

Suppose we have the following query:

$$q = springer \wedge (inform \vee info) \tag{5.10}$$

In this case the system will retrieve documents d_1 and d_3 from the collection D. Concluding, the Boolean Model is a very simple retrieval model based on Boolean Algebra and very easy to implement. This model only retrieves exact matches: in the example above, the document d_3 matches more keywords than the other retrieved document d_1 but nevertheless this retrieval system returns both d_1 and d_3 at the same level of relevance. Put into practice, this model appears to retrieve too much, or too little. This is also partially due to the way users write their queries. In fact, users are often not familiar with Boolean queries and tend to write too relaxed or too constraining queries that affect the search effectiveness. Finally, there is no weighting of terms in a document or in the query expression, hence all terms are equally important in this model. In the above example the term *Springer* appears in all documents of our collection, thus being somewhat less *useful* than other terms regarding the retrieval effectiveness, as highlighted in the guidelines of Subsection 5.2.1. Nevertheless the Boolean model is quite appealing for its simplicity and computational performance.

[10] After the pre-processing phases.

5.2.3 Probabilistic Model

In the Probabilistic Model [68, 44], a document (and even a query) is modeled through a binary weight vector, as is done in the Boolean Model. The difference is to be seen in the model for the calculation of the query-document similarity function. Indeed, the probabilistic model tries to answer the following Basic Question [44]:

What is the probability that a certain document is relevant to a certain query?

Furthermore, the objective of asking the Basic Question is to rank documents according to their probability of relevance. This maximizes the system effectiveness, as retrieved documents are ranked by decreasing probability of relevance. The user inspects the ranked list of documents and assess their relevance by him/herself. Assuming that terms are distributed differently in relevant and non-relevant documents, one could base the representation and retrieval of documents on term distribution. Both for query q and for the document d_j, index terms are represented by binary weights: $w_{ij} \in \{0,1\}$ and $w_{iq} \in \{0,1\}$ and the *similarity function* $sim(d_j, q)$, i.e., the function that calculates the query-document similarity, is the following:

$$sim(d_j, q) = \frac{P(R/d_j)}{P(\overline{R}/d_j)} \tag{5.11}$$

where R is the set of documents known as relevant documents while the set $\overline{R}$ is the complement of set R, namely the set of non-relevant documents. $P(R|d_j)$ is the probability that document d_j will be relevant for the query q, and $P(\overline{R}|d_j)$ is the probability that d_j will not be relevant for q.

Using the Bayes theorem [30]:

$$P(a/b) = \frac{P(b/a)P(a)}{P(b)} \tag{5.12}$$

and using odds rather than probabilities, defined as:

$$O(z) = \frac{P(z)}{P(\overline{z})} = \frac{P(z)}{1 - P(z)} \tag{5.13}$$

we can calculate Eq. 5.11 as follows:

$$sim(d_j, q) = \frac{P(d_j/R)P(R)}{P(d_j|\overline{R})P(\overline{R})} \tag{5.14}$$

Assuming that $P(R)$ and $P(\overline{R})$ are constant for each document we get:

$$sim(d_j, q) \sim \frac{P(d_j/R)}{P(d_j/\overline{R})} \tag{5.15}$$

Assuming that terms occur independently of each other and switching to logarithms (skipping some steps for brevity) we then have:

$$sim(d_j, q) \sim \sum_{i=1}^{n} log \frac{P(t_i/R)P(\overline{t_i}/\overline{R})}{P(t_i/\overline{R})P(\overline{t_i}/R)} \tag{5.16}$$

Having thought of d_j as a vector of n binary independent term occurrences:

$$P(d_j/R) = \prod_{t=1}^{n} P(t_i/R) \tag{5.17}$$

where $P(t_i/R)$ is the probability that term t_i appears in a document randomly selected from set R, and $P(\overline{t_i}/R)$ the probability that term t_i is not present in a document randomly selected from set R. Eq. 5.16 is not usable at start up, when there are no retrieved documents. In this case a number of simplifying assumptions can be done as follows:

$$P(t_i/R) = 0.5 \tag{5.18}$$

$$P(t_i/\overline{R}) = n_i/N \tag{5.19}$$

with $N = |D|$ and $n_i = |ITD|$. Let's see now a practical example of this simple probabilistic model at work for our imaginary collection D, formed by 3 documents. Tab. 5.6 illustrates, in the last column, the value calculated by the Eq. 5.19 of $P(t_i/\overline{R})$, used to start up the system. By this technique, the initial values calculated through Eq. 5.18 and Eq. 5.19 are modified taking into account the distribution of terms t_i in the retrieved documents, considering the non retrieved documents as non-relevant.

Table 5.6. Example of a document collection D consisting of 3 documents, represented by the Probabilistic Model

ITD	Terms	d_1	d_2	d_3	$P(t_i/\overline{R})$
t_1	benef	1	0	0	0.33
t_2	attract	1	1	0	0.66
t_3	sav	1	1	0	0.66
t_4	springer	1	1	1	1
t_5	book	1	0	1	0.66
t_6	sign	0	1	0	0.33
t_7	email	1	1	0	0.66
t_8	titl	0	1	1	0.66
t_9	info	0	0	1	0.33
t_{10}	special	0	0	1	0.33
t_{11}	announc	0	0	1	0.33
t_{12}	alert	0	0	1	0.33
t_{13}	pai	0	1	0	0.33
t_{14}	inform	1	0	1	0.66
t_{15}	notif	0	1	1	0.66
t_{16}	servic	1	0	1	0.66

The final calculation formula for $P(t_i/R)$ and $P(t_i/\overline{R})$ is played down to Eq. 5.20 and 5.21 [5].

$$P(t_i/R) = \frac{V_i + \frac{n_i}{N}}{V+1} \tag{5.20}$$

$$P(t_i/\overline{R}) = \frac{n_i - V_i + \frac{n_i}{N}}{N-V+1} \tag{5.21}$$

where $N = |D|$, $n_i = DT(t_i)$, V is the overall number of documents currently retrieved and V_i is the number of such documents containing the term t_i.

The main asset of the probabilistic model is that retrieved documents may be ranked in a descending order, according to their relevance probability, while the main drawbacks of this model are the following:

- Division of the set of documents into relevant and non-relevant documents.
- Index Terms. This approach does not take into account the frequency of index terms in documents: all weights are binary.
- Assumption that terms are all independent from each other: this assumption enables to devise a formula of the calculation of Eq. 5.11 actually computable.

5.2.4 Vector Space Model

In this Subsection, we introduce the *Vector Space* representation of a document [74]. This model provides a technique to retrieve relevant documents from any set of documents. Still taking cue from the Web page previously used (www.springeronline.com), with a collection D consisting of 3 HTML pages taken from this web site, firstly submit each one of them to the pre-processing phases illustrated in Subsection 5.2.1. Secondly, build the Term-Document Frequency Matrix A, as defined in Subsection 5.2.1 and shown in Tab. 5.7. Thirdly, compute the TFxIDF values for each index term of each document included in the collection D.[11]

Table 5.7. Representation of our example collection of documents by Term-Document Frequency Matrix

ITD	Terms	d_1	d_2	$d3$
t_1	benef	1.0	0.0	3.0
t_2	attract	1.0	2.0	0.0
t_3	sav	1.0	0.0	0.0
t_4	springer	1.0	3.0	0.0
t_5	book	1.0	0.0	1.0

[11] The exposed values of Tab. 5.7 are purely indicative, considering the calculation simplicity and clarity.

The last step is to consider a document as a vector of real numbers in an m-dimensional space, namely, considering documents as points in an m-dimensional space, with $m = |ITD|$, where every term t_i represents a dimension. Such an m dimension could be very high, even several thousands, depending on the number, on the length of documents in the collection D and on the semantic domain from where collection D was gathered. To avoid computational problems in computing the Vector Space Model, due to high-dimension [67],[12] it is possible to resort to a *Dimension Reduction* process where, starting from the original m-dimension high space, a new n-dimension space is built, with $n \ll m$. Generally a document d is a very sparse vector, i.e., it contains a very small subset of the ITD. Every term $t_i \in d_j$ has a weight w_{ij}, so that: $w_{ij} > 0$ iff $t_i \in d_j$ and t_i does not belong to all documents of the collection D. The value of w_{ij} is thus the coordinate calculated according to the correspondent index term, as shown in Eq. 5.22. Obviously such weights represent the relevance of the associated terms in the very document.

$$\overrightarrow{d_j} = \{w_{1j}, w_{2j}, \dots w_{mj}\} \tag{5.22}$$

In the case in point, by enforcing the rules of formation of weights TFxIDF (see Subsection 5.2.1) we have the three vectors represented in the rows of Tab. 5.8.

Table 5.8. Representation of our example collection of documents by the Space Vector Model with TFxIDF weighting

Document	w_1	w_2	w_3	w_4	w_5
$\overrightarrow{d_1}$	0.176	0.176	0.417	0.176	0.176
$\overrightarrow{d_2}$	0.000	0.350	0.000	0.528	0.000
$\overrightarrow{d_3}$	0.528	0.000	0.000	0.000	0.176

Each direction of the vector space corresponds to a unique index term in the document collection, while the component of a document vector along a given direction corresponds to the importance of that term to the document. In such a geometric representation, versors $\overrightarrow{T_k}$ are assumed to be orthonormal (namely, index terms are assumed to appear independently from each other in documents). The importance of this type of document representation is the fact that it enables, in a simple way, even the representation of a query q, since, in this space, both query and documents can be represented through vectors of weights. In particular, even query q is transformed as if it were a document, following the same rules: an m-dimensional vector representation is obtained, as expressed in Eq. 5.23. Obviously, the q vector will be a very sparse one.

$$\overrightarrow{q} = \{w_{1q}, w_{2q}, \dots w_{mq}\} \tag{5.23}$$

The advantage of this document representation is that of enabling to retrieve relevant documents through a very simple document-query similarity function. In fact, the similarity function is fulfilled through a function that also calculates the similarity ranking

[12] In the IR literature such a problem is known as the so-called *Curse of Dimensionality.*

of single documents $\overrightarrow{d}$ with respect to the query $\overrightarrow{q}$. The most employed similarity function in the literature is the cosine one:

$$sim(d_j, q) = \cos(\overrightarrow{d_j}, \overrightarrow{q}) = \frac{\overrightarrow{d_j} \bullet \overrightarrow{q}}{|\overrightarrow{d_j}||\overrightarrow{q}|} \tag{5.24}$$

Indeed, this measure is equal to the cosine of the angle formed by the two vectors $\overrightarrow{d_j}$ and $\overrightarrow{q}$ in the m-dimension vector space. Some of the main benefits of this model include:

– Term weighting that enhances response quality.
– Since partial matching between documents and queries may occur, it is possible to obtain responses by approximating the user's requests. In fact, the cosine angle formula allows to rank documents according to similarity document-query.

We may assert that the main drawback of this representation is the assumption that terms are independent from each other, although there is no proof this actually is a drawback in real IR systems. In Chapter 3, [54], Chapter 10 [59] and Chapter 18 [12], the reader can find some examples of the use of the Vector Space Model to represent documents.

More complex text analyses dealing with morphologic-syntactic analysis, semantics analysis and structure terminology analysis are not included in this chapter.[13] Should the reader want to study more in-depth what is illustrated in this Section, s/he can download *IRTools*, a software toolkit intended for IR research.[14] Finally, in Chapter 14 [21] the reader can find other methods of document modeling apt for representing 3D graphical objects, such as X3D-based Web documents.

5.3 Web-IR Document Representations

The representations of documents illustrated in the previous Section come from the traditional IR research field, whose original goal was to retrieve relevant documents matching user needs, expressed through queries. With the advent of the Web and of the HTML language, used to write the vast majority of web documents, other document representations have been suggested, not only based on the single terms that made them up, as occurs for classic IR, but also on other features typical of the hypertext environment, expressed through hyperlinks and/or HTML tags. Indeed, the Web can be considered a huge hypertext and hence interesting for the *Hypertext Information Retrieval* (HIR) research field [2, 24, 33, 23]. That's how the Web-IR came about. It expresses the union between classic IR and HIR for the Web, namely the enhancement of the classic pre-Web IR systems. It exploits the characteristics of hypertext languages of documents [1]: a term t of a Web document d is given a weight also according to the HTML tags between which it is. For example, if a term t appears between the two HTML tags <TITLE> and $<$ /TITLE>, it means it is part of the document title, and must thus be significantly weighted, as it could be correlated to the semantics of the

[13] Links to several IR resources, including test collections, IRS lists and text-analysis tools can be found on the http://www.dcs.gla.ac.uk/idom/ir_resources Web site.

[14] Web site: http://sourceforge.net/projects/irtools.

very document, unlike a term included between the two HTML tags <BODY> and < /BODY>, a part of a plain text, where its importance could be relative. An example is to be seen in Fig. 5.3, which highlights the terms between the two HTML tags <TITLE> and < /TITLE>:

```
<TITLE>

 Springer UK - Academic Journals, Books and Online Media

</TITLE>
```

there are only a few terms, but they are all significant and correlated to the general semantics of the Web page, while in the following text fragment, taken from the same source:

```
<span class="bodytext">

Benefit from attractive savings on Springer books
by signing up for Springer's free new book e-mail
notification service. New title info, news and
special announcements: with Springer Alerts it
pays to be informed.


</span>
```

there are some terms not all strictly correlated to the semantics of the page in which the text fragment is included (e.g.,"new" and "free").

Some important algorithms adopted by Web search engines, such as *HITS* [46] and *PageRank* [14], mainly exploit hypertext links between pages, considerably improving retrieval processes. In order to better guide the reader in an in-depth analysis of these modeling techniques, this Section has been broken down into two parts: the first part shows document modeling techniques mainly based on HTML tags, with simple, explicative examples. The second part illustrates modeling techniques based on hypertext links, and clarifies *HITS* and *PageRank*, two algorithms which are currently the most commonly used search ones on the Web.

For in-depth studies on semi-structured document modeling examples on the Web, the reader can refer for example to INEX conferences, Initiative for the Evaluation of XML Retrieval [37].[15]

5.3.1 Tag-Based Document Models

As mentioned in the introduction of this section, many proposals in literature have tried to enhance the performance of classic IR systems, mostly based on Vector Space Model, taking into account the possibilities offered by the HTML language. The idea is that of modifying the document pre-processing process, illustrated in Subsection 5.2.1, taking

[15] Visit http://www.informatik.uni-trier.de/∼ ley/db/conf/inex/index.html.

into account the HTML structure of the Web page as well. Clear examples of document modeling based on HTML tags are to be found in [27, 55, 87, 88]. Now we describe three relevant examples on the use of such techniques.

The IRIS System. In [87] we have a Vector Space representation of the document, where the IRIS system [89] calculates the weight of index terms even according to the tags between which they are comprised, such as heading texts (i.e., terms between <Hn> and < /Hn>tags), titles, and meta-keywords. We now illustrate the document modeling phase and the term-weighting phase, in order to better understand the retrieval and modeling process.

- *Document Modeling Phase.* The document is modeled into three main parts:
  – Body text terms.
  – Heading text terms.
  – A combination of the two.

  In the case in point, we have the scheme illustrated in Fig. 5.4, where, in order to make things more straightforward, the single terms have been left as they appear in the fragment of the Web page used as an example, without undergoing the stopword and stemming elimination phases of Subsection 5.2.1. After having subdivided the document into three parts, as shown in Fig. 5.4, the IRIS system, depending on the query length, uses them as index term sources.

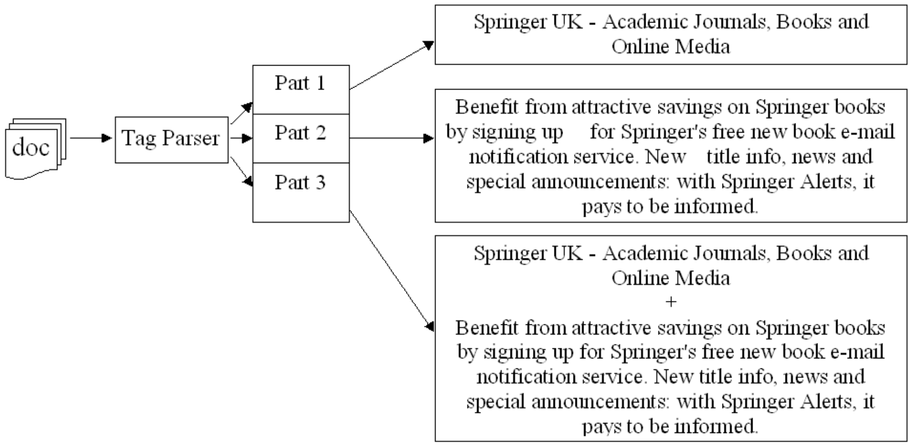

doc
Tag Parser
Part 1
Part 2
Part 3
Springer UK - Academic Journals, Books and Online Media
Benefit from attractive savings on Springer books by signing up for Springer's free new book e-mail notification service. New title info, news and special announcements: with Springer Alerts, it pays to be informed.
Springer UK - Academic Journals, Books and Online Media
+
Benefit from attractive savings on Springer books by signing up for Springer's free new book e-mail notification service. New title info, news and special announcements: with Springer Alerts, it pays to be informed.

Fig. 5.4. The document modeling process used by the IRIS system

- *Term Weighting Phase.* The single terms are weighted differently, even depending on their position within the document itself. For example, the frequency of single terms comprised in header texts is multiplied by 10. Subsequently, one of the classic weighting system technique, the *SMART lnu* weighting schema with slope 0.3 [17, 16], is adopted.

The example document thus provides three different frequency tables, to be used to weight terms.

An Indexing HTML Model. Another interesting example of tag-based document modeling is to be seen in [55, 62]. In this case, the layout structure of an HTML document is taken into consideration: a *Tag Ranking* weights the several index terms as illustrated in Tab. 5.9.

Table 5.9. The tag classes hierarchy used by the Indexing Model in [62]

Rank	CLASS NAME	CLASSIFIED TAGS/PARAMETERS
1	Title	TITLE, META keyword
2	Header 1	H1, FONT SIZE=7
3	Header 2	H2, FONT SIZE=6
4	Header 3	H3, FONT SIZE=5
5	Linking	HREF
6	Emphasized	EM, STRONG, B, U, I, STRIKE, S, BLINK, ALT
7	Lists	UL, OL, DL, MENU, DIR
8	Emphasized 2	BLOCK QUOTE, CITE, BIG, PRE, CENTER, TH, TT
9	Header 4	H4, CAPTION, CENTER, FONT SIZE = 4
10	Header 5	F5, FONT SIZE = 3
11	Header 6	H6, FONT SIZE=2
12	Delimiters	P,DT,....

- *Document Modeling Phase*. After the HTML parsing stage, the document is shown as modeled in the 12 classes of terms illustrated in Tab. 5.9. Basically, 12 classes of terms are formed, which together make up the document model after the parsing phase, as illustrated in Fig. 5.5. The overall document is then represented by a vector $\overrightarrow{S} \equiv \{S_1, S_2, \ldots S_{12}\}$, where $S_i \geq 0$ represents the number of terms in the tag class C_i.

- *Term Weighting Phase.* The frequency distribution of a term t is represented by a vector $\overrightarrow{T} \equiv \{T_1, T_2, \ldots T_n\}$, where $T_i \geq 0$ represents the frequency of term t in the tag class C_i and $n = |ITD|$. Firstly, a weight w_i is assigned to each tag class C_i. The weight to be associated with the single term t, $F(d,t)$, is then calculated as the weighted average, with weight w_i, of the several normalized frequencies of the term in all the tag classes, F_{ctag_i}, multiplied by the inverse of the term frequency in the documents collection, IDF_t, as expressed by the Eq. 5.25, similar to the classic formula TFxIDF (see Section 5.2.1):

$$F(d,t) = \sum_{i=1}^{n} w_i F_{ctag_i}(d,t)(IDF_t) \tag{5.25}$$

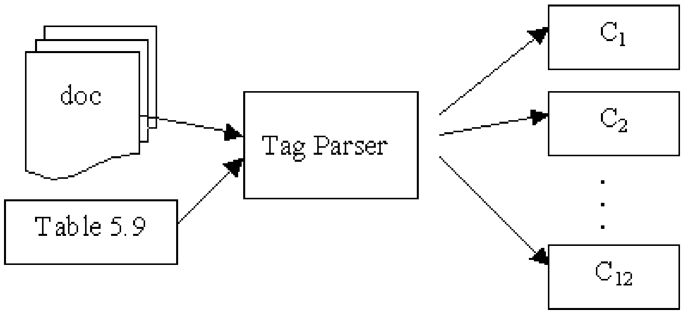


Fig. 5.5. The document modeling process used in [55]

The WEBOR System. Another useful example for the reader is to be seen in [27, 26], where the tag hierarchy is formed only by the six classes illustrated in Tab. 5.10. The IR system employed is *WEBOR*, WEB-based search tool for Organization Retrieval. [16]

Table 5.10. The tag class hierarchy used by the WEBOR system

Class Name	HTML Tags
Anchor	A
H1-H2	H1-H2
H3-H6	H3, H4, H5, H6
Strong	STRONG, B, EM, I, U, DL, OL, UL
Title	TITLE
Plain Text	None of the above

- *Document Modeling Phase.* The document modeling method consists of two stages:
  – Parsing HTML. The document is broken down into 6 parts, each one formed by the groups of terms of the document contained between the tags as shown in Tab. 5.10.
  – Vector Space Modeling. In this stage, the document is modeled with the classic Vector Space Model, to lead the query-similarity operation following the Eq. 5.24.
- *Term Weighting Phase.* Firstly, a vector $\overrightarrow{CIV}$, *Class Importance Vector*, $\overrightarrow{CIV} \equiv \{civ_1, civ_2, \dots civ_6\}$ is formed, where $civ_i \geq 0$ represents a weight determined experimentally by means of genetic algorithms. Such a weight is the contribution of the $i-th$ tag class to the formation of the overall weight of term t in document d. Another vector is then formed, $\overrightarrow{TFV}$, *Term Frequency Vector*, which contains the occurrence frequency of term t in document d for each tag class of Tab. 5.10: $\overrightarrow{TFV} \equiv \{tif_1, tif_2, \dots tif_6\}$. The last step is the calculation of weight w, to be associated with term t by the following formula:

[16] See http://nexus.data.binghamton.edu/∼ yungming/webor.html.

$$w_t = (\overrightarrow{CIV} \bullet \overrightarrow{TFV})idf \tag{5.26}$$

where idf is the inverse document frequency of term t in the documents of the collection D.
Finally, the document vector $\overrightarrow{d} \equiv \{w_1, w_2, \ldots, w_n\}$ is built. It contains the weight of each term of the ITD with respect to the document, to be used to carry out the query-similarity operation, as illustrated in Subsection 5.2.1.

5.3.2 Link-Based Web Document Models

The previous Subsection described the document modeling techniques that use HTML features to strengthen classic IR systems for the Web. This section ends with the illustration of document modeling examples based on HTML hypertext links, and on the two algorithms *HITS* [46] and *PageRank* [14] which utilize them. As previously mentioned, the Web can be considered a huge hypertext $G = (V, E)$, formed by documents V connected through links E. In this context, an HTML page can be modeled as a hypertext node featuring hypertext in-links (i.e., anchors) and outgoing links.

HITS. The first step in order to run a hypertextual algorithm, such as HITS, is to build the underlying hyperlink-directed graph from the set of available pages $G = (V, E)$. It consists of a set of nodes V and of a set of edges E, where each edge is an ordered pair of nodes. The *in-degree* and the *out-degree* of a node u are the number of nodes v such that $(v, u) \in E$ and the number of nodes v, such that $(u, v) \in E$ respectively. Of course, each node corresponds to a page and an edge is a hypertextual link between two pages.

Following the idea of backlink count that assigns a rank to a page according to its popularity, that is, the number of pages pointing to it, the in-degree can also be employed in more sophisticated algorithms where the importance of pointing pages is recursively taken into consideration.

Regarding scholarly publications, where it is possible to recognize a class of papers related to surveys and reviews on a particular topic that cite many significant research works, Kleinberg identifies two classes of Web pages. *Authorities*, that have relevant content about a particular topic, and *Hubs*, which contain several links to relevant authoritative pages [46].

A recursive algorithm has been conceived to identify highly relevant authority and hub pages, a recursive algorithm has been conceived. A query-dependent subgraph of the Web is chosen for the analysis, retrieving the first results of a *broad-topic query* from a search engine. This kind of query is characterized by an appreciable number of relevant pages that can overload user search activity. The initial root set of highest-ranked pages is then expanded considering all the pages that point to it and all the pages pointed by it, by means of a search engine that allows this kind of query on the link structure among pages. The obtained set is the input of the HITS iterative algorithm that assigns each page the two measures, authority and hubness, following this method:

$$a_p \leftarrow \sum_{q:(q,p)\in E} h_q \tag{5.27}$$

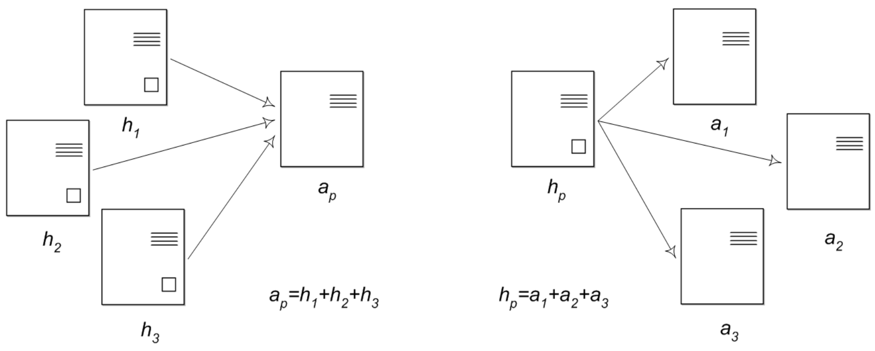


Fig. 5.6. An example of how the authority a and hubness h measures for a page p are drawn

$$h_p \leftarrow \sum_{q:(p,q)\in E} a_q \tag{5.28}$$

where a and h are the authority and hubness of pages respectively, and E is the link set (see also Fig. 5.6). After a certain number of iterations, the equilibrium is reached and the two measures are assigned to each page. The basic steps of the HITS algorithm is itemized in Algorithm 1. An interesting way to analyze the output is to make two copies of each page and visualize the graph as bipartited, where the hubs point to the authorities. Details on the algorithm and the related matrix formulation, the time and the conditions of convergence are to be seen in [46].

Algorithm 1. Pseudo-code of the HITS algorithm, where authorities a_p and hubs h_p of the pages are stored in the vectors $\boldsymbol{a}$ and $\boldsymbol{h}$.

```
$\boldsymbol{V} \leftarrow$ collection of n pages
$\boldsymbol{N} \leftarrow$ number of iterations
$\boldsymbol{z} \leftarrow (\boldsymbol{1}, \boldsymbol{1}, ..., \boldsymbol{1}) \in \Re^{|V|}$
$\boldsymbol{a}_0 \leftarrow \boldsymbol{z}$
$\boldsymbol{h}_0 \leftarrow \boldsymbol{z}$
for $\boldsymbol{i} = \boldsymbol{0}$ to $\boldsymbol{N}$ do
    {apply Eq. 5.27 to $(\boldsymbol{a}_{i-1}; \boldsymbol{h}_{i-1})$ and draw the new authority vector $\hat{\boldsymbol{a}}_i$}
    {apply Eq. 5.28 to $(\hat{\boldsymbol{a}}_i; \boldsymbol{h}_{i-1})$ and draw the new hub vector $\hat{\boldsymbol{h}}_i$}
    {normalize both $\hat{\boldsymbol{a}}_i$ and $\hat{\boldsymbol{h}}_i$ to 1}
    $\boldsymbol{a}_i \leftarrow \hat{\boldsymbol{a}}_i$
    $\boldsymbol{h}_i \leftarrow \hat{\boldsymbol{h}}_i$
end for
```

The output of the HITS algorithm consists of two lists: the pages with the highest hubness, and the ones with the highest authority. The latter can be used to better rank the pages of the initial set built from the results of broad-topic queries, a process named *Topic Distillation*. The hub list is useful for users or for crawling systems that need valid starting points to begin their seeking activities. The pages included in that list point to high authoritative pages, and are therefore useful to quickly find good resources. The HITS has also been used to identify and analyze fine-grained hyperlinked communities, and the related topics of interest [20]. Another method to discover hub pages through link analysis is discussed in [56].

Several enhancements of the HITS algorithm that achieve improvements in terms of precision, and fix some potential problems on the original formulation, such as the *Topic Drift* phenomenon, are described in [10]. Amento *et al.* [3] offers a comparison of hyperlink-based metrics, such as HITS and PageRank, with judgments given by humans about the quality of pages.

PageRank. While Kleinberg proposes a two-level weight propagation scheme where authorities and hubs are the two measures taken into consideration, Page and Brin suggest to assign a single measure to the pages, with value being high if the sum of the measures of its backlinks is high [14]. Therefore, it is possible to see the related PageRank algorithm as a direct enhancement of the classic backlink count.

One of the assets of PageRank over HITS is that we do not need to restrict the calculation to a subgraph of the Web relevant to a given query. For a general-purpose search engine, hypertext-based algorithms, such as the described ones, are computational, intensive processes, that cannot be performed for each submitted query. The only way to include those kinds of relevance measures is to periodically run the algorithm off-line, keeping all sets of values up-to-date. Focused crawlers tailor their search to a subset of topics and retrieve much less documents, and for this reason it is possible to run algorithms such as HITS on-line, during the crawling.

Assuming the Web as a strongly connected graph, the PageRank can be described through the *Random surfer model*. A surfer in that model is able to randomly click on one of the links contained in a page p with equal probability $1/N_p$, where N_p is the number of links in p. A simplified formula to compute the PageRank is:

$$rank(p) = c \sum_{q:(q,p)\in E} \frac{rank(q)}{N_q} \tag{5.29}$$

where c is a normalization constant less than 1. Each page q contributes with a quantity that is proportional to its rank $rank(q)$ but inversely proportional to the number of links N_q. So the PageRank flows from one page to another, decreasing its value if the out-degree of a page is high. Like the HITS algorithm, the equation is recursive and it must be computed until convergence. In the random surfer model the rank could be seen as the probability that the surfer is currently browsing a given page.

The Web is not strongly connected thus there cannot be a situation in which two pages are linked to each other and do not have any outgoing links. In this scenario, the two pages will increase their rank without transferring the value to other documents outside the loop. For this reason, the previous formula has been enhanced:

$$rank(p) = c \sum_{q:(q,p)\in E} \frac{rank(q)}{N_q} + \frac{(1-c)}{N} \tag{5.30}$$

where N is some vector that gives the source of the rank for each page in the corpus.

The second term denotes the event that the imaginary surfer who is randomly clicking on links will eventually stop clicking and start visiting a different page. It is generally assumed that this factor assumes the constant value 0.15.

As in the case of HITS, PageRank can be periodically calculated to keep the ranks associated with the retrieved pages in a given collection up-to-date. The best ranked pages can be further analyzed by focused crawlers, for example, by extracting the outgoing links and putting them at the top of the URL QUEUE. In order to tackle the Web-spam issue, an enhanced version of PageRank named *TrustRank* has been proposed [40]. In Chapter 7 [50] the reader can find several examples on the use of a link-based document representation for Focused Crawling.

5.4 Concept-Based Document Modeling: Latent Semantic Indexing

With the Latent Semantic Indexing (LSI) technique, the reader may complete the general survey on the most common document modeling techniques employed in IR literature. This Section illustrates a further extension of the document model, toward a representation based on concepts and semantic relations between index terms.

LSI [32, 28] is a document representation technique that assumes there is some hidden structure in using the terms included in a collection of documents: the topic dealt with by a text is more associated with the concepts that are used to describe it rather than with the terms actually used; hence, the idea is to represent a document through concepts, rather than through index terms. In order to do so, the high dimensional space $ITD \equiv \{t_1, t_2, \ldots, t_m\}$, formed by all the m index terms of a document collection, is mapped by means of Linear Algebra techniques into a lower dimensionality space S^n, with $n << m$, where every component s_j represents a *concept*. This can be obtained by clustering the terms t_i into sets s_j, to form a sort of *association by concepts*. This technique automatically solves synonymity problems, since the terms that appear most frequently together are grouped in the same concept. The fundamental LSI techniques chiefly focus on the correlations between documents and terms. Once such nexuses are found, the goal becomes that of understanding such relations and highlighting the most relevant ones, even when a linguistic nexus is not available. However, it should be pointed out that, broadly speaking, the result of LSI techniques cannot be interpreted from a linguistic viewpoint. Such a result has a purely mathematical value.

In order to better understand this document modeling technique, the reader can refer to the example of the set of documents used in Subsection 5.2.4. The starting point is the term-document matrix A_{ij} that, in our example, is the matrix shown in Tab. 5.11. It is obtained with $D \equiv \{d_1, d_2, d_3\}$, $n = 3$ documents, for a total of $m = 7$ index terms, i.e., $|ITD| = 7$. Thus, we still have to decide which is the best technique to reveal these hidden relations within the example matrix illustrated in Tab. 5.11. In order to do so, we can resort to a mathematical technique called *Singular Value Decomposition* (SVD).
SVD is a widespread technique in the solving of problems such as matrix rank estimation and the canonical analysis of correlations [38, 9]. Given a matrix $A_{n,m}$, without losing in generality, we can suppose $m \geq n$ and $rank(A) = r$. The SVD of matrix A, indicated as $SVD(A)$, is defined as follows:

Table 5.11. Term-Document Matrix A

ITD	d_1	d_2	d_3
new	1	1	0
benefit	1	1	0
attractive	0	1	0
service	1	0	0
springer	0	0	1
info	0	1	1
special	0	0	1

$$A = U \cdot \Sigma \cdot V^T \tag{5.31}$$

where U is a matrix $m \times r$ orthonormal ($U^T \cdot U = I_r$), V is a matrix $n \times r$ orthonormal ($V^T \cdot V = I_r$), $\Sigma = \text{diag}(\sigma_1,\sigma_2,\ldots,\sigma_n)$, with $\sigma_i > 0$, $1 \leq i \leq r$ and $\sigma_i > \sigma_{i+1}$.
The matrix calculus that reduces the space is represented through the following notation:

$$A_k = U_k \cdot \Sigma_k \cdot V_k^T \tag{5.32}$$

where U_k is a matrix $m \times k$ obtained by taking the first k columns from U, V_k is a matrix $n \times k$ obtained by taking the first columns from V, Σ_k is a matrix $k \times k$ obtained from the first k values of the diagonal of Σ.

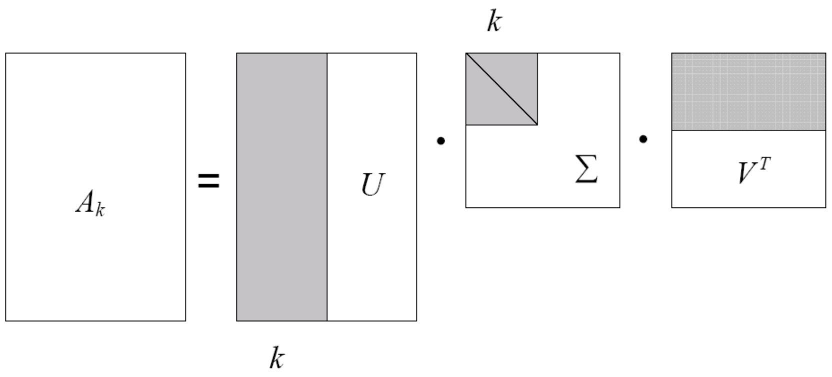


Fig. 5.7. Singular Value Decomposition

Fig. 5.7 reflects the reduction (to rank k) of matrix A, which is used by LSI to get hold of the semantic structure of the used index terms. If $k = r$ then LSI executes the similarity of literal matching (since $A_k = A$, thus two documents are similar only if they are identical). If $k = 1$ all the documents are associated with the same concept. Intuitively, k represents the measure of the overall number of concepts that are to be found in the several documents. By using the above mentioned reduction process, lesser terminology discrepancies are ignored.

Going back to the matrix in the previous example, its rank is $r = 3$, and Σ is given by:

$$\Sigma = diag(2.41, 1.73, 1.10) \tag{5.33}$$

By choosing, for example, $k = 2$, we get:

$$\Sigma = diag(2.41, 1.73) \tag{5.34}$$

The resulting matrix A_k is shown in Tab. 5.12.

Table 5.12. The resulting Matrix A_k

-1.3457	1.0762	1.6799
-1.3518	-0.0000	-1.0829

In this matrix, every column represents one document in the new compressed space. In Chapter 6 [51] the reader can find an example of an SVD application to model a click-through element.

5.5 AI-Based Document Representations

The previous Sections illustrated document modeling techniques based on classic IR, on Web-IR methods and on concept sets. We saw that such methods have the advantage of being simple, allowing to develop automatic IR systems offering good performances. For these reasons, they are widely used in real systems. Nonetheless, literature offers several other document representation methods, such as the ones based on Artificial Intelligence techniques, on which this very paragraph focuses. A typical feature of these methods is that of modeling a document through a richer and more complex knowledge representation of the domain, even though it sometimes entails a higher computational effort. The following Subsections illustrate some of these techniques used is several adaptive Web-Based systems: Artificial Neural Networks, Semantic Networks and Bayesian Networks.

5.5.1 Artificial Neural Networks

Artificial Neural Networks (ANN) provide a method for the automatic understanding of classification and regression functions [42, 13, 71]. An ANN comprises of a certain number of nodes or *neurons* connected by arcs or *synapses*, each one associated to a real value w called *weight*. Each neuron is characterized by an *Activation State*, as determined by the input values and by the weight of the corresponding connections via an *Activation Function*. In literature there are many types of ANN [71]: from the classical *Multi-Layer Perceptron* (MLP), consisting of an input layer, one or more hidden layers and an output layer, with feed-forward synapses and supervised learning, to *Self Organizing Maps* (SOM) networks, consisting of only two layers, of the feed-forward type, but with a unsupervised learning. The latter are very commonly used in Web document clustering (see for example [86, 83, 73]). Other types of networks, such as ART networks, entail more complex architectures, which lie outside the purposes illustrated in this chapters. Several IR systems use ANN to model documents, queries and, in the case of adaptive systems, even users. However, the goal is still that of retrieving the document that mostly fits the user's query.

Generally, a multi-layer system is built to model a document with an ANN: the query's terms are associated with the input layer of neurons, the terms of a set of documents with an intermediate layer, and each document with an output neuron. Following a query, the network is *trained* to provide, in output, a ranking of the documents that are most similar to that query.

An example of an ANN used for document modeling is the one illustrated in [85, 5], where the network actually forms an IR system, whereas a document, through its terms, forms a layer. In this model, an ANN is formed by the terms of the query, input neurons, the terms of all the documents of the collection, neurons of the hidden layer, and by the documents themselves, neurons of the output layer. The reader may find interesting a brief description of the system's operating mechanism on the whole, in order to better comprehend the use of an ANN for document modeling. Consider a collection D of documents, $D \equiv \{d_1, d_2, d_3\}$, consisting of the following three documents, after the pre-processing phases (see Subsection 5.2.1): $d_1 \equiv$ {new, benefit, service}, $d_2 \equiv$ {new, benefit, attractive, info} and $d_3 \equiv${springer, info, special} built from three Web pages taken from the Web site shown as an example in Section 5.2. The ITD is given by:

$$T \equiv \{\text{new, benefit, attractive, service, springer, info, special}\}$$

Finally, consider a query q:

$$q \equiv \{\text{benefit, service, info}\}$$

According to this model, all the terms of set T form an intermediate layer of neurons, each one bi-directionally connected to the document or documents containing it. The inputting of query q activates only the neurons corresponding to the neuron-terms in the set of the intermediate layer, belonging to the query, highlighted in Fig. 5.8; subsequently, an iterative process activates the last layer's neurons, which represent the documents to be retrieved, as illustrated in Fig. 5.8. The connections or synapsis between the generic term t_i of the intermediate layer and the generic documents d_j that contain it, is indicated by w_{ij} and calculated with the formula of a normalized TFxIDF [85]:

$$w_{ij} = \frac{w^*_{ij}}{\sqrt{\sum_{j=1}^{n} d^2_{ij}}}$$

being w^*_{ij} the TFxIDF weight and $n = |ITD|$. If the sum of the signals received by the terms activated by the query exceeds a certain threshold, then the neuron-document emits a signal that is input, weighted by synapsis, into all the nodes of the terms that form it, and that's the path it follows until the activated node-documents are stable. Each document is thus represented as a set of neurons. In this way, documents are selected according to the query and ranked on the basis of the query-document similarity. This

ANN features a mixed topology: the first connections are feed-forward while the others are bidirectional, as shown in Fig. 5.8.

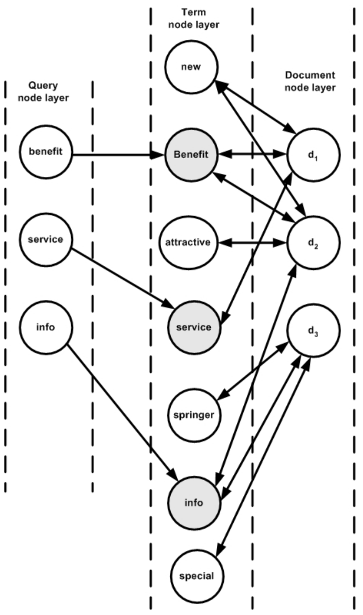


Fig. 5.8. An Example of an IR System Based on Artificial Neural Networks

Another interesting type of document modeling through ANN can be seen in [45]. Here, the basic idea is to use a set of neural networks, represented by three MLP, to perform a score representing the correlation among terms and documents as shown in Fig. 5.9. Two ANN are trained starting from the pairs (term, document) taken as input, while the term's absence or presence in the document is taken as output: if the term t_i belongs to the document d_j, the score is high; low on the contrary. Starting from a TFxIDF weighing for document d_j and from a *one-hot* representation for term t_i, two more representations of terms and documents are built, respectively t_i^* and d_j^*. In particular, the d_j^* representation is an enriched representation of the document through the neural network, that takes into account the probability distribution of single terms within the set of documents [8]. In this way, given a term and a document, it is possible to perform a term-document score, to be used for retrieval.

The reader can find another interesting example of an ANN application to model small documents such as Web links in Chapter 7 [50] while an example to e-mail routing can be found in [22]. In this routing system, named *LINGER*, incoming e-mails that are first pre-processed (see Subsection 5.2.1) form the input layer of a MLP. Every output node of the network represents a single predefined category.

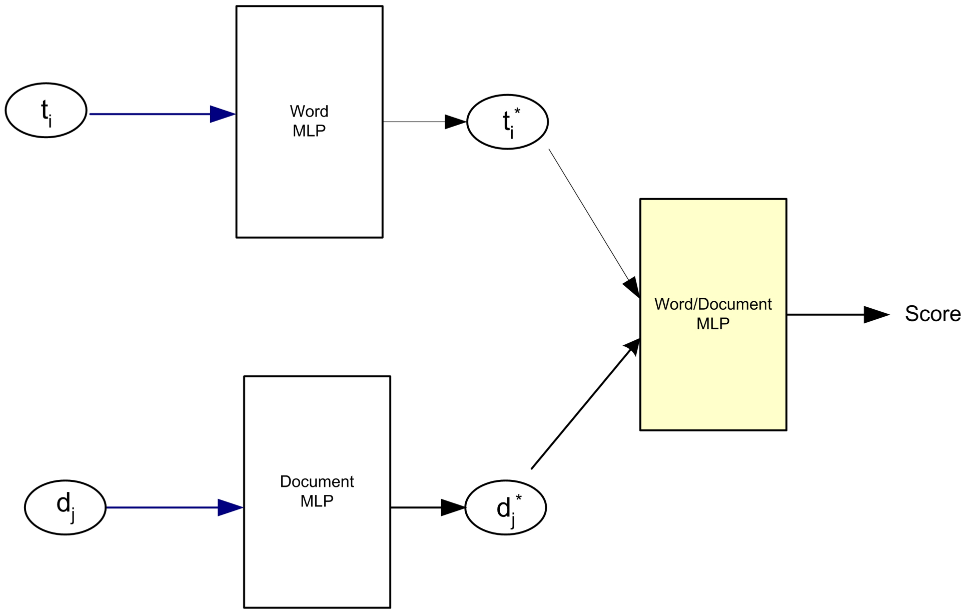


Fig. 5.9. A system based on Artificial Neural Networks to perform a term-document similarity score

5.5.2 Semantic Networks

Semantic Networks (SN) are useful for representing conceptual knowledge and, in particular, the relationship between concepts [66, 79]. In general, a SN is formed by a directed graph, whose nodes are organized in hierarchic structures, while the arcs connecting them represent the binary relations between them, such as relations *is-a* and *part-of* (see for example [72]).

An example of a system using SN for document modeling is to be seen in [6]. This system presents a conceptual indexing method based on WORDNET, a large lexical database, organized as a freely available semantic network,[17] which has received a lot of attention within the computational linguistics community. Nouns, verbs, adjectives and adverbs are organized into synonym sets (i.e., *synsets*), each one representing an underlying lexical concept. Synsets are linked by different semantic relations and organized in hierarchies. The synsets identified in WORDNET derive from a thorough lexicographic work and many one-grained sense distinctions are made [53] (there are $99,642$ synsets in version 1.6). In Baziz's system the document is mapped on the SN WORDNET and converted from a set of terms to a set of concepts (*Concept Detection* phase). The extracted concepts (single or multi-words) are then weighted as in the classical index term case, according to a kind of TFxIDF weighting technique which is also a variant of the OKAPI system. By this stage, the document, having been transformed from a set of terms to a set of concepts, is treated by the system.

[17] http://wordnet.princeton.edu/.

Another example is the one shown in [49], where the *SiteIF* system is illustrated: a personal agent for a bilingual news web site that learns user's interests from the requested pages. Even in this case, the system utilizes WORDNET to suggest a word meaning-based document representation, used subsequently to build the user model, along with the extension WORDNET DOMAINS where each term of the lexical database is also labeled with one or more semantic domains, to which it belongs. A document is treated to extract its semantic contents, and is eventually represented through a list of synsets that are relevant for a certain domain. The obtained list is further treated to form a new SN, called *Word Sense Document Representation*, which is the starting point to build the user model.

The last example of such a representation is the WIFS system illustrated in Chapter 6 [51] and in Chapter 2 [35] of this volume. Therein, a document is represented by a set of terms and co-occurrences between them [52]. Fig. 6.6b, in Chapter 6, shows an example of document modeling through simple SN: the document terms that also belong to a database of terms, known to the system, represent the *planets* of the networks, whereas all other terms form the so-called *satellites*, namely the secondary nodes, linked to all the planets with which they co-occur. Even the frequency in the document is calculated for each one of the aforesaid terms. This representation is used to match the document with the user model, in order to assess the document's relevance for that specific user.

5.5.3 Bayesian Networks

A Bayesian Network (BN) is a probabilistic model for Knowledge Representation [60, 61]. It consists of an acyclic direct graph with nodes and arcs linking them. Each node is labeled with a random variable. For example, as illustrated in Fig. 5.10, node o, named *antecedent node* or *parent node*, is connected to node p, called *consequent node* or *child node*. Each arc is associated with a causal relation, expressed by a probability matrix which, for every pair of values V_p and V_q of random variables associated with node p and q respectively, indicates the probability that value V_q occurs once value V_p has occurred, i.e., $P(V_q/V_p)$. The *Evaluation* of the network requires the calculation of the probability of consequent nodes, following a probability distribution of the antecedent nodes. BN are commonly used in adaptive filtering [7, 19, 4] and in IR [5, 15, 64, 63]. In some cases BN turned out to be very useful in improving the retrieval performance, as shown for example in [82].

A document is represented through a network with nodes distributed on two levels (document network): the document is a consequent node, whose previous nodes are the terms contained in the document itself. The weight associated with arcs is calculated through the TFxIDF technique. An example of document representation through this technique is presented in [34], which shows an IR system entirely based on BN. Through the network, the system ranks the document-query similarity. BN are built following these rules:

1. Build the set of terms included in the collection D of all documents.
2. Build the set of possible topics with which the query can be input.
3. Build a BN associated with each possible topic with which the query can be input by the user. Fig. 5.11 illustrates this network. The node *Topic* T_i is associated with

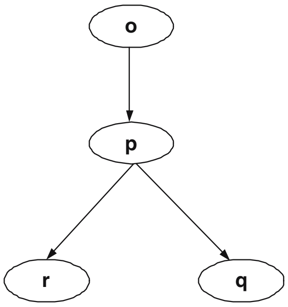


Fig. 5.10. An example of a simple Bayesian Network

event: *the document is relevant for topic* T_i, whereas node t_{ij} is associated with the event *term* t_{ij} *is present in the document*. In order to build the network, each arc (T_i, t_{ij}) is associated with a probability, in automatic or in manual manner.

Once the network is built, the document is represented by the document nodes that form it. Bayes' theorem is used to calculate the query-document similarity:

$$P(T_i/t_{1i}, t_{2i} \ldots, t_{ni}) = \frac{P(T_i)P(t_{1i}, t_{2i} \ldots, t_{ni}/T_i)}{P(t_{1i}, t_{2i} \ldots, t_{ni})} \tag{5.35}$$

In order to simplify the calculation, it is possible to resort to a linear simplification, using function g [75]:

$$g(T_i/t_{1i}, t_{2i} \ldots, t_{ni}) = \sum_k P(t_{ik})w(t_{ik}, T_i) \tag{5.36}$$

where $P(t_{ik}) = 1$ if term $t_{ik} \in d$, otherwise $P(t_{ik}) = 0$ and $w(t_{ik}, T_i)$ equivalent to a weight associated with each term. The query is thus represented as a topic T, and the document as a set of BN nodes. The calculation of the BN through the simplified function g gives the query-document similarity ranking.

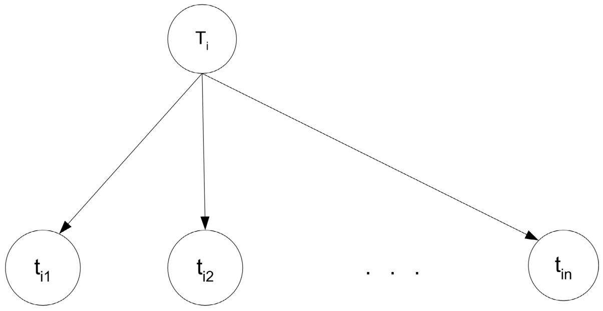


Fig. 5.11. An Example of a Two-Level Bayesian Network Model of IR

Another example of a document representation through BN can be seen in [15], where the *Inquery* system is used, that is an IR system utilizing a document modeling technique based on BN.

Finally, a recent work of B. Piwowarski shows the use of a BN for an IR system on XML-based documents, tested on set INEX of documents [63], whereas in [81] the reader may find another relevant example of Web document representation through a BN.

5.6 Conclusions

In this chapter we discussed the various high level approaches to document modeling and representation. The first approaches to be illustrated derived from the classic IR field, such as the Vector Space Model and its variants, called pre-Web modeling techniques. Another group of techniques followed: they were conceived with the Web, just like those based on HTML tags and hypertext links. The concept modeling was illustrated in the third part, while the final one presented several other approaches inspired by AI techniques, such as Neural Networks, Semantic Networks and Bayesian Networks. The reader will surely have noticed that the Vector Space Model is still very popular, even for the Web, owing to its simplicity and efficacy. These techniques are applied by the various systems described in the rest of the book and by a number of application domains such as Focused Crawling, Content Adaptation, Intelligent Information Search and others. In conclusion, this chapter introduced some of the basic issues related with document modeling in complex and adaptive Web-based systems and applications, providing references to actual uses throughout the rest of the book.

References

1. Agosti, M., Melucci, M.: Information retrieval on the web. In Agosti, M., Crestani, F., Pasi, G. (eds.): ESSIR, Lecture Notes in Computer Science, Vol. 1980. Springer (2000) 242–285
2. In Agosti, M., Smeaton, A.F. (eds.): Information Retrieval and Hypertext. Kluwer Academic Publishers, Dordrecht, NL (1997)
3. Amento, B., Terveen, L., Hill, W.: Does authority mean quality? predicting expert quality ratings of web documents. In: SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (2000) 296–303
4. Asnicar, F., Tasso, C.: ifWeb: a prototype of user models based intelligent agent for document filtering and navigation in the World Wide Web. In P.Brusilovsky, Fink, J., Kay, J. (eds.): Proceedings of Workshop Adaptive Systems and User Modeling on the World Wide Web at Sixth International Conference on User Modeling, UM97, Chia Laguna, Sardinia, Italy (June 2 1997) 3–11
5. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley (1999)
6. Baziz, M., Boughanem, M., Traboulsi, S.: A concept-based approach for indexing documents in IR. In: Actes du XXIII ème Congrès INFORSID, Grenoble, Grenoble, INFORSID (May 24–27 2005) 489–504
7. Belkin, N.J., Croft, W.B.: Information filtering and information retrieval: Two sides of the same coin? Communications of the ACM **35**(12) (1992) 29–38

8. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. Journal of Machine Learning Research **3** (2003) 1137–1155
9. Berry, M.W.: Large-scale sparse singular value computations. International Journal of Supercomputer Applications **6**(1) (Spring 1992) 13–49
10. Bharat, K., Henzinger, M.R.: Improved algorithms for topic distillation in a hyperlinked environment. In: SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, ACM Press (1998) 104–111
11. Billsus, D., Pazzani, M.J.: User modeling for adaptive news access. User Modeling and User-Adapted Interaction **10**(2-3) (2000) 147–180
12. Billsus, D., Pazzani, M.J.: Adaptive news access. In Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.): The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag, Berlin Heidelberg New York (2007) this volume
13. Bishop, C.M.: Neural Networks for Pattern Recognition. Clarendon Press, Oxford (1995)
14. Brin, S., Page, L.: The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems **30**(1-7) (1998) 107–117
15. Broglio, J., Callan, J.P., Croft, W.B., Nachbar, D.W.: Document retrieval and routing using the INQUERY system. In Text REtrieval Conference (TREC) TREC-3 Proceedings, Department of Commerce, National Institute of Standards and Technology (1994) 29–38 NIST Special Publication 500-226: Overview of the Third Text REtrieval Conference (TREC-3).
16. Buckley, C., Singhal, A., Mitra, M.: Using query zoning and correlation within SMART: TREC 5. In Text REtrieval Conference (TREC) TREC-5 Proceedings, Department of Commerce, National Institute of Standards and Technology (1996) NIST Special Publication 500-238: The Fifth Text REtrieval Conference (TREC-5).
17. Buckley, C., Singhal, A., Mitra, M., Salton, G.: New retrieval approaches using SMART: TREC 4. In Harman, D. (ed.): NIST Special Publication 500-236: The Fourth Text REtrieval Conference (TREC-4), Department of Commerce, National Institute of Standards and Technology (November 1995)
18. Budzik, J., Hammond, K.J., Birnbaum, L.: Information access in context. Knowl.-Based Syst. **14**(1-2) (2001) 37–53
19. Callan, J.: Document filtering with inference networks. In Frei, H.P., Harman, D., Schäuble, P., Wilkinson, R. (eds.): Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, ACM Press (August 18–22 1996) 262–269
20. Chakrabarti, S., Dom, B.E., Kumar, S.R., Raghavan, P., Rajagopalan, S., Tomkins, A., Gibson, D., Kleinberg, J.: Mining the web's link structure. Computer **32**(8) (1999) 60–67
21. Chittaro, L., Ranon, R.: Adaptive 3d web sites. In Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.): The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag, Berlin Heidelberg New York (2007) this volume
22. Clark, J., Koprinska, I., Poon, J.: A neural network based approach to automated E-mail classification. In: IEEE/WIC International Conference on Web Intelligence (WI'03), IEEE Computer Society (2003) 702–705
23. Croft, W.B., Belkin, N.J., Bruandet, M.F., Kuhlen, R.: Hypertext and information retrieval: What are the fundamental concepts? (panel). In: ECHT. (1990) 362–366
24. Croft, W.B., Turtle, H.R.: Retrieval strategies for hypertext. Information Processing & Management **29**(3) (1993) 313–324
25. Cummins, R., O'Riordan, C.: Evolving local and global weighting schemes in information retrieval. Information Retrieval **9**(3) (June 2006) 311–330

26. Cutler, M., Deng, H., Maniccam, S., Meng, W.: A new study on using HTML structures to improve retrieval. In: Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'99), Chicago, Illinois, USA, IEEE Computer Society (8-10 November 1999) 406–409
27. Cutler, M., Shih, Y., Meng, W.: Using the structure of HTML documents to improve retrieval. In: USENIX Symposium on Internet Technologies and Systems. (1997) 241–252
28. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for Information Science **41** (1990) 391–407
29. DeRose, S.J.: The SGML FAQ Book : Understanding the Foundation of HTML and XML. Kluwer Academic Publications (1997)
30. Devore, J.L.: Probability and Statistics for Engineering and the Sciences. 3rd edn. Brooks/Cole (1991)
31. Dumais, S.T.: Improving the retrieval of information from external sources. Behavior Research Methods, Instruments and Computers **23** (1991) 229–236
32. Dumais, S.T.: Latent semantic indexing (LSI) and TREC-2. In: Text REtrieval Conference (TREC) TREC-2 Proceedings, Department of Commerce, National Institute of Standards and Technology (1993) 105–116 NIST Special Publication 500-215: The Second Text REtrieval Conference (TREC 2).
33. Frei, H.P., Stieger, D.: Making use of hypertext links when retrieving information. In: Proceedings of the Fourth ACM Conference on Hypertext. Information Retrieval (1992) 102–111
34. Fung, R., Del Favero, B.: Applying Bayesian networks to information retrieval. Communications of the ACM **38**(3) (March 1995) 42–48
35. Gauch, S., Speretta, M., Chandramouli, A., Micarelli, A.: User profiles for personalized information access. In Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.): The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag, Berlin Heidelberg New York (2007) this volume
36. Gentili, G., Micarelli, A., Sciarrone, F.: Infoweb: An adaptive information filtering system for the cultural heritage domain. Applied Artificial Intelligence **17**(8-9) (2003) 715–744
37. Geva, S., Sahama, T.: The NLP task at INEX 2004. SIGIR Forum **39**(1) (2005) 50–53
38. Golub, G.H., Loan, C.F.V.: Matrix Computations. second edn. The Johns Hopkins University Press, Baltimore, MD, USA (1989)
39. Gourley, D., Totty, B.: HTTP: the definitive guide. First edn. O'Reilly Media (September 2002)
40. Gyöngyi, Z., Garcia-Molina, H., Pedersen, J.: Combating web spam with TrustRank. In: Proceedings of the 30th International Conference on Very Large Databases, Morgan Kaufmann (2004) 576–587
41. Hawking, D., Upstill, T., Craswell, N.: Toward better weighting of anchors. In: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Posters (2004) 512–513
42. Haykin, S.: Neural Networks: A Comprehensive Introduction. Prentice-Hall International Editions (1999)
43. Joachims, T., Freitag, D., Mitchell, T.M.: Webwatcher: A tour guide for the world wide web. In: Proceedings of the 15h International Conference on Artificial Intelligence (IJCAI1997). (1997) 770–777
44. Jones, K.S., Walker, S., Robertson, S.E.: A probabilistic model of information retrieval: development and comparative experiments - part 2. Information Processing & Management **36**(6) (2000) 809–840

45. Keller, M., Bengio, S.: A neural network for text representation. In: Proceedings of the 15th International Conference on Artificial Neural Networks: Biological Inspirations, ICANN, Lecture Notes in Computer Science, volume LNCS 3697. Springer-Verlag, 2005. (2005) 667–672
46. Kleinberg, J.: Authoritative sources in a hyperlinked environment. Journal of the ACM **46**(5) (November 1999) 604–632
47. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, MA, USA (1992)
48. Liu, J., Zhong, N., Yao, Y., W.Ras, Z.: The wisdom web: New challenges for web intelligence (WI). Journal of Intelligent Information Systems **20**(1) (2003) 5–9
49. Magnini, B., Strapparava, C.: User modelling for news web sites with word sense based techniques. User Modeling User-Adapted Interaction **14**(2-3) (2004) 239–257
50. Micarelli, A., Gasparetti, F.: Adaptive focused crawling. In Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.): The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag, Berlin Heidelberg New York (2007) this volume
51. Micarelli, A., Gasparetti, F., Sciarrone, F., Gauch, S.: Personalized search on the world wide web. In Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.): The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag, Berlin Heidelberg New York (2007) this volume
52. Micarelli, A., Sciarrone, F.: Anatomy and empirical evaluation of an adaptive web-based information filtering system. User Modeling and User-Adapted Interaction **14**(2-3) (2004) 159–200
53. Miller, G.A.: WordNet: A lexical database for English. Communications of the ACM **38**(11) (1995) 39–41
54. Mobasher, B.: Data mining for web personalization. In Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.): The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag, Berlin Heidelberg New York (2007) this volume
55. Molinari, A., Pereira, R.A.M., Pasi, G.: An indexing model of HTML documents. In: Proceedings of the 2003 ACM Symposium on Applied Computing (SAC), March 9-12, 2003, Melbourne, FL, USA, ACM (2003) 834–840
56. Pant, G., Menczer, F.: Topical crawling for business intelligence. In Koch, T., Sølvberg, I. (eds.): Research and Advanced Technology for Digital Libraries, 7th European Conference, ECDL 2003, Trondheim, Norway, August 17-22, 2003, Proceedings. Volume 2769 of Lecture Notes in Computer Science., Springer (2003) 233–244
57. Park, L.A.F., Ramamohanarao, K., Palaniswami, M.: A novel document retrieval method using the discrete wavelet transform. ACM Transactions on Information Systems **23**(3) (July 2005) 267–298
58. Pazzani, M.J., Billsus, D.: Learning and revising user profiles: The identification of interesting web sites. Machine Learning **27** (1997) 313–331
59. Pazzani, M.J., Billsus, D.: Content-based recommendation systems. In Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.): The Adaptive Web: Methods and Strategies of Web Personalization, Lecture Notes in Computer Science, Vol. 4321. Springer-Verlag, Berlin Heidelberg New York (2007) this volume
60. Pearl, J.: Fusion, propagation, and structuring in belief networks. Artificial Intelligence **29**(3) (1986) 241–288
61. Pearl, J.: Probabilistic Reasoning in Intelligent Systems-Second Edition. Morgan Kauffmann, Los Altos, CA (1988)

62. Pereira, R.A.M., Molinari, A., Pasi, G.: Contextual weighted representations and indexing ieee computer society models for the retrieval of HTML documents. Soft Computing **9**(7) (2005) 481–492
63. Piwowarski, B., Gallinari, P.: A bayesian network for XML information retrieval: Searching and learning with the INEX collection. Information Retrieval **8**(4) (December 2005) 655–681
64. Piwowarski, B., Vu, T., Gallinari, P.: Bayesian networks for structured information retrieval. In: Learning Methods for Text Understanding and Mining, Grenoble, France (January 26–29 2004)
65. Porter, M.F.: An algorithm for suffix stripping. Program **14**(3) (1980) 130–137
66. Quillian, M. In: Semantic Memory. MIT Press (1968)
67. Rijsbergen, C.J.V.: Information Retrieval. Second edn. Department of Computer Science, University of Glasgow (1979)
68. Robertson, S.E., Jones, K.S.: Relevance weighting of search terms. Journal of the American Society for Information Science **27** (1976) 129–146
69. Robertson, S.E., Walker, S.: Okapi/keenbow at TREC-8. In: Text REtrieval Conference (TREC) TREC-8 Proceedings, Department of Commerce, National Institute of Standards and Technology (1999) 151–162 NIST Special Publication 500-246: The Eighth Text REtrieval Conference (TREC 8).
70. Robertson, S.E., Walker, S., Hancock-Beaulieu, M.: Experimentation as a way of life: Okapi at TREC. Information Processing and Management **36**(1) (2000) 95–108
71. Rumelhart, D.E., McClelland, J.L.: Parallel Distributed Processing, Volume 1:Foundations, (ed. w/ PDP Research Group). MIT Press Cambridge, Massachusett (1986)
72. Russel, S., Norvig, P.: Artificial Intelligence: a modern approach. Prentice Hall International (1998)
73. Salem, A.B.M., Syiam, M.M., Ayad, A.F.: Unsupervised artificial neural networks for clustering of document collections. Egyptian Computer Science Journal **26**(1) (2004)
74. Salton, G.: The Smart Retrieval System. Experiments in Automatic Document Processing. First edn. Prentice Hall, Englewood Cliffs (1971)
75. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management **24**(5) (1988) 513–523
76. Salton, G., McGill, M.J.: An Introduction to modern information retrieval. Mc-Graw Hill (1983)
77. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys **34**(1) (2002) 1–47
78. Segal, R.B., Kephart, J.O.: MailCat: an intelligent assistant for organizing e-mail. In Etzioni, O., Müller, J.P., Bradshaw, J.M. (eds.): Proceedings of the Third International Conference on Autonomous Agents (Agents'99), Seattle, WA, USA, ACM Press (1999) 276–282
79. Shastri, L.: Why semantic networks. In Sowa, J.F., ed.: Principles of Semantic Networks: Explorations in the Representation of Knowledge. Morgan Kaufmann, San Mateo, CA (1991) 108–136
80. Thomas, S.: HTTP Essentials: Protocols for Secure, Scalable Web Sites. Wiley (2001)
81. Tsikrika, T., Lalmas, M.: Combining evidence for web retrieval using the inference network model: an experimental study. Information Processing & Management **40**(5) (2004) 751–772
82. Turtle, H.R., Croft, W.B.: Evaluation of an inference network-based retrieval model. ACM Transactions On Information Systems **9**(3) (1991) 187–222
83. Vlajic, N., Card, H.C.: An adaptive neural network approach to hypertext clustering. In: IEEE International Conference on Neural Networks (IJCNN'99). Volume VI., Washington DC, IEEE (July 1999) 3722–3726
84. Walker, S.: The Okapi online catalogue research projects. In Hildreth, C., ed.: The online catalogue. Research and directions. Library Association, London, UK (1989) 84–106

85. Wilkinson, R., Hingston, P.: Using the cosine measure in a neural network for document retrieval. In Bookstein, A.; Chiaramella, Y.; Salton, G.; Raghavan, V.V. (eds.): Proceedings of the 14th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval, Chicago, Ill., USA, ACM Press (October 1991) 202–210
86. Yang, C.C., Chen, H., Hong, K.: Visualization of large category map for Internet browsing. Decision Support Systems **35**(1) (2003) 89–102
87. Yang, K.: Combining text and link-based retrieval methods for web IR. In Voorhees, E., Harman, D. (eds.): The Ninth Text REtrieval Conference (TREC 9) (2001) 609–618
88. Yang, K., Albertson, D.E.: WIDIT in TREC-2003 web track. In: Text REtrieval Conference (TREC) TREC 2003 Proceedings. (2003) 328–336
89. Yang, K., Maglaughlin, K.L.: IRIS at TREC-8. In: Text REtrieval Conference (TREC) TREC-8 Proceedings, Department of Commerce, National Institute of Standards and Technology (1999) 645–656 NIST Special Publication 500-246: The Eighth Text REtrieval Conference (TREC 8).
90. Yao, Y., Zhong, N., Liu, J., Ohsuga, S.: Web intelligence (WI). Lecture Notes in Computer Science **2198** (2001) 1–17
91. Yao, Y., Zhong, N., Liu, J., Ohsuga, S.: Web intelligence: exploring structures, semantics, and knowledge of the web. Knowledge-Based Systems **17**(5-6) (2004) 175–177
92. Zhong, N., Liu, J., Yao, Y.: In search of the wisdom web. IEEE Computer **35**(11) (2002) 27–31
93. Zhong, N., Liu, J., Yao, Y.: A New Paradigm for Developing the Wisdom Web and Social Network Intelligence. In: Web Intelligence. Springer-Verlag, Berlin Heidelberg (2003) 1–15