CAIRO UNIVERSITY
INSTITUTE OF STATISTICAL STUDIES AND RESEARCH, CAIRO UNIVERSITY
EGYPT

AN INTELLIGENT AGENT FOR ARABIC WEB INFORMATION RETRIEVAL

By

Mohamed Ibrahim Eldesouki Tawfik Mohamed

Under the Supervision of

Dr. Mervat H. Gheith

Dr. Waleed M. Arafa

Dr. Kareem Darwish

A Thesis Submitted to the Department of Computer and Information Sciences in Partial Fulfillment of the Requirements for the Degree of MASTER OF SCIENCE In Computer Science

2012

CAIRO UNIVERSITY
INSTITUTE OF STATISTICAL STUDIES AND RESEARCH, CAIRO UNIVERSITY
EGYPT

Approval Sheet

AN INTELLIGENT AGENT FOR ARABIC WEB INFORMATION RETRIEVAL

By

Mohamed Ibrahim Eldesouki Tawfik Mohamed

A Thesis Submitted to the
Department of Computer and
Information Sciences
In Partial Fulfillment of the
Requirements for the Degree of
MASTER OF SCIENCE
In
Computer Science

Approved by the Examining Committee

Prof. Dr. Neveen Mahmoud Darwish ----------

Prof. Dr. Reem Mohamed Reda Bahgat ----------

Dr. Mervat Hassan Gheith ----------

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank all those who have contributed to this thesis, directly or indirectly.

Thank you Dr. Mervat Gheith, I couldn't do such a great work without your support. I would like to express my sincere thanks and gratitude for your constructive criticism, guidance, support, and for sharing your immense wealth of knowledge. I would also like to thank you for providing me the appropriate environment to research, and for being available all the time.

I must thank Dr. Waleed Arafa for believing in my abilities and being supportive all the way. I would like to thank him for his great comments and his continuous encouragements during all the time that I have been under his priceless supervision.

Special thanks go to Dr. Kareem Darwish for his beneficial comments in this thesis.

I would like to thank my parents the most, they have continually supported me. Words cannot express my gratitude and love.

ABSTRACT

Web personalization systems came to differentiate between the users while accessing the Web. However, these systems are still in the development stage.
In this thesis, we develop a personalization system, called *ArabAgent*, to individualize the access into the Web i.e. through personalizing search and filtering news articles. The ArabAgent has the characteristics of an intelligent agent. It depends only on pure content-based approach to personalize the Web and it is mainly concerned with the Arabic content on the Web. ArabAgent is based on content to describe and represent their users' interests and knowledge. It uses documents submitted explicitly as relevant to extract the features that may interest the user. These features are any entities (such as objects, person names, place names, events, concepts, etc) that are mentioned through the text in each document. It uses such data to build a model to the user in the form of a time frame over a semantic network that keeps track of the entities that frequently occur in his documents and entities that frequently co-occur with each other within a 30-day time frame. If the entities didn't occur in the user's feedback documents, the user interest attenuates until it vanishes after 30 days. In Evaluation, the average accuracy of the 30 user profiles of the 10 test days takes about 15 days to converge. We consider only an average of 25 relevant news articles (chosen randomly) of an average of 80 relevant articles. The convergence, definitely, takes less time if more than 25 relevant news articles are chosen and more time if less relevant articles are chosen.

INTRODUCTION

The *World Wide Web, or shortly WWW*, is a human revolution! A revolution in its amount of available information! A revolution in its growth rate of human users! A revolution in the unprecedented diversity of its participants in terms of backgrounds, motivations and languages! A revolution in its possibility of sharing global information! A revolution in the way it makes its participants use their computers and perform their daily tasks!

Despite too much success, these amounts of information in the Web, in addition to lacking a well defined structure, have led to information overload state in which in turn led to production paradox (i.e. the user is overwhelmed with too much information that hampers his productivity instead of helping ease his work). Search engines were invented to alleviate this information overload in accessing the internet. Nonetheless, the diversity in users' languages, attitudes, knowledge, interests, motives and goals makes one way of search difficult to satisfy the needs of about 2 billion[1] users of the web.

Here come the web personalization systems which individualize the user experience to the web. These systems collect data and information about the user and monitor his behavior in order to know his interests and preferences and then anticipate the information in the web.

In this thesis, we develop a personalization system, called *ArabAgent*, to individualize the access into the Web i.e. through personalizing search and filtering news articles. The ArabAgent has the characteristics of an intelligent agent. It depends only on pure content-based approach to personalize the Web and is mainly concerned with the Arabic content on the Web.

ArabAgent is based on content to describe and represent their users' interests and knowledge. It uses documents submitted as relevance feedback from the

[1] http://www.internetworldstats.com/ [last accessed: 4 Aug 2011]

user to extract the features that represent the interests of this user. These features are any entities (such as concepts, objects, persons, place names, etc) that are mentioned through the text in each document. It uses such data to build a model of the user in the form of a time frame over a semantic network.

ArabAgent represents the user interests by keeping track of the entities that frequently occur in his documents and entities that frequently co-occur with each other within a 30-day time frame. If the entities didn't occur in the user's feedback documents, the user interest attenuates until it vanishes after 30 days. Although this could mean that he lost *interest* in some topic, this does not mean he lost the *knowledge* about it. Therefore, ArabAgent represents this knowledge through two things; the existence of the entities that reflect the topic of interest in the semantic network and a weight to represent how much the user is familiar with the topic.

The ArabAgent assists the user while searching the web by tagging the results with color tags indicating the likelihood of relevance of each item in the result. Meanwhile, the ArabAgent ranks again the results asynchronously in a sidebar in the browser to prevent the user from feeling the tardiness of the ranking process. The ArabAgent modifies the user query to be more representative for his need. In addition to personalized search, ArabAgent is used to filter news article from certain news websites.

1. Problem Definition

The tremendous amount of information in the Web has lead to the so-called *information overload*. Although, web search engines are doing great job in satisfying users' needs, the huge number of users and the diversity of their languages, motivations, and backgrounds in unprecedented way make it impossible for one approach of searching to satisfy them all. Web personalization systems came to differentiate between the users while accessing the Web. However, these systems are still in the development stage.

2. Research Objective

In this thesis, we develop a personalization system, named *ArabAgent*, to individualize the access into the Web i.e. through personalizing search and filtering news articles. The ArabAgent has the characteristics of an intelligent agent. The ArabAgent is mainly concerned with the Arabic content on the Web. It depends only on pure content-based approach to personalize the Web.

The ArabAgent assists the user while searching the web by tagging the results with color tags indicating the likelihood of relevance of each item in the result. Meanwhile, the ArabAgent ranks again the results asynchronously in a sidebar in the browser to prevent the user form feeling the tardiness of the ranking process. The ArabAgent modifies the user query to be more representative for his need. In addition to personalized search, ArabAgent is used to filter news article from certain news websites.

3. Research Motivation

There are many works and researches in the topic of Web personalization and adaptive systems. Nevertheless, it is still an ongoing field of research so far and it seems that this field won't reach the stage of maturity in the near future.

One reason that hampers the spread of internet use in Arab world is the limited Arabic content and services on the internet. Unfortunately, compared to other languages, efforts to improve Arabic information search and retrieval performance are limited and modest [Abdulla R., 2007, 2008; Bakry S., 2010; Al-Assi R., 2010]. The problem gets worse especially when you know that 65% of the internet users in the Arab world do not speak English [Abdulla R., 2007]. Accordingly, we intended that this work is a step in the way of supporting Arabic content and services in the Web.

4. Contributions of the Thesis

To our best knowledge, this work is the first that invades the problem of personalizing access to the Arabic content on the web. Other contributions made by this thesis are as follows:

- Representing the user through a model that maintains both his knowledge and interests
- The user's interests and knowledge are modeled using a semantic network of concepts that are found in the relevance feedback documents of the user (not just token or single words as the other systems that uses semantic network do).
- Novel update methods that updates user's profile and takes into consideration the following:
    - Concept weight in the document.
    - Documents count in the previous 30 days.
    - Differentiate between the days by considering day recency.
- Novel way for processing Arabic text that depends on identifying concepts on the text not just the terms of single tokens.
- Taking into consideration the importance of terms in the documents while computing document relevance likelihood in the graph comparison algorithm.

5. Thesis Structure and Overview

The thesis consists of an introduction, seven chapters, and finally a conclusion at the end of the thesis. The introduction defines the problem that this thesis addresses and the motivation and the objectives of the work in the thesis.

Chapter 1 and chapter 2 give a background and the needed concepts that lie under this work. Chapter 1 is divided into three parts; an

introduction about the World Wide Web, the state of the Arabic content and users on the internet and finally a general introduction about the field of information retrieval.

Chapter 2 discusses two matters; it defines the notion of intelligent Agent and its characteristics, and then discusses the web personalization systems.

Chapter 3 is comprised of the related works in the field of web personalization and especially personalized search. It discusses the previous systems in terms of their user models, user profiles, data acquisition techniques, construction of user profiles, updating methods of their user profiles, and personalization techniques. Furthermore, chapter 3 reviews the previous work in processing Arabic documents using stemming techniques.

Chapter 4 concerns our system; ArabAgent. It discusses the general framework of the ArabAgent system and its main components. Also, this chapter proposes the application architecture of the ArabAgent system as well.

Chapter 5 illustrates the techniques and approaches that is used to apply the components of ArabAgent (except for text processing component which is in a separate chapter)

Chapter 6 discusses two things; firstly, it discusses the techniques and heuristics used to prepare the data and statistics from the Arabic Wikipedia project. Secondly, it illustrates the technique used in processing Arabic text in ArabAgent.

Chapter 7 evaluates the ArabAgent system through three axes; the overall system evaluation, text processing component of the system, and stemming techniques for Arabic text.

CHAPTER 1 : THE WEB AND ARABIC LANGUAGE

The *World Wide Web, or shortly WWW*, is a human revolution! A revolution in its amount of available information, A revolution in its growth rate of human users[2], A revolution in the unprecedented diversity of its participants in terms of backgrounds, motivations and languages, A revolution in its possibility of sharing global information, A revolution in the way it makes its participants use their computers and perform their daily tasks!

This chapter discusses three topics in three sections; the Web, the situation of Arabic Language in the Web, and information retrieval especially for Web documents. Section (1.1) demonstrates the great features of the World Wide Web, and how these features have changed our use of computers. The section touched on the increase of the Web in terms of content and users, and the challenges resulted from this increase as well.

Before try helping in improving the performance of Arabic retrieval systems, we need to know what we are dealing with here. Accordingly, section (1.2) offers a survey of estimates of the current state of both the Arab users and Arabic content on the internet as appeared in the literature.

Section (1.3) highlights the peculiarities and characteristics that differentiate the Web from other document collection and that make retrieval system for web documents so special.

---

[2] During the late 1990s, it was estimated that traffic on the public Internet grew by 100 percent per year, while the mean annual growth in the number of Internet users was thought to be between 20% and 50% [Coffman, 1998]. The estimated population of Internet users is 1.97 billion as of 30 June 2010 (http://www.internetworldstats. com/stats.html: last accessed: 4/8/2010)
.

1.1. The Web

The amount of information available on the World Wide Web is massive and continually increases. A number of studies [Bergman M., 2001] show that there were over 550 billion documents on the Web in 2001 this number had been increased in 2008 as Jesse Alpert and Nissan Hajaj, two software engineers working for Google, announced. They have mentioned that Google Search had discovered one trillion unique URLs[3].

The Web pages comprise big portion of these documents in the Web. There were more than 2024 million Web pages in the Web in 2002 as shown in a survey conducted in that year[4]. The survey determines that most of the content was dominated by English language; over 56.4%. The number of Web pages increased as stated by [Gulli A., 2005] to more than 11.5 billion Web pages in 2005. However, Yahoo! Search claims that its index provides more than 19.2 billion web documents in the same year[5]. A more recent statistics in 2011[6], showed that the Web contains at least 50.21 billion pages. See Figure 1.1. These Web pages were the content of over than 346 million websites, occupying more than 130.8 million domains, about three-quarters of them were under the .com domain as stated by the DomainTools[7] website and Netcraft Web Server Survey[8].

[3] http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html [last accessed: 4 Aug 2011]

[4] http://www.netz-tipp.de/languages.html [last accessed: 4 Aug 2011]

[5] http://www.ysearchblog.com/2005/08/08/our-blog-is-growing-up-and-so-has-our-index/ [last accessed: 4 Aug 2011]

[6] http://www.worldwidewebsize.com/ [last accessed: 4 Aug 2011]

[7] http://www.domaintools.com/internet-statistics/ [last accessed: 4 Aug 2011]

[8] http://news.netcraft.com/archives/category/web-server-survey/ [last accessed: 4 Aug 2011]

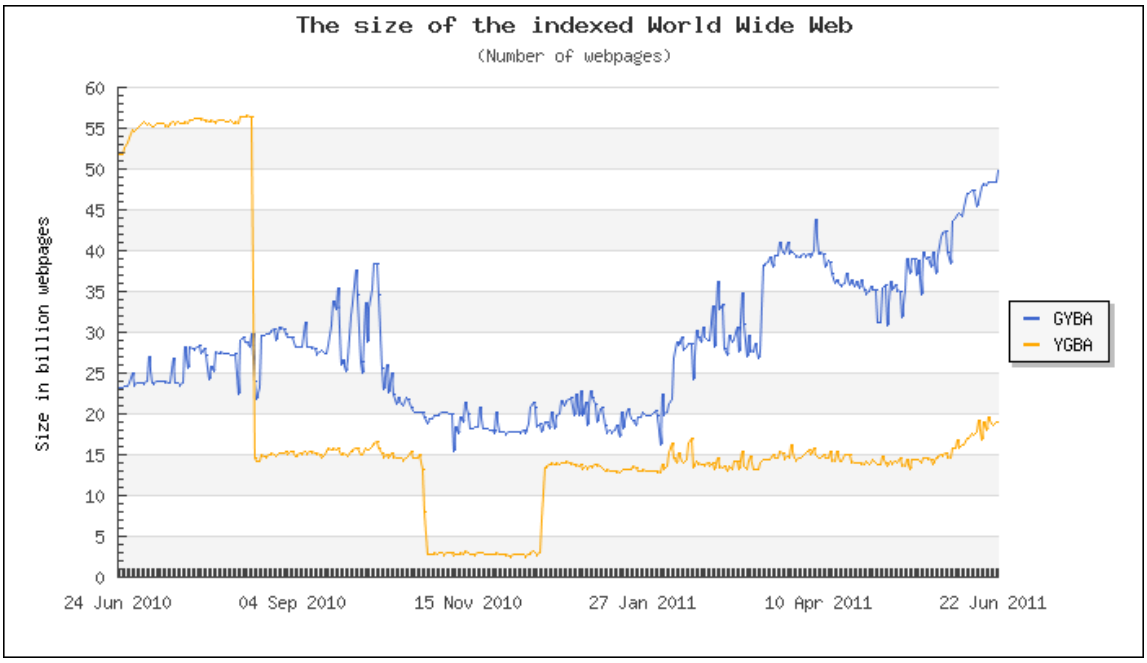

Figure 1.1: The size of the indexed Web pages using two estimates, YGBA and GYBA, as presented in (www.worldwidewebsize.com/)

The WWW[9], or just the Web, "is developed to become a universal repository of human knowledge and culture which has allowed unprecedented sharing of ideas and information in a scale never seen before" [Baeza-Yates R., 1999; Wardrip-Fruin N., 2003]. As a result, "the Web has become a new publishing medium accessible to everybody. Users can create their own Web documents and make them point to any other Web documents without restrictions" [Baeza-Yates R., 1999].

1.2. Arabic on the Web

Reliable statistics about language content and language use on the Internet are very scarce, partly due to the difficulty of assessing such data. Most available data estimates the number of users who are native speakers of particular languages, rather than evaluate the actual use of certain languages and their number of websites. This section offers a survey of estimates of the current

[9] The terms *Internet* and *the Web* are often used in everyday speech without much distinction. However, the Internet and the World Wide Web aren't the same. The Internet is a global data communications system. However, the Web is one of the services communicated via the Internet.

state of both the Arab users and Arabic content on the internet as appeared in the literature.

1.2.1. Arab users and the Web

In 2006, as reported by [Abdulla R., 2007][10], the Arab users of the internet are less than 2% of the total internet users in the whole world. There are about 1.09 billion users of the Internet world wide, whereas 21 million users in the Arab world were estimated to use the internet. The distributions of Arabic internet users in 2006 as a percentage of the population are significant: The numbers are greatest in Kuwait (26.6%), Bahrain (21%), Qatar (20.7%), Lebanon and Morocco (15%), followed by Saudi Arabia and Oman (10%). All other countries in the region are lower, ranging from Iraq (0.2%), Sudan (1.6%) to Egypt (7%), and Algeria (5.8%).

In 2010, the Internet World Stats Web site, reported that the world population is more than 6.8 billion with more than 1.966 billion users are accessing the Internet Worldwide. This is about 28.7% of the world population is accessing the internet. Meanwhile, the population of the Arabic World is about 347 million Arabs; about 5% of the World population. The site estimates that there are more than 65.4 million users of the Internet in Arabic World. This is roughly 18.8% of the Arab are accessing the internet. The percentage of the Arabic users is about 3.3% of the internet users in the whole World. In other words, whereas the Arab world population is about 5% of the World population, the Arabic users are only about 3.3% of the users over the world.

The "Internet World Stats Web site" places the Arabic users among the top ten of other languages' users. The English native users come first with about 27.3% then come Chinese users with 22.6% at the end come the Korean users with about 2%. The Arabic users come seventh before French, Russian, and Korean users. Table 1.1 shows the order of the languages.

[10] http://findarticles.com/p/articles/mi_7081/is_1_27/ai_n28509547/ [accessed: 05/06/2011]

Table 1.1: Number of Internet Users by Language as Internet World Stats Website

Top ten Languages in the Internet	Internet Users by Language	Internet Penetration by Language[11]	Growth in Internet (2000 - 2010)	Internet Users % of Total	World Population for Language (2010 Estimate)
English	536,564,837	42.0 %	281.2 %	27.3 %	1,277,528,133
Chinese	444,948,013	32.6 %	1,277.4 %	22.6 %	1,365,524,982
Spanish	153,309,074	36.5 %	743.2 %	7.8 %	420,469,703
Japanese	99,143,700	78.2 %	110.6 %	5.0 %	126,804,433
Portuguese	82,548,200	33.0 %	989.6 %	4.2 %	250,372,925
German	75,158,584	78.6 %	173.1 %	3.8 %	95,637,049
Arabic	**65,365,400**	**18.8 %**	**2,501.2 %**	**3.3 %**	**347,002,991**
French	59,779,525	17.2 %	398.2 %	3.0 %	347,932,305
Russian	59,700,000	42.8 %	1,825.8 %	3.0 %	139,390,205
Korean	39,440,000	55.2 %	107.1 %	2.0 %	71,393,343
Top 10 Languages	1,615,957,333	36.4 %	421.2 %	82.2 %	4,442,056,069
Rest of the Languages	350,557,483	14.6 %	588.5 %	17.8 %	2,403,553,891
World	1,966,514,816	28.7 %	444.8 %	100.0 %	6,845,609,960

The good news is that Arabic users have the biggest growth rate in the internet in the last decade by about 2501.2% growth rate.

Table 1.2 shows the distributions of Arab users by country in 2011. Note that only six million have access to broadband internet in the Arab world as mentioned by [Al-Assi R., 2010].

Table 1.2: Native Arabic Speakers on the Internet

Country	Population (2011 Est.)	Internet Users Latest Data	Penetration (% Population)	User Growth (2000-2011)
Algeria	34,994,937	4,700,000	13.4 %	9,300.0 %
Bahrain	**738,004**	**649,300**	**88.0 %**	**1,523.3 %**
Comoros	794,683	24,300	3.1 %	1,520.0 %
Djibouti	740,528	25,900	3.5 %	1,750.0 %

[11] The percentage of users from the population

Egypt	**80,471,869**	**17,060,000**	**21.2 %**	**3,691.1 %**
Iraq	29,671,605	325,000	1.1 %	2,500.0 %
Jordan	6,407,085	1,741,900	27.2 %	1,268.3 %
Kuwait	2,789,132	1,100,000	39.4 %	633.3 %
Lebanon	4,125,247	1,000,000	24.2 %	233.3 %
Libya	6,461,454	353,900	5.5 %	3,439.0 %
Mauritania	3,205,060	75,000	2.3 %	1,400.0 %
Morocco	31,627,428	10,442,500	33.0 %	10,342.5 %
Oman	2,967,717	1,236,700	41.7 %	1,274.1 %
Qatar	840,926	436,000	51.8 %	1,353.3 %
Saudi Arabia	25,731,776	9,800,000	38.1 %	4,800.0 %
Somalia	**10,112,453**	**106,000**	**1.0 %**	**52,900.0 %**
Sudan	41,980,182	4,200,000	10.0 %	13,900.0 %
Syria	22,198,110	3,935,000	17.7 %	13,016.7 %
Tunisia	10,589,025	3,600,000	34.0 %	3,500.0 %
United Arab Emirates	4,975,593	3,777,900	75.9 %	414.0 %
Palestine	2,461,267	355,500	14.4 %	915.7 %
Yemen	23,495,361	420,000	1.8 %	2,700.0 %
TOTAL	347,379,442	65,364,900	18.8 %	2,501.2 %

One reason that hampers the spread of internet use in Arab world is the limited Arabic content and services on the internet [Abdulla R., 2007, 2008; Bakry S., 2010; Al-Assi R., 2010]. Especially when you know that 65% of the internet users in the Arab world do not speak English and the Arabic content constitutes only 0.2% of total content of the internet[12]. This problem inhibits prospective users, who do not speak English, from using the internet [Bakry S., 2010].

Internet-censorship which is exerted widely by regimes in the Arab world may play a role in inhibiting people from using the internet. Countries such as Saudi Arabia, Syria, United Arab Emirates and Iraq impose a controlled censorship on the internet both for social and political reasons [13] [Whitaker B., 2010].

[12] http://www.ict.gov.qa/output/page567.asp [last accessed: 05/06/2011]

[13] http://www.guardian.co.uk/world/2009/jun/30/internet-censorship-arab-regimes [last accessed:05/06/2011]

Other reasons impeding internet user penetration in Arab world are high cost of bandwidth and equipment, inadequate telecommunication infrastructures, and surfing the web is often a slow and expensive experience [Abdulla R., 2007, 2008]; there are only six million who have access to high broadband internet in the Arab world.

1.2.2. Arabic Content on the World Wide Web

The Arabic content on the internet has no better state than the Arab users on the Internet. Reports have shown, in their optimistic estimates, that "Arabic-language content on the Internet constituted less than 2% in 2006" as stated in [Abdulla R., 2007][14]. In the same year, "the Egyptian Ministry of Communications and Information Technology [MCIT, 2006] estimated Arabic language content to account for only 0.3% of the total content available on the Web" as mentioned by [Abdulla R., 2008]. However, Hessa Al-Jaber, the Secretary General ICT Qatar, stated that "the number of Arabic web pages relative to the total web pages of the whole world is around 0.2%" as cited in [Bakry S., 2010]. As a result [Al-Assi R., 2010] stated that "Internet giants like Microsoft and Google are eying the region, placing Arabic in their top ten languages in need of prioritized attention."

Ayman Shaban Edakroury characterizes the bad quality of the Arabic content on the internet as the following:

- The Arabic content is shallow and badly organized
- The Arabic content not frequently updated
- Underdeveloped Archiving System in terms of indexing, classification and searching techniques
- Bad design of Web sites; no sitemaps and "bread-crumb trials"[15]
- The rare use of Content Management Systems (CMS)

[14] http://findarticles.com/p/articles/mi_7081/is_1_27/ai_n28509547/ [05/06/2011]

[15] A navigation aid used in user interfaces. It allows users to keep track of their locations within programs or documents.

A report by Professor Saad Haj Bakry is published about the current state of the Arabic internet content, from which future development directions can be concluded. The report tries to evaluate the state of the current Arabic content through conducting a survey questionnaire that was filled in by specialists and influential people in the field of Arabic content while they were attending the 2nd International Symposium on Computer and Arabic Language[16], held in Riyadh, Saudi Arabia in October 2009. The questionnaire was filled in by 74 attendees of the Symposium.

The report has divided the Arabic content on the internet into five sectors. These sectors are the *scientific and educational* sector, *media and entertainment* sector, *conversational and social* sector, *government services* sector, and *banking and business* sector. Each sector has been evaluated in six criteria with five grades of evaluation starting from poor grade and ending with target grade. These criteria are the *overall state of the sector*, the *investment state*, the *awareness state*, the *trust state*, the *state of follow-up and updating*, and *the state of using standards*.

The report shows that "development of the Arabic content is needed in all sectors." It also shows the status of *scientific and educational* sector is in the worst section situation. Although *banking and business* sector has the best status, it is far beyond target level.

Whereas Professor Saad Haj Bakry categorizes the Arabic content of the Internet into five sectors, Ayman Shaban Edakroury [Edakroury A., 2010] divided them into three divisions. These divisions are *business and commercial content*, *public services content*, and *media, entertainment and educational content*.

One example that could view the bad situation of the Arabic content on the internet is the volume and quality of content generated collaboratively by the volunteers in Wikipedia project. A snapshot[17] of the statistics of Wikipedia

[16] http://www.iscal.org.sa/iscal2/ [last accessed: July, 2011]

[17] http://meta.wikimedia.org/wiki/List_of_Wikipedias [last accessed: Dec, 2011]

project, taken in the ninth of June in 2011, highlights the extent of struggling of Arabic language among other languages on the internet. The Arabic Wikipedia project has about 148,574 articles, whereas Polish Wikipedia has over 806,934; a language spoken by only 48 million people[18]. There were about 2193 Active users[19] in Arabic Wikipedia, in contrast to 5307 for Polish Wikipedia. In terms of quality, Arabic Wikipedia has only 146 featured articles[20] of the 148,574 articles whereas Polish Wikipedia has 488 featured articles, note that English Wikipedia has 3295.

1.3. Information Retrieval

One of the main factors of success of the Web is the existence of powerful search engines. Before search engines[21], people had been forced to maintain a list of websites that exist in the Web those days[22]. This was a suitable manner, since the number of websites didn't exceed few hundred at that time. As the exponential growth of websites and web pages, the idea of maintaining websites list is no longer efficient and the necessity of a tool that index the web was inevitable.

Nowadays, search engines such as Yahoo and Google are the main tools for using today's Web. It is clear that the Web would not have become the huge success it is, without search engines.

Many techniques and methods used in the web search trace their origins to the field of information retrieval (IR). Information retrieval

[18] http://www.cactuslanguagetraining.com/us/english/view/the-importance-of-polish-as-a-language-today/ [last accessed: July, 2011]

[19] Users who made at least one edit in the last 30 days

[20] Feature articles are considered to be the best articles in Wikipedia, as determined by Wikipedia's editors. Before being deemed as featured, articles are reviewed at featured article candidates for accuracy, neutrality, completeness, and style according to Wikipedia featured article criteria.

[21] The first search engine was founded in 1993, as showed in http://www.searchenginehistory.com/ [last accessed: July, 2011]

[22] http://www.w3.org/History/19921103-hypertext/hypertext/DataSources/WWW/Servers.html [last accessed: July, 2011]

is “a field concerned with the structure, analysis, organization, storage, searching, and retrieval of information” [Salton G., 1983].

Information Retrieval systems come in three scales; the personal scale, the firm scale, and the global scale [Manning C., 2008]. Each scale has its own models and paradigms as performance change between scales. For example, personal scale is used usually by single user through a personal computer. This raises a specific requirement for such a system. As for capacity of a personal computer the indexing process shouldn’t burden the user while it works. The indexing process could start only when the computer is idle. Also, the system must take care of the second storage space. However, since it’s used personally no need for distribution.

The second degree of scale is the enterprise, and domain-specific search scale. The retrieval task is provided usually for collections such as documents related to a company or a corporation. In this case, the documents will typically be stored on centralized file systems and one or a handful of dedicated machines will provide search over the collection [Manning C., 2008].

The final scale of information retrieval systems is global scale. Examples of such systems are web search engines. Subsections (1.3.4) and (1.3.5) are devoted to the search engines and its collection of documents i.e. the Web. They demonstrate the characteristics that differentiate the Web from other document collections and make web information retrieval so special.

1.3.1. Text Processing

No matter what scale of the information retrieval system is used; all IR systems should adopt text processing techniques to conquer the problems of inflectional and derivational morphology of the language.

As this thesis is concerning Arabic language, it focuses on the difficulties and challenges introduced by Arabic language to information retrieval and natural language processing.

Text operations take place for both documents and queries before indexing and processing, respectively. Text operations are usually divided into four operations [Manning C., 2008]:

1. Tokenization of text, which needs lexical analysis of text in order to convert an input stream of characters into tokens.
2. Stopwords removal step, which removes words with little impact on the retrieval process.
3. Normalization.
4. Stemming and lemmatization.

Other operations could also be considered. Three of them are discussed here. *Selection of indexing terms* operation could be adopted as one step for text preprocessing; conducted after the four previously mentioned steps. For instance, noun words are usually selected to express the documents to be indexed since they more semantically representative than other *Part Of Speech* (POS) such as verbs, adjective, or adverb, see [Baeza-Yates R., 1999]. This process may go under the elimination of stopwords task.

Another operation is the *construction of term categorization structures*, see [Baeza-Yates R., 1999]; categorization structures are built in order to aid later in the process of query expansion which help to improve the overall system performance. An example of such categorization structures are automatic and manual thesauri.

1.3.1.1. Lexical analysis and Tokenization

Manning [Manning C., 2008] defined the tokenization process as, "*Given a character sequence and a defined document unit: tokenization is the job of chopping it up into pieces, called tokens. The tokenization also contains the process of throwing away certain characters such as punctuation*."

The tokenization process depends on the *lexical analysis process* which is "*the process of converting an input stream of characters into a stream of words or tokens*" [Frakes W., 1992]. This process has to

"*treat digits, hyphens, punctuation marks, and the case of letter*" [Baeza-Yates R., 1999; Manning C., 2008].

1.3.1.2. Elimination of Stopwords

As a matter of fact, "a word that occurs in 80% of the documents in the collection is useless for purposes of retrieval" as declared in [Baeza-Yates R., 1999]. These words are usually referred to as *stopwords*. Articles, prepositions, and conjunctions are the typical examples of such words which IR systems preserve lists of them to filter them out while indexing documents or processing queries. Abu El-Khair [Abu El-Khair I., 2006] has examined three examples of Arabic stopwords lists for their effectiveness in information retrieval systems.

In addition to playing an adversarial role in the discrimination of the documents in the retrieval system, the elimination of stopwords reduces the size the index drastically. Baeza-Yates [Baeza-Yates R., 1999] has mentioned that "it is typically to obtain a compression in the size of the indexing structure of 40% or more solely with the elimination of the stopwords".

1.3.1.3. Term Normalization

Sometimes there are many cases where sequences of characters, such as words, are not quite the same but you would like a match to occur. One way to solve this problem is *normalizing text*.

Token normalization, as in [Manning C., 2008], is the "process of canonicalizing tokens so that matches occur despite superficial differences in the character sequences of the tokens".

There are two ways to perform normalization [Manning C., 2008], either by implicitly maintaining *equivalent classes* for token to be normalized, which is the most standard method, or by explicitly providing a relationship between tokens by indexing documents of

both token under each one of them or by later expanding queries that have one or more of the tokens.

In Arabic retrieval system [Eldesouki M., 2009], normalization is done by removing any characters other than the 28 alphabet of Arabic, these include the short vowels (some people use them and other neglect) Kashida (which is used for decoration in some words) and by replacing the "ة" –a letter in Arabic- by "ه" and replacing "ى" by "ي" and replacing "آ", "أ", and "إ" by "ا".

1.3.1.4. Stemming and Lemmatization

For syntactical and morphological reasons, one may use different derivational and/or inflectional forms of a word in the context of a sentence. However, to help retrieval systems increase their recall sometimes it is better to collapse these forms into one class due to their similarity in meaning. The goal of stemming is to reduce derivationally related words. However, lemmatization conflates the inflectional forms of a *lemma*, using a dictionary and morphological analyzer, to return the base or dictionary form of a word, see [Baeza-Yates R., 1999; Manning C., 2008]; as a root in the example of Arabic language.

For example, in Arabic language, if we would like to stem this sentence " وتتم الطباعة باستخدام طابعات ليزر", it could be " تم طباع استخدام طابع ليزر". However, lemmatization would produce the following sentence, "تم طبع خدم طبع ليزر".

Frakes [Frakes W., 1992] distinguishes four types of stemming strategies: affixes removal, table lookup, successor variety, and n-grams. Affixes removal stemming is most intuitive, simple, and can be implemented efficiently.

For English language, there are several suffix removal algorithms. The most popular one is the Porter stemmer [Porter M., 1990]. Other stemmers are Lovins stemmer [Lovins J., 1968], Paice/Husk stemmer [Paice C., 1990], and Krovetz stemmer [Krovetz R., 1993]. For Arabic language, the light10

stemmer of Leah Larkey [Larkey L., 2007] is considered the stemmer with the best performance [Eldesouki M., 2009]. Other stemmers are [Darwish K., 2002b], [Aljlayl M., 2002], [Larkey L., 2002], [Chen A., 2002], [Al Ameed H., 2005], [Nwesri A., 2005], [Kadri Y., 2006], [Nwesri A., 2007], and [El-Beltagy S., 2009].

1.3.2. Query Operations

Many users introduce very short queries i.e. two to three keywords for the web search engines [Manning C., 2008, Büttcher S., 2010]. These short queries could introduce ambiguity to the systems of information retrieval, which in turn reduce the precision of the overall system performance.

Another problem that may face the users while searching the documents is that users may represent their needs by particular keywords and the document collection represents the same need using other keywords, which makes the user spends longer time reformulating his query.

Query operations address these problems. These operations are divided into two categories [Manning C., 2008], namely *local methods* and *global methods.* The local methods depend on the user query and/or the documents returned as a response of the query initially posted by the user to reformulate and expand the query.

Global methods, in the other hand, don't rely on the query of the user or the set of documents that return from the initial request of the user or any collected data from the user. The global methods usually depend on data collected from the document collection in general or an outside resource of knowledge such as thesaurus. These methods are used to aid the user to expand his original query by suggesting new query. This is mostly done by using a thesaurus or based on query log mining (by using the manual query formulations of other users) [Baeza-Yates R., 1999] [Manning C., 2008].

Global methods increase system recall. Another advantage of using global methods is that it doesn't need user intervention. However, it may decrease precision if the query contains keywords for different context. Totally, global methods are less successful than local methods [Manning C., 2008].

Local methods or operations are further divided into three types based on the level of *certainty* of the information collected about the user into *explicit user relevance feedback*, *indirect relevance feedback*, and *Pseudo relevance feedback*.

Explicit feedback methods need a full user intervention by, for example, marking documents which are relevant to his needs as relevant and documents which are irrelevant to his information needs as irrelevant. The main idea consists of (1) selecting important terms, or expressions, attached to the documents that have been identified as relevant by the user, and (2) enhancing the importance of these terms in a new query formulation. The excepted result is that the new query will be moved towards the relevant documents and away from the non-relevant ones. This type of feedback requires direct request of information from the user by asking him directly to provide the information. The advantage of this method is that it makes the information retrieval system using this method certain of his information. However, this method forms burden on the user which makes him neglect providing it usually.

Another type of local methods is *indirect relevance feedback*. This type of feedback collects information about the user without user intervention. The method looks over the user's shoulders. The type collects data such as most visited links, time spent in examining a page, and etc. Although the certainty of this type of feedback is less than the explicit feedback techniques, it provides good information that increases the system performance indeed. Also, this method is

more useful than *pseudo relevance feedback*, discussed later, which contains no evidence of user judgments.

Pseudo relevance feedback is the final method in local methods. Pseudo relevance feedback is sometimes called *blind relevance feedback*. The system automatically chooses top n documents from the result set returned by the initial query and then extracts m words from them. The m words are used to expand the user original query and reweighting its original terms. There are different methods to select the m words from the top n document. This method assumes that the top n documents are relevant. This method is less certain from both methods mentioned earlier in this paragraph [Manning C., 2008].

1.3.3. Text Filtering

In ordinary information retrieval (ad-hoc retrieval) task, the user has at any given time one or more relatively static collections. The collections may be updated, but not so rapidly as to change their basic properties overnight. New documents may come on-line, but they will have the same relatively static characteristics as the existing collections. The user generates many queries against these collections. In other words, the collections are relatively static; the queries are not [Greengrass E., 2000].

In the classification environment, there is no fixed collection. Instead, there is a steady (perhaps high volume) stream of incoming documents. There is a well-defined set of topics of interest, or a well-defined set of users, each with his own well-defined set of interests and concerns. The problem is to classify each document according to which topic(s) it is about or which user(s) the document would interest, and then route the document to the appropriate class(s) [Manning C., 2008]. Documents that are not about any topic of interest are neglected. The set of documents to be classified and routed is not static at all; rather it is constantly changing. Moreover, these documents are not available initially for the purpose of studying their

statistical or other properties. Rather, they will arrive over a (perhaps long) period of time [Greengrass E., 2000]. On the other hand, the set of user needs are presumed to be (relatively) stable, although new needs and users may arrive over time, and old needs become obsolete. Hence, the queries or informational needs are relatively static; the documents are not.

In theory, *classification* problem is identical to the information retrieval problem. Hence, in principle, the same methods are applicable to both problems [Greengrass E., 2000]. It uses almost the same techniques to identify documents that match a specified query or information need. However, the practical differences between the two problems affect which methods are practical for each.

A class need not be as narrowly focused as the standing query; e.g. *multi-core computer chips*. Often, a class is a more general subject area like *China* or *coffee*. Such general classes are usually referred to as *topics*, and the classification task is then called *text classification*, *text categorization*, *topic classification*, or *topic spotting* [Manning C., 2008].

Standing queries and topics differ in their degree of specificity, but the methods for solving routing, filtering, and text classification are essentially the same. We therefore include routing and filtering under the rubric of text classification in this and the following chapters [Manning C., 2008].

1.3.4. The Web Differs

In terms of retrieving and searching, the web has unique peculiarities that differentiate it from other document collections. Web users, documents structure, size and nature, queries submitted into search engines, and other characteristics need special handling different than the traditional information systems.

1.3.4.1. Web Documents

The individual Web pages themselves may be highly complex and structured objects. They often include menus, images, and advertising. Scripts may be used to generate content when the page is first loaded and to update the content as the user interacts with it. A page may serve as a frame to hold other pages or may simply redirect the browser to another page. These redirections may be implemented through scripts or through special HTTP responses, and may be repeated multiple times, thus adding further complexity. Along with HTML pages the Web includes pages in PDF, Microsoft Word, and many other formats.

Many pages are part of the so-called "hidden", "invisible" or "deep" Web. This hidden Web includes pages that have no links referencing them, those that are protected by passwords, and those that are available only by querying a digital library or database. Although these pages can contain valuable content, they are difficult or impossible for a Web crawler to locate.

Along with these benefits come various problems, primarily associated with the relative "quality", "authority" or "popularity" of Web pages and sites because not all writers are professional editors and because many Web pages are actually spam[23] [Manning C., 2008], [Buttcher, 2010].

1.3.4.2. Flat structure vs. Graph Structure

The information retrieval in context of the Web provides us with the benefit of document features that cannot be assumed in the generic context. One of the most important features is the structure supplied by hyperlinks, which gives rise to what is called a *Web graph*. These links from one page to another, often labeled by an image or anchor text,

[23] Malicious pages deliberately posing as something that they are not in order to attract unwarranted attention of a commercial or other nature

provide us with valuable information concerning both the individual pages and the relationship between them [Page L., 1998], [Manning C., 2008], [Buttcher, 2010].

1.3.4.3. Web Users

Another characteristic is the unprecedented relationship between the creators of the web pages (which are the documents of system) and the search engine (which is the information retrieval system). The web pages are in an adversarial relationship with the search engine's operators. The owners of most Web sites wish to enjoy a high ranking from the major commercial search engines, and may take whatever available steps to maximize their ranking. Therefore, they may actively attempt to subvert the features used for ranking by, for example, creating the *spam pages* and making *link spam* and *click spam* to deceive search engines [Manning C., 2008], [Buttcher, 2010].

1.3.4.4. Scale difference in terms of users query and documents

Other problems derive from the scale of the Web - billions of pages scattered among millions of hosts which are needed to be gathered from across the Web by a crawler and stored locally by the search engine for processing. In addition many pages may change daily or hourly, this copy of the Web must be refreshed on a regular basis. While gathering data, the crawler may detect duplicates and near-duplicates of pages, which must be dealt with appropriately [Manning C., 2008] [Buttcher, 2010].

1.3.4.5. Web Search Engines' Queries

Queries are often short [Manning C., 2008] [Buttcher, 2010]. Several studies of query logs from major search engines are summarized by [Spink A., 2004] see [Buttcher, 2010]. Although the exact number differs from study to study, these studies consistently reveal that many queries are just one or two terms long, with a mean query length

between two and three terms. The topics of these queries range across the full breadth of human interests, with sex, health, commerce, and entertainment representing some of the major themes.

The user queries in the Web are ambiguous. Although ambiguity exists in traditional information retrieval systems, it reaches its extreme level in the Web information retrieval [Buttcher, 2010].

Several researchers have examined collections of queries issued to Web search engines in attempt to characterize the user intent underlying these queries. They mentioned the immense volume and variety of queries commercial Web search engines receive [Buttcher, 2010]. [Broder A., 2002] surveyed users of the Altavista search engine and examined its query logs to develop a taxonomy of Web search. He classified Web queries into three categories reflecting users' apparent intent, as follows: navigational queries, informational queries, and transactional queries.

Rose and Levinson [Rose D., 2004] extended the work of Broder [Broder A., 2002] by expanding his three categories into a hierarchy of user goals. They retained Broder's categories at the top level of their hierarchy but renamed the transactional category as the "resource" category. Under both the informational and the transactional categories they identified a number of subcategories.

1.3.5. Bringing Order to the Web

Web retrieval works in two phases. The first phase takes place during the indexing process, when each page is assigned a static rank [Richardson M., 2006]. Informally, this static rank may be viewed as reflecting the quality, authority, or popularity of the page. Ideally, static rank may correspond to a page's prior probability of relevance - the higher the probability, the higher the static rank; see [Büttcher S., 2010].

Static rank is independent of any query. At query time the second phase of Web retrieval takes place. During this phase the static rank is

combined with query-dependent features, such as term proximity and frequency, to produce a dynamic rank.

Assigning a static rank during indexing allows a Web search engine to consider features that would be impractical to consider at query time. The most important of these features are those derived from *link analysis* techniques. These techniques extract information encoded in the structure of the Web graph. Other features may also contribute to static rank, including features derived from the content of the pages and from user behavior.

Several link analysis techniques are used, the most famous one is the PageRank [Page L., 1998; Brin S., 1998]. Other techniques are HITS [Kleinberg J., 1998, 1999] and SALSA [Lempel R., 2000]; see [Buttcher, 2010].

Static ranking provides a query-independent score for each Web page. Although link analysis is an essential component in the computation of static rank, other features may contribute. One feature that is incorporated in the static ranking is the implicit user feedback. Of course, the feedback here is not for personalizing the Web search, however, it is used as a collaborative method to (with the use of other users' feedbacks) to be another feature of estimating the popularities of web pages. The implicit user feedback is used to compute the popularity of a web page but using other methods that involve number of users' visits, and through the frequency and length of their visits, see [Buttcher, 2010]. Another feature is the content of the pages themselves which may contribute to their static rank by considering the quantity and complexity of text, the placement and formatting of graphical elements, and the choice of fonts and colors [Ivory M., 2002]. Finally, the content and structure of a URL may be considered [Upstill T., 2003].

From a practical standpoint it is important to remember that when we refer to a Web graph we are referring to more than just this

mathematical abstraction (1) that pages are grouped into sites, and (2) that anchor text and images may be associated with each link. (3) Links may reference a labeled position within a page as well as the page as a whole. (4) The highly dynamic and fluid nature of the Web must also be remembered; pages and links are continuously added and removed at sites around the world; it is never possible to capture more than a rough approximation of the entire Web graph.

Summary

The amount of information available on the World Wide Web is massive and continually increases. The simplicity of web attracted massive number of users. Different users with different backgrounds, motivations, goals and languages have been attracted to either produce information or consume it. Accordingly, they differ in their needs.

The amount of available information introduced an unprecedented state of information that overwhelmed the user. This state some time referred to as *Information overload.* As a solution of information overload search engines were invented.

All the aforementioned peculiarities of the World Wide Web (i.e. amount of available information, growth rate of human users, diversity of its participants, being the World knowledge repository, etc) could introduce challenges in the domain of Information Retrieval (IR) and Natural Language Processing (NLP) from which further development of the Web could be hampered.

The Arabic language has an extreme importance not only for their native speakers but also for All the Muslims all around the globe. This importance appears vividly in the number of people use Arabic as their first and second language. Arabic users have the biggest growth rate in internet in the last decade by about 2501.2% growth rate.

Arabic language content constitutes about 0.2-0.3% of the total content available on the Web. Development of the Arabic content is

needed in all sectors. The status of *scientific and educational* sector is in the worst section situation. Although *banking and business* sector has the best status, it is far beyond target degree. The reasons that led to the Arabic content situation vary between reasons concerning the quantity and other concerning quality.

The web has unique peculiarities that differentiate it from other document collections. Web users, documents structure, size and nature, queries submitted into search engines, and other characteristics need special handling different than the traditional information systems.

CHAPTER 2 : INTELLIGENT AGENTS AND WEB PERSONALIZATION

Using intelligent agent's paradigm has several advantages over using ordinary object paradigm. Wooldridge [Wooldridge M., 1996] demonstrated this as "such significance is attached to intelligent agents because the metaphor of software as a sophisticated assistant capable of autonomously solving the user's goals is intuitively appealing, computationally powerful, and makes software accessible to non-computer specialists. Another important aspect of this metaphor is that the agents can be personalized to reflect the user's needs, preferences, and constraints."

This chapter is divided into two main sections; section 2.1 devoted to intelligent agent and section 2.2 devoted to web personalization. In subsection 2.1.1, we discuss the controversy over agent definition. Subsection 2.1.2 lists the different characteristics of agents. Subsection 2.1.3 discusses different agent classifications. The significance of intelligent agent is illustrated in subsection 2.1.4. We try to differentiate between agents, programs and expert systems in subsection 2.1.5. Finally, subsection 2.1.6 introduces the multi-agent systems.

Web personalization is considered a typical example for using intelligent agents. Primarily, web personalization systems are systems work on behalf of their users to customize and individualize the experience of accessing the internet.

Section 2.2 is dedicated to web personalization. In it, we describe the necessity of web personalization to overcome the problem of information overload. Subsection 2.2.1 defines the web personalization and depict general framework for web personalization systems. Subsection 2.2.2 presents the different applications of the web personalization. In subsection 2.2.3, we

comprehensively elaborate classifications of different data acquisition techniques based on three features. In subsection 2.2.4 we discuss the different methods to represent the users of the system as user profiles. Two ways to use user's information in building the user profiles are discussed in subsections 2.2.5 and 2.2.6, respectively. Finally, subsection 2.2.7 discusses the different techniques to personalize the experience of the user.

2.1. Intelligent Agent

A broad field in Artificial Intelligence, AI, is called Distributed Artificial Intelligence (DAI). The DAI consists of three areas; multi-agent systems (MAS), Distributed Problem Solving (DPS), and Parallel Artificial Intelligence (PAI). The software agents' domain evolved from multi-agent systems; therefore, it inherits its characteristics, goals and motivations such as speed and reliability [Huhns M., 1994]. The notion of an 'agent' is central to AI. Wooldridge [Wooldridge M., 1995] emphasizes this fact by defining the Artificial Intelligence in terms of Agent concept,"*... the subfield of computer science which aims to construct agents that exhibit aspects of intelligent behavior ...*"

2.1.1. Agent Definition

Many definitions have been introduced by researches for the Agent-based systems; for example [Smith D., 1994] has defined it as "*a persistent software entity dedicated to a specific purpose*"; [Selker T., 1994] has emphasized agents to be "*computer programs that simulate a human relationship by doing something that another person could do for you*"; and [Janca P., 1995] defines an agent as "*a software entity to which tasks can be delegated*". Russell and Norvig [Russell J., 1995] defined agent as, "An agent is anything that can be viewed as perceiving its environment through sensors and acting on that environment through effectors."

Maes [Maes P., 1995] defined the agent as "Autonomous agents are computational systems that inhabit some complex dynamic environment,

sense and act autonomously in this environment, and by doing so realize a set of goals or tasks for which they are designed."

Gilbert has defined agent as [Gilbert D., 1997] "An intelligent agent is software that assists people and acts on their behalf. Intelligent agents work by allowing people to delegate work that they could have done to the agent software. Agents can, just as assistants can, automate repetitive tasks, remember things you forgot, intelligently summarize complex data, learn from you, and even make recommendations to you."

Hermans [Hermans B., 1996] defines agent as "a piece of software that performs a given task using information gleaned from its environment to act in a suitable manner so as to complete the task successfully. The software should be able to adapt itself based on changes occurring in its environment, so that a change in circumstances will still yield the intended result"

After examining some of the previous definitions for agency notion, researchers showed that there is no common agreement upon the definition of exactly what an agent is, [Wooldridge M., 1996] [Nwana H., 1996] [Franklin S., 1996] [Bradshaw J., 1997], or how agents differ from programs [Franklin S., 1996].

The term agent is widely used. However, it faces obstacles for attempts to produce a single definition [Wooldridge M., 1996]. There are two reasons that led up to the difficulty to agree on one definition for the word agent. Firstly, agent researchers do not own this term in the same way as fuzzy logicians/AI researchers. Secondly, the word agent is really a broad topic for different categories of agents' research and development [Nwana H., 1996]. [Bradshaw J., 1997] shows another reason when he emphasizes on that agent can't be defined by its attributes list only but rather defined by both its attributes list and as an attribution from some person. And since the attribution differs from one to another, there hasn't been one sole definition for agency.

Wooldridge [Wooldridge M., 1996] tried to compose a definition that possesses the general aspects of most existing definitions at that time and have defined unspecific notion of an agent as, "*a self-contained program capable of*

controlling its own decision making and acting, which refer to autonomy, based on its perception of its environment, which refer to responsiveness or reactivity, in pursuit of one or more objectives will be used here, which refer to pro-activeness." [Franklin S., 1996] formalized a definition containing these requirements as "*autonomous agent is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future.*" Nwana H., 1996 tries to define it as, "*we define an agent as referring to a component of software and/or hardware which is capable of acing exactingly in order to accomplish tasks on behalf of its user*."

[Bradshaw J., 1997] claims that there are two approaches to the definition of agent. The first approach looks for the agent as ascription made by someone. And the other approach defines agent according to attributes that the agent possess. [Bradshaw J., 1997] summarizes the *ascription definition* of agents by using the words of [Bartlett J., 1980] saying; 'Agent is that agent does'. He emphasizes on the notion that agent can't be defined by its attribute list only but rather consists as an attribution from some person. The second approach that have mentioned in [Bradshaw J., 1997] is definition according the attributes that an agent may hold; or the description definition of an agent. He listed the attributes that an agent may have as of the ones that is enumerated in [Etzioni O., 1995] and [Franklin S., 1996]; Reactive, to simply sense and act; Autonomy, to have a goal and that need to be proactive; Collaborative, to work with other agents; Knowledge-level [Newell A., 1982] communication ability; Inferential capability; Temporal continuity, persistence of identity over long periods of time, can act on abstract task specification; Personality, to show a 'believable' character; Adaptivity, to learn; Mobility, to migrate from host to host by itself.

Some researchers conclude that time and experience will ultimately determine both the meaning and the longevity of the term "agent" [Bradshaw J., 1997]. Also as public exposure to applications increases the term will either come to

mean something that everyone understands because they have seen many examples of it [Wooldridge M., 1996] [Bradshaw J., 1997].

2.1.2. Characteristics of Agent

Although there is no agreement between researchers on what agent is, agent-based systems possess some characteristics that distinguish them from other systems:

Table 2.1: Characteristics of Agent as they appear in the Literature

No.	Attribute	Description	Considered by
1	*Autonomy (autonomous)*	carrying actions out independently	[Foner L., 1993] [Wooldridge M., 1995] [Franklin S., 1996] [Nwana H., 1996]
2	*Personalizability or learning*	the capability of ***learning*** and ***memory***	[Foner L., 1993] [Franklin S., 1996]
3	*Discourse*	may be in the form of a single conversation, or a higher-level discourse between the user and the agent	[Foner L., 1993]
4	*Risk and trust*	putting ourselves to a certain risk that the agent will do something wrong make us compromise between taking the risk and trust the agent	[Foner L., 1993]
5	*Domain*	domain of interest for the agent, i.e. Medical Agent, games Agent, Nuclear System Controller Agent	[Foner L., 1993]
6	*Graceful degradation*	If most of a task can still be accomplished, instead of failing to accomplish any of the task's elements	[Foner L., 1993]
7	*Cooperation or Collaboration (Communicative or Social ability)*	communicates with other agents, perhaps including people	[Foner L., 1993] [Wooldridge M., 1995] [Franklin S., 1996] [Nwana H., 1996]
8	*Anthropomorphism (Represent them visually)*	Pretending to be human	[Foner L., 1993] [Wooldridge M., 1995]
9	*Expectations*	interaction goes better if one's expectations match reality	[Foner L., 1993]
10	*Reactive*	agents perceive their environment and respond, sense and act, in a timely fashion to changes that occur in it	[Wooldridge M., 1995] [Franklin S., 1996] [Nwana H., 1996]
11	*goal-oriented, Proactiveness or deliberative*	have long-term goals, and the ability to planning and taking the initiative	[Wooldridge M., 1995] [Franklin S., 1996][Etzioni O., 1994] [Nwana H., 1996]
12	*Temporally continuous*	acts continually over some period of time,	[Franklin S., 1996]
14	*Mobile*	is the ability of an agent to move around in a network	[Wooldridge M., 1995] [Franklin S., 1996] [Nwana H., 1996]
15	*Flexible*	actions are not scripted	[Franklin S., 1996][Etzioni O., 1994]
16	*Character*	believable "personality" and emotional	[Franklin S., 1996][Etzioni

		state.	O., 1994]
17	*Veracity*	an agent will not knowingly communicate false information	[Wooldridge M., 1995] [Nwana H., 1996]
18	*Benevolence*	not have conflicting goals in addition to try to do what is asked of it	[Wooldridge M., 1995] [Nwana H., 1996]
19	*Rationality*	act in order to achieve its goals plus will not act in such a way as to prevent its goals being achieved	[Wooldridge M., 1995]
20	*Mentalistic notion*	such as knowledge, belief, intention, and obligation	[Wooldridge M., 1995] [Nwana H., 1996]
21	*Emotional agents*	e.g. the agent is getting annoyed from being asked the same thing again and again.	[Wooldridge M., 1995] [Nwana H., 1996]

In [Franklin S., 1996], agent should have four main properties. These properties are *autonomy*, *reactive*, *goal-oriented* and *temporally continuous*. The other properties create a subclass of agents. However, [Foner L., 1993] considers that the first nine properties are compulsory criteria to be an agent.

Wooldridge and Jennings have divided the notion of agency into two types; *weak notion* of agency and *strong notion* of agency. The weak notion of agency is defined as a software (or hardware) that has the properties of *autonomy*, *social ability*, *reactivity* and *Proactiveness*; which are properties 1, 7, 10, and 11. The weak notion of agency is just a natural development of the Object-Oriented programming paradigm. In contrast, strong notion of agency have the properties identified in the weak notion of agency in addition to some characteristics possessed by human such as having mentalistic notion, being emotional agent or could be represented visually; properties 8, 20, and 21. They have mentioned other attributes that may be discussed in the context of agency such as mobility, veracity, benevolence and rationality; properties 14, 17, 18 and 19.

Maes [Maes P., 1994] stated two main problems, which could be thought of as measures to the agent, and have to be solved or observed when building software agents:

- *Competence:* the information about the time to help the user, the tool to help with, and the way to help with.

- *Trust:* to guarantee the user feel comfortable delegating tasks to an agent.

2.1.3. Agent Classification

There are many classification schemes for agents [Franklin S., 1996] such as:

- Classification by the set of properties they own.
- Classification according to the tasks they perform.
- Classification according to agents' control architectures.
- Classification according to the ranges and sensitivity of agents' senses.
- Classification according to the ranges and effectiveness of agents' actions.
- Classification by how much internal state they possess.
- Classification according to the environment in which the agent finds itself.

[Franklin S., 1996] refers to Brustoloni's taxonomy of software agents who classified the agents to regulation agents, planning agents, and adaptive agents. They said that there are many such possibilities for classification.

One of the key ideas of [Franklin S., 1996]'s paper is the natural kind taxonomy of autonomous agents. Recall the biological taxonomy; it takes the form of a tree with "living creatures" at top of the tree and species at the leafs. [Franklin S., 1996] put conceptualization taxonomy for autonomous agent that looks like the biological taxonomy. At the kingdom level there are biological, robotic, or computational agents. At the *phylum level*, they classified the computational agents into software agents and artificial life agents. And at *class level* they classified the software agents into task-specific agents, entertainment agents, and viruses. See Figure 2.1.

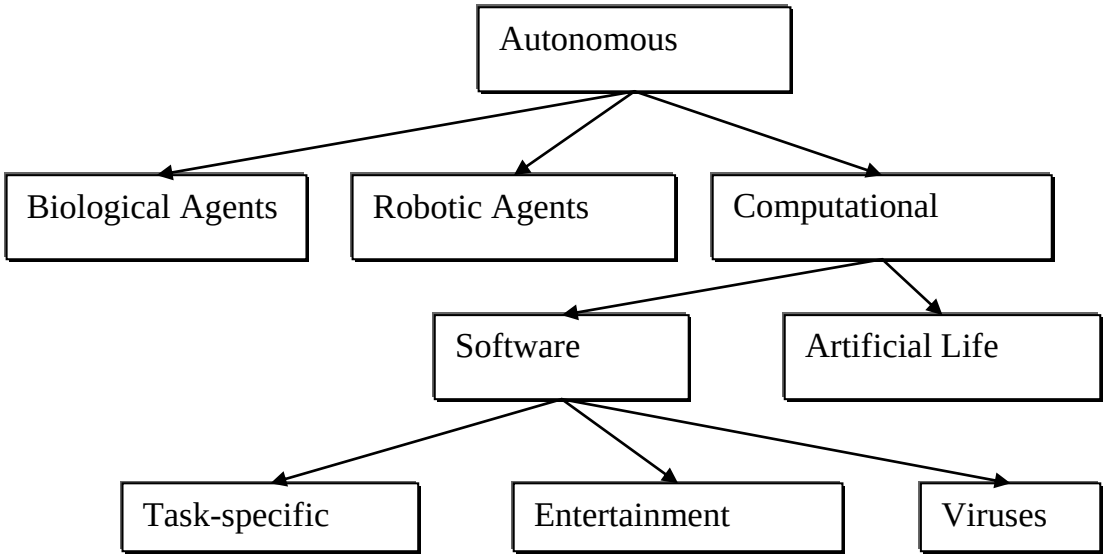

Figure 2.1: A Natural Kinds Taxonomy of Agents

Wooldridge [Wooldridge M., 1995] stated that there are three distinct classes of agent that can be identified:

- "Gopher" agents; execute straightforward tasks based on pre-specified rules and assumptions.
- "Service performing" agents; execute a well defined task at the request of a user.
- "Predictive" agents; volunteer information or services to a user, without being explicitly asked whenever it is considered appropriate.

Nwana [Nwana H., 1996] has found that the number of agents at that time has increased so much. So, he decided to overview the typology of agents. He mentioned that his work complements [Wooldridge M., 1995a] work.
Nwana has made a typology of agents. This typology has different dimensions. These dimensions are:

1. Agent can be *static* or *mobile*.
2. Agent can be *deliberative* or *reactive*.
3. Agents must have at least two of the attributes *autonomy*, *learning* and *cooperation*.
4. Agents may sometimes be classified by their roles.

5. Agents could combine two or more agent philosophies in a single agent. In that sense we called them *hybrid agents*.

6. Agents could have other *secondary attribu*tes such as:
    - Versatile
    - Benevolent or non-helpful
    - Antagonistic or altruistic
    - Veracity: Lie knowingly or always truthful
    - Trustful
    - Degrade gracefully or failing drastically
    - Others are emotional attributes to agents

The third dimension is the primary attributes of the typology. The agent should have at least two of these primary attributes. That means depending on these attributes the agent could be *interface agent*; who has the two attributes autonomy and learning to be the main ones on the agent. If the agent has autonomy and cooperation attributes, as the main attributes, it is called *collaborative agent*. But when it contains the learning and cooperation attributes it would be the *collaborative learning agent*. Finally, the agent may have the three attributes at main, in that sense, it is called *smart agent*. See Figure . [Nwana H., 1996] says that an agent to be *intelligent* it has to be able to learn.

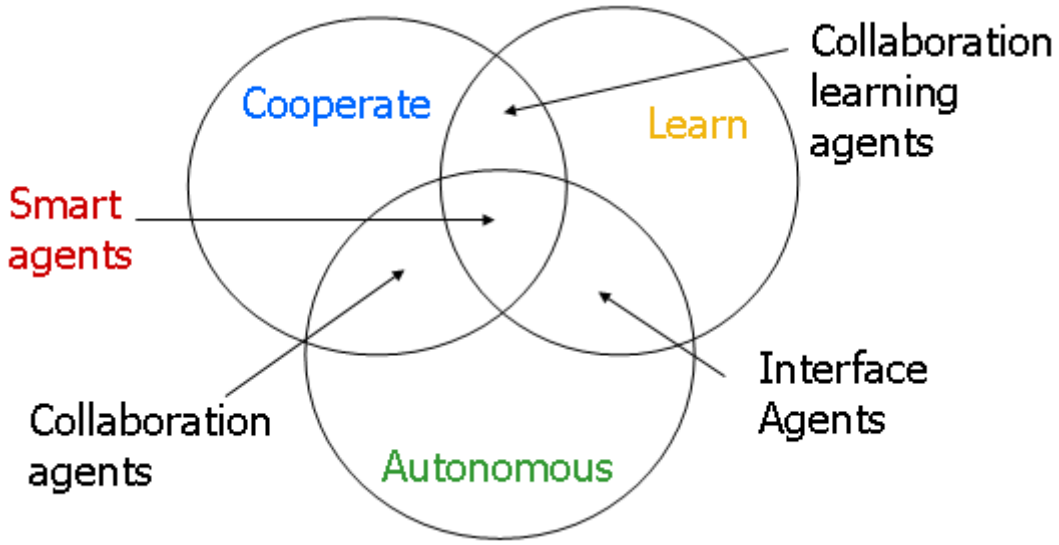

Figure 2.2: Agent Classification Based on Nwana's primary attribute dimension

[Nwana H., 1996] justifies his typology by saying, *"agents exist in a truly multi-dimensional space, which is why if we have not used a two or three-dimensional matrix to classify them – this would be incomplete and inaccurate*" [Nwana H., 1996].

Bradshaw [Bradshaw J., 1997] exhibits different classifications and taxonomies of different researchers. He explained seven different classifications. He included [Shoham Y., 1997; Wooldridge M., 1995] which distinguish between weak and strong notions of agency; agents of strong notions are designed to possess explicit mentalistic or emotional qualities, [Moulin B., 1996], which characterize agents by degree of problem-solving capability, [Gilbert D., 1995], that describes intelligent agents in terms of a space defined by the three dimensions of agency, intelligence, and mobility. He also mentioned Nwana's work [Nwana H., 1996] which is explained earlier in this thesis, [Franklin S., 1996] and their definition for autonomy, "an autonomous agent is a system situated within and part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future," and, finally, he stated the [Petrie C., 1996] research that includes various attempts of researchers to distinguish agents from other types of software.

2.1.4. Why Software Agents

Wooldridge [Wooldridge M., 1996] mentioned two reasons for such significance that is attached to intelligent agents; the capability of being autonomous and being personalized to reflect the user's needs. He justified by "such significance is attached to intelligent agents because the metaphor of software as a sophisticated assistant capable of autonomously solving the user's goals is intuitively appealing, computationally powerful, and makes software accessible to non-computer specialists. Another important aspect of this metaphor is that the agents can be personalized to reflect the user's needs, preferences, and constraints."

Furthermore, Maes [Maes P., 1994] justified the importance of the autonomous property by "The currently dominant interaction metaphor of *direct manipulation* requires the user to initiate all tasks explicitly and to monitor all events. So if we want that untrained users to make effective use of the computer the direct manipulation metaphor should change. In this sense the "Autonomous agents" can be used and this is referred to *indirect management*".

Bradshaw [Bradshaw J., 1997] explained the motivation for the development of software agent into two folds. Firstly, the software agent paradigm attempts to simplify the complexities of distributed computing by hiding the too much details of the task being performed. The other motivation is to overcome the limitations of current user interface approaches.

2.1.5. Programs vs. Expert Systems vs. Agents

Nwana [Nwana H., 1996] considers his primary attributes condition to an agent. To be an agent, it should possess two of the three primary attributes he mentioned; learning, autonomous and cooperate. Other entities are not considered as agents; in other words, no agent has only one attribute of the primary attributes. For example, expert systems are not considered to be agents since neither can learn nor can cooperate (although they are autonomous).

Franklin [Franklin S., 1996] considered that at the extreme high end of the agents' definition are Humans and some animals and at the extreme low end of their definition are the thermostat and a bacterium. A thermostat satisfies all the requirements of the definition. They also noted that software agents are, by definition, programs, but a program must measure up to several marks to be an agent.

2.1.6. Multi-Agent System

When adopting an agent-oriented view of the world, it soon becomes apparent that a single agent is insufficient. Most problems require or involve multiple

agents to represent the decentralized nature of the problem. Various definitions from different disciplines have been proposed for the term multi-agent system. A multi-agent system is defined by Durfee in [Durfee E., 1989] as "a loosely coupled network of problem-solver entities called agents that work together to find answers to problems that is beyond the individual capabilities or knowledge of each agent."

More recently, the term multi-agent system has been given a more general meaning, and it is now used for all types of systems composed of multiple autonomous components showing the following characteristics:

- Each agent has incomplete capabilities to solve a problem.
- There is no global system control
- Data is decentralized
- Computation is asynchronous

2.2. Web Personalization

The term *information overload* has become very clear nowadays. The easy and free publishing to the internet has contributed in spectacular increase in the number of web pages in the internet. Although web search engines have partially resolved this problem, the result of Google search engine contains about 935,000 for Arabic query " موقف المجلس الأعلى للقوات المسلحة من المبادئ فوق الدستورية".

Furthermore, the advent of the new Web 2.0 has encouraged the users to be more involved in the creation of the content of the internet. With the Web 2.0, people are no longer passive consumers. Previous consumer users are now participating in enriching the web through social networks and blog websites such as YouTube, Twitter and Blogger.

On the other hand, As of March, 2011, more than 2 billion people regularly access the internet[24]. Although, they are almost all use the same ways for retrieving information on the Web i.e. same search engines, it is unlikely that 2

[24] http://www.internetworldstats.com/stats.htm [last accessed: July, 2011]

billion people are so similar in their interests that one standardized way of retrieving information fits them all.

This information overload has led to a *productivity paradox* where the increase of the web content has decreased the capability of users to process and consume the data [O'Riordan A., 1995]. The web contains a lot of useful websites and web pages that may benefit the user; but they are scattered among billions all over the internet and search engines are used to rank web pages objectively using link structure of the web and the content of the web pages without being concerned with the user preferences and interests. Accordingly, web personalization has emerged to alleviate the problem of information overload by customizing the user experience of the Web.

Although, it may seem early to define what Web 3.0 is, sometimes Web 3.0 is defined as the web that involves semantic web and personalization techniques[25].

2.2.1. Definition of Web Personalization

Web personalization [Anand S., 2007] can be defined as "any set of actions that can tailor the Web experience to a particular user or set of users."

Web personalization systems are usually used as a component that integrates with other components in a bigger information system.

Six things are involved in a web personalization system; domain of application, user profile representation, data acquisition techniques, filtering or customizing techniques, being adaptive, individual vs. collaborative approach. See Figure 2.3.

[25] http://www.readwriteweb.com/archives/web_30_is_it_about_personalization.php [last accessed: Sep, 2011]

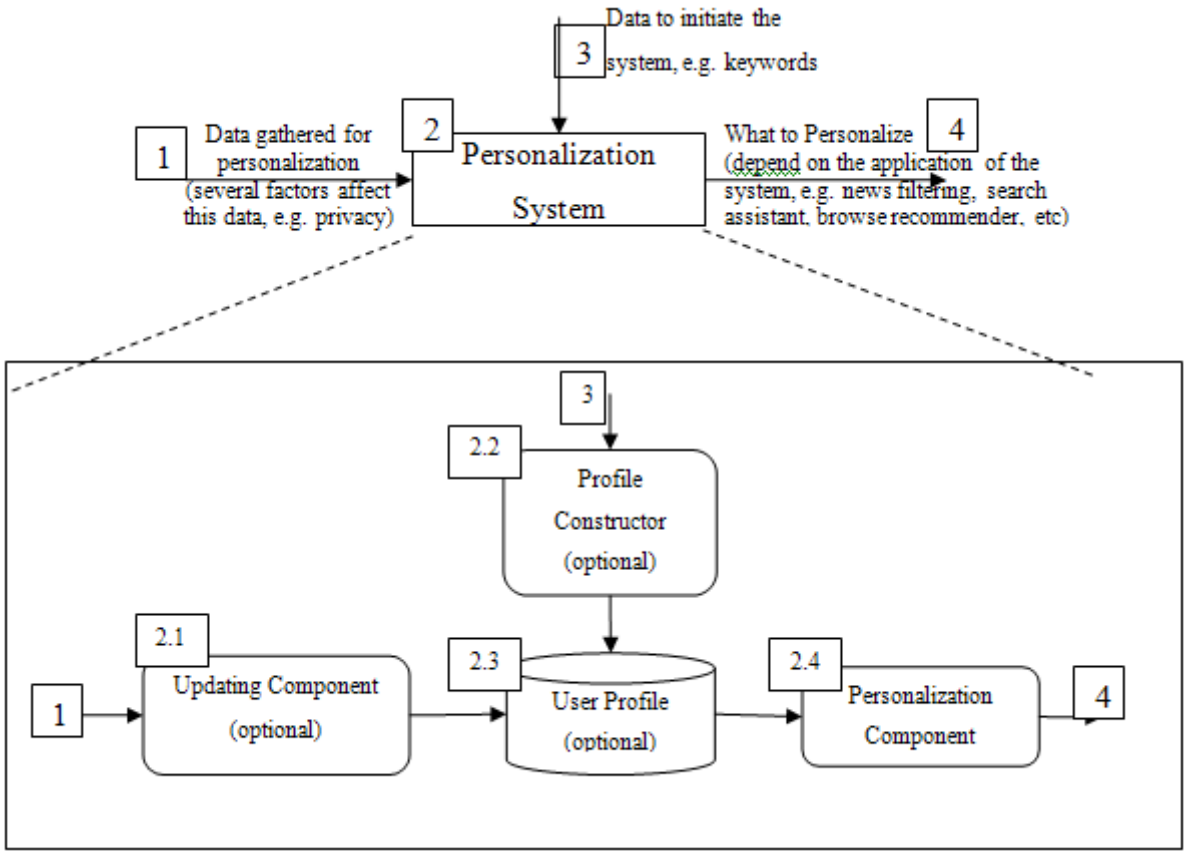

Figure 2.3: General Framework for Personalization Systems on the Web

2.2.2. Application Domains of Web Personalization

Kim and Chan [Kim H., 2006] stated that "Web personalization is used mainly in four categories of applications; *predicting web navigation*, *assisting personalization information*, *personalizing content*, and *personalizing search results*." Systems that predict web pages while user is navigating through the web attempt to predict the next web page the user can visit. Predicting web navigation systems try to anticipate the next user request, subsequently it provides guidance to the user. The role of the assisting personalization information system is to organize user information in the web, thus increase its usability. Personalizing content systems tailor individual pages, site-sessions or entire browsing sessions.

Web personalization systems also could be used to recommend information items such as movies [Hill W., 1995; Good N., 1999; Basu C., 1998],

Purchase, travel and store recommender [Krulwich 1997; Cunningham P., 2001], music [Shardanand U. 1994; Shardanand U., 1995; Hayes C., 1999; Hayes C., 2000], books[26,27], news [Resnick P., 1994; Billsus D., 1999; Lang 1995; Sorensen H., 1995], images [Barrilero M., 2011], web pages, scientific literature[28] (such as research papers etc), Web recommender [Moukas 1997; Balabanovic M., 1997; Minio M., 1996; Asnicar F., 1997; Lieberman H., 1995; Lieberman H., 1999; Mladenic D., 1996; Stefani A., 1998; Pazzani M., 1996; Pazzani M., 1997; Chen L., 1998; Cooley 1999; Armstrong R., 1995; Joachims T., 1997], see [Montaner, 2003].

Personalization systems also could be used in filtering E-mails [Boone G., 1998; Goldberg D., 1992] and web search results [Chen Z., 2000] as well as recommending social elements such as friends, groups, and events (Facebook is a typical example) that are possibly of interest to the user.

2.2.3. Data Acquisition

In order to maintain long-term interests and needs of the user, the acquired information about the previous activities and history from user needs are to be stored. There are three features that characterize the data acquired by the web personalization system; historical data vs. instant data, item data vs. user relevance data, and explicit data vs. implicit data. We illustrate these categories of data in the following subsections.

2.2.3.1. Historical Data vs. Instant Data

Data could be further divided into historical data and instant data. Historical data are gathered and collected for long time to be used in learning and constructing the user profile in the first time to use the system. In the other hand, instant data, which is data collected as user interacts with the system, is used to update the user profile gradually as he/she interacts with the system.

[26] Amazon

[27] http://books.google.com [last accessed: Sep 2011]

[28] http://www.citeulike.org/ [last accessed: Sep 2011]

Current systems that use historical data usually uses techniques like data mining, machine learning, vector space model, etc to construct the profile.

In contrast, other systems which don't use historical data to build a user profile in advance use adaptive methods to update user profile frequency. The adaptive systems have the advantage of coping with the user change in interest, sometimes called interest shifting. However, using only such adaptive techniques makes the system suffer from the problem of cold start.

Using historical data assists the system to avoid the cold start problem, but using it alone makes the system miss the continuous changing and drifting of user interests over time. Also, using instant data alone may suffer from cold start problem, but it keeps track of the drifting in user interests. As a result, a lot of systems combine both types of data for avoiding cold start to detect shifting in user interest.

2.2.3.2. Item Data vs. User Relevancy Data

When web personalization system judges an item as relevant for the user, for instance a web page or a certain audio or video file, this provides two types of data. The first type is the data of the item itself (could be text if the items are web pages for example). This first type is called *item data*. The other type is data used by the system as indication (judge) of relevancy of first type; we called this type the relevancy data of the user (such as the time a user spent in a certain web page). Once the user provides the second type of data (the user relevancy data) (either explicitly or implicitly), that this document is relevant, the system analyses the document to extract the first type of data which is data about the document [Mobasher B., 2005b].

Any item could be of the first kind of data. This data is not confined only to the content of the item, but it could be a preferred search engine or a trusted source of news. Examples of such are web pages, keywords that indicate topics, queries and snippets provided by search engines, URLs, preferred search engines, preferred source of information, or user's part or entire file system. This type of data is usually stored in the user profile for future use. This type of data differs according to whether the system does off-line

learning. For example the system that builds a profile before it starts to serve the user, uses data such as web pages in form of histories (e.g. web search histories, navigation histories, bookmarks).

The user relevancy data, which indicates the relevancy of the previous category of data, needs a *mapping function* to map to either Boolean indicator of relevancy or a weight that indicates the relevancy of the data above. The user relevancy data is useful only in deciding the relevancy of the previous class of data. This data usually is not stored in the profile. Examples of such data are time spent in a certain web page, a mouse movement, scrollbar movement, printing or saving a page, saving as a bookmark, rating an object, marking a hyper link as relevant or irrelevant, etc. As might be obvious, the types of data differ depending on providing the relevancy of first class explicitly or implicitly. It seems that explicit feedback mapping function is straight foreword considering the user decision.

2.2.3.3. Implicit Data vs. Explicit Data

The user relevancy data could be either implicit or explicit. Most of the web personalization systems are seeking to acquire information that represents user implicitly. Implicit methods avoid the intervention of the user to provide relevancy feedback or categories of interest by monitoring his/her behavior. The reason is simply because users avoid providing such data and provision of this data distracts the user from his/her main goal. However, the implicit methods much less certain about the user relevancy of the item data provided. Examples of such methods are time user spent on a page, activities like scrolling, peeking at, maximizing, opening articles in new windows, or saving them to a scrapbook, mouse gestures, or links followed from the currently visited page probably mean a user is interested in that article.

On the other hand, and as one may expect, explicit methods of providing relevancy data of a user have the advantage of being trustworthy since the user provides it voluntarily. However, they sidetrack the user from his pursued goal. Examples of such methods are clicking on categories of Interest, giving a

list of keywords, deciding that a document is relevant or irrelevant manually, rating an object such as a video.

2.2.4. User Representation

The user representation is usually stored in so-called *user models*. A very important aspect of the personalization process is the representation of user data in the user model.

Wahlster and Alfred Kobsa [Kobsa A., 2001] define the user model as *"A user model is a knowledge source in a system which contains explicit assumptions on all aspects of the user that may be relevant to the behavior of the system. These assumptions must be separable by the system from the rest of the system's knowledge."*, see [Heckmann D., 2005].

We here differentiate between two concepts; user profile and user model [Koch N., 2000; Fröschl C., 2005]. User profile stores the raw data of the user. For example, a user profile for a web personalization system could store web pages the user visited, certain domain the user always considers, user queries and other raw data that has not been exposed to analysis (or exposed to a little analysis to avoid the tardiness of analysis task while user request). On the other hand, user models are more sophisticated than user profiles. User model is the result of analyzing the data in the user profile; sometimes is called modeling. Usually, the user profile is used to store the user data in a persistent way for later analysis; for instance the user profile could be a file in a secondary storage. However, user model is essentially used at the application run-time. Therefore, the user model is usually in the memory[29].

Two types of user model are used to represent user in adaptive systems; *dependent* and *independent user models*. Dependent user model is the user model that depends on the application of their systems. Independent user model, sometimes called generic user model, is the user model that is

[29] Sometimes user profile and user model terms are used interchangeably in literature; this is due to that both user profile and user model of the mentioned systems are the same. The system stores the fully analyzed form of the data in the user profile and restores back the data into the user model with no more analysis.

independent from the system application. Generic user model is deemed as a component in the system and is administrated by a so-called *user modeling system*.

Commonly, used user models are still simple to some degree; representing the user as vector of ratings or using a set of keywords. This could be attributed to the system don't know what to represent.

Other sophisticated representations of user model are using ontology structure to build a concept hierarchy for user modeling [Trajkova J., 2004], [Sieg A., 2007a], [Kim H., 2008]. These systems are further categorized to systems that use general reference ontology to represent user and system that builds the concept hierarchy automatically[30].

Other representation uses semantic networks [O'Riordan A., 1995; Sorensen H., 1997], a time window of a certain size for user information [Sugiyama K., 2004] and neural networks [Eliassi-Rad T., 2001]. In addition to representing what the user would like (some systems involve the uninteresting documents in the representation as well to avoid showing their similar documents to the user).

2.2.5. Constructing User Profile

According to section 2.2.3.1, the data could be divided according to the time it is acquired in and used to learn the user profile to data which is acquired and used to build a user profile before the system goes in operation and data which is acquired while the system is online.

User profile construction is an off-line process that is performed before the system gets into operation. The construction process depends on historical data about the user that has been collected before construction.

Generating user profile has the advantage of addressing the problem of *cold start*; sometimes called the *new user problem* [Mobasher B., 2005b]. Systems that depend only on data acquired while operating suffer from bad performance as they initially installed. The system spends some time to collect

[30] Please refer to "previous work review" chapter for more details about different user presentations

information about the user (either explicitly or implicitly) and leverage such data to present item of interest to the user.

Some systems consider user profile construction as data mining process [Eirinaki M., 2003] [Mobasher B., 2007] [Anand S., 2007]. Here, the discovered knowledge is the user profile. Anand and Mobasher describe this as "Efficient and intelligent techniques are needed to mine this data (the historical data) for actionable knowledge, and to effectively use the discovered knowledge to enhance the users' Web experience."

There are several techniques used to build the user profile. One could classify them according to the representation of the user model; for example, techniques used to build concept hierarchy user profiles [Chaffee J., 2000], [Ramanathan K, 2008], and [KIM H., 2003], [KIM H., 2008], techniques used to build semantic networks profiles [O'Riordan A., 1995; Sorensen H., 1997] or techniques used to build several (unrelated) topic profiles. Furthermore, these techniques could be further categorized, for example, the techniques to build concept hierarchy user profiles could be divided into techniques that use pre-existing reference ontology to build their profiles and techniques that build such profile automatically. They, also, could divided into techniques that build concept hierarchy that reflect general concepts and other that build profiles that their concepts reflect personal tailored concepts (that don't reflect real-life concepts). Examples of such systems are discussed in great details in chapter 4; the previous work.

Systems that used the reference ontology to build their users' profiles suffer from the following disadvantages:

1. The web personalization is limited to languages of the category hierarchy,
2. They only use few top levels of categories in the ODP hierarchy, this makes the user profiles do not cover the low-level categories, which are more specific. Consequently, this may reduce the ranking quality for individuals with more specific interests, not represented as high-level categories.

3. When using the full category hierarchy to represent user profile, the large number of nodes will be built this slows down the search process.

4. Using an existing hierarchy can make the user profile contain many irrelevant categories since all high-level categories are in the user profiles.

5. (in representation) mostly the profile does not represent the actual user interests since the user's interests are overlapped but the category hierarchy which consists of isolated categories.

2.2.6. Updating User Profile

User interests and knowledge are changing over time. Several web personalization systems attempt to address this problem by continually updating their user profiles.

Systems that don't have a technique for updating user profile soon become far away from the user's interests and preferences.

As user profile construction techniques, user profile updating techniques depend on the representation of the user profiles.

Other systems realized that the user interests and knowledge changes in different paces; some interests are transient which last for one session of search or browsing, other interests are permanent. There are several explanations of the different paces. For example, if a user is a programmer, a lot of his activities on the web would reflect this interest (he searches about algorithms or information about certain features of a programming language). However, he may be interested in a topic that is more specialized of the programming topic for a while. For example he may be interested in multi-threading programming in the java programming language. Definitely, this interest in the multi-threading programming would last for a period shorter than the programming topic itself. A lot of researches assumed that general topics usually reflect long-term user interests. In the contrary, specialized topic which is a sub-class of the general topics are usually more ephemeral interests. Examples of such systems are discussed in chapter 4.

Other systems assume a window of fixed time (for example 30 days as in [Sugiyama K., 2004]) to reflect the user changing interests.

2.2.7. Filtering and Personalization techniques

Personalization Techniques, sometimes *called Ranking/Filtering scheme*, identify the way the user profile is used to individualize the experience of the system application. In other words, it describes the task of mapping from user profile to the mentioned final form of personalization. Simply, it is how the system uses the profile to, for example, rank the web search results, filter news, provide useful web links, recommend video to user, expand user query while web search, etc.

Summary

Although there is no agreement between researchers on what agent is, agent-based systems possess some characteristics that distinguish them from other systems such as *Autonomy, learning, Discourse, Risk and trust, Domain, Graceful degradation, Reactive, Proactive, Communicative.*

Nwana assumes that agents should possess at least two of the attributes *autonomy, learning* and *cooperation* and the agent that possesses learning attribute called *intelligent agent.*

Agents could be classified according to different criteria such as the set of properties they own, the tasks they perform, agents control architectures, the ranges and sensitivity of agents senses, the ranges and effectiveness of agents' actions, how much internal state they possess, or the environment in which the agent finds itself.

There are two reasons for such significance that is attached to intelligent agents; the capability of being autonomous and being personalized to reflect the user's needs.

Web personalization has emerged to alleviate the problem of information overload by customizing the user experience of the Web.

Six things are involved in a web personalization system; domain of application, user profile representation, data acquisition techniques, filtering or customizing techniques, being adaptive, individual vs. collaborative approach.

In order to maintain long-term interests and needs of the user, the acquired information about the previous activities and history from user needs to be stored. There are three features that characterize the data acquired by the web personalization system; historical data vs. instant data, item data vs. user relevancy data, and explicit data vs. implicit data.

The user representation is usually stored in so-called *user models*. A very important aspect of the personalization process is the representation of user data in the user model.

User profile construction is an off-line process that is performed before the system gets into operation. The construction process depends on historical data about the user that has been collected before construction.

User interests and knowledge are changing over time. Several web personalization systems attempt to address this problem by continually updating their user profiles.

CHAPTER 3 : LEADING AND SIMILAR WORK REVIEW

In this chapter we are going to review the previous work. The chapter is divided into two sections; the first section highlights the most related and leading work of ArabAgent system. The second section reviews the work that has been done in the domain of text processing of Arabic information retrieval (especially stemming techniques) in the past two decades.

However, due to the tremendous number of existing web personalization systems, more attention is being paid to the work related to our system; ArabAgent. Web personalization systems are going to be examined in terms of their application (especially news filtering and web search), profile representation, personalization technique, and user feedback.

Most of the work mentioned in the text processing section, section (3.2), has been compared and evaluated against ArabAgent system technique for text processing in chapter 7.

The section of web personalization work, discusses the work in a vertical way, it doesn't discuss each system alone, but discusses the various ways of representation, then the various ways of data acquisition, and so on.

3.1. Web Personalization

As been mentioned early, the systems that are mentioned here are closely related to our system. In case you may want to check more systems, there are many useful surveys that mention the leading systems in the area of web personalization [Mobasher B., 2005b; Godoy D., 2005; Jansen B., 2006].

3.1.1. User Profile Representation

Different presentations and user models have been introduced in the literature. In this section we are going to highlight the most prominent user models.

3.1.1.1. A Hierarchy of Concepts Representation

Many personalization systems represent the user as *a hierarchy of concepts or categories* (the concepts sometimes called nodes); this representation could be dubbed other names such as taxonomy or ontology. Examples of such representation are [Pretschner A., 1999a] [Pretschner A., 1999b], [Kurki T., 1999], [Chaffee J., 2000], Persona, [Chen C., 2002] , [KIM H., 2003], [KIM H., 2008], [Trajkova J., 2004], [Challam V., 2004], [Speretta M., 2005], [Chirita P., 2005], [Sieg A., 2005], [Sieg A., 2007a], [Ramanathan K, 2008], [Zhu D., 2008], and [Xu Y., 2007]. Presumably, the concepts or nodes represent user's interest; each node represents one user interests. As a result there should be a huge number of concepts to cover most users.

Some representations have their nodes attached with one or several "attributes" or "tags". For example, [Dai H., 2006], [Ramanathan K, 2008] have transactional or recreational attributes attached with each of their nodes, [Ramanathan K, 2008] has recency of the node, [Pretschner A., 1999a], [Pretschner A., 1999b] attached to attributes the time spent by the user to any page to the node belong to it and length of the page.

The system in [Pretschner A., 1999b] adopted a concept hierarchy comprised of 4,300 nodes each node is represented as a weighted keyword vector. The system tries to be unique by making different interests being kept different; there is no average as in other approaches that use the vector space model.

In [Chaffee J., 2000], the profile is a mapping file. The mapping file contains mapping between reference ontology and a personal ontology. Each concept in the personal ontology has many concepts from the reference ontology and opposite is not right; every concept in reference ontology is mapped under only one concept in the personal ontology. The concept from the reference ontology associated with it the matching weight with concept of personal ontology that it has been mapped to.

The user profiles in both [Trajkova J., 2004] [Challam V., 2004] and [Speretta M., 2005] use the Open Directory Project (ODP) concepts hierarchy as the reference ontology. They all used only the top 3 levels of the concept

hierarchy only. In [Trajkova J., 2004], the system further neglects concepts associated with less than 30 pages. The total number of concepts in [Trajkova J., 2004] is 2991 concepts. However, the total number of concepts in [Speretta M., 2005] is about 1869. In addition to representation of the other two profiles, the user profile in [Challam V., 2004] is as user's contextual profile within a time window as a weighted ontology. The weight of the concept in the ontology represents the amount of information recently viewed or created by the user that was classified into that concept.

Also, [Sieg A., 2007a], [Sieg A., 2007b], [Sieg A., 2007c] use ODP as a reference ontology. There is no mention of using certain number of levels. The user profile represents a concept hierarchy. Each concept in the hierarchy is represented as a weighted term vector. This vector represents the semantic lied under the web pages of the corresponding category in the reference ontology as well as all the subcategories of the corresponding category. Associated with each concept an attribute its value determines user degree of interest to the concept. The user profile in [Sieg A., 2007a] is used mainly to keep track of the long-term interests.

The system adopted by [Chirita P., 2005] represents the user profile as set of topic connected with each other using generalization/specialization relationships. For example,

/Arts/Architecture/Experimental/
/Arts/Architecture/Famous_Names/
/Arts/Photography/Techniques_and_Styles/

In [KIM H., 2003], [KIM H., 2008] and [Kim H., 2006] system, the profile is represented as a hierarchy of nodes. Each node consists of several terms in which leaf nodes are considered more specific (to keep track of short-term interests) whereas the parent nodes are more general (to consider the long-term user interests). Near the root of [KIM H., 2003], [KIM H., 2008], general nodes are represented by larger clusters of terms; while towards the leaves more specific nodes are represented by smaller clusters of terms. The root node contains all distinct terms in the documents used to construct the profile.

In [Sieg A., 2005], the user profile is concept hierarchy transferring into an *induced lattice* by adding *user's information contexts*. The concept hierarchy is constructed by crawling Yahoo! concept hierarchy. Each node of the hierarchy is a vector of term weights that represents the web pages associated to the corresponding category in Yahoo! hierarchy and the subcategories of those categories as well. As user selects and deselects categories while browsing or searching activities, the hierarchy is transferred into induced lattice by adding context as concepts into the hierarchy. The context concepts are considered the user's long-terms interests. However, the user's original query is considered as the short-term interests.

3.1.1.2. A weighted Semantic Networks Representation

Systems such as [Asnicar F., 1997], [O'Riordan A., 1995], [Sorensen H., 1997], [Stefani A., 1998] and [Cesarano C., 2003] have chosen to represent the user using semantic networks. Semantic networks represent terms and their context by linking nodes with arcs which represent co-occurrences in some documents (It is worth to mention that some system, such as [Asnicar F., 1997], use these networks to represent disinterests as well). Usually semantic networks are used to represent user interests.

The INFOrmer [O'Riordan A., 1995; Sorensen H., 1997] system represents user interests through representing the documents the user is interested in. Each document is represented as a semantic network. After determining those documents, [O'Riordan A., 1995], [Sorensen H., 1997] builds the user semantic network from documents. The analysis of each document considers the context of terms occurring in the text rather than just their frequency of occurrence.

The [Stefani A., 1998] system uses semantic net as a user model to represent user interests. Every node is an interesting word and arcs between nodes are the co-occurrence relation of two words; every node and every arc has a weight that represents a different level of interest for the user. The weights are periodically reconsidered and possibly lowered overtime. In addition, the less useful nodes and arcs are removed from the model.

3.1.1.3. Other Representations

Sugiyama [Sugiyama K., 2004] has presented the user preferences as a vector of keywords. The vector is built from a user profile of web pages visited in the last 30 days. The profile is divided into two parts: the first part reflects the long-term interests and consists of a user defined window of last 30 days without the current day. Each day consists of the web pages that considered relevant to user in that day.

The second part represents the short-term interests and also divided into two parts today's sessions without current session and current session. Each session comprises of Web pages that are considered relevant to the user in that session.

Some systems maintain the profile in the form of a single vector of keywords. The vector represents the keywords of the interesting documents for the user. The vector could be Boolean [Yan T., 1995], weighted as in [Kamba T., 1995], [Sakagami H., 1997], and in Latizia [Lieberman H., 1995], [Lieberman H., 1997], or frequencies [Meng X., 1999] of keywords. This vector is maintained to provide a context to user information access.

In some other systems, several vectors of keywords represent several user profiles [Sheth B., 1993], [Sheth B., 1994], [Moukas A., 1996], [Moukas A., 1997], [Balabanovic M., 1997], [Parent S., 2001]. In [Chen L., 1998] the vectors represent clusters' centers gained from clustering web pages. In addition to clusters' centers, the profile representation comprises the associated documents to the clusters.

Later systems tried to improve the last representation by involving the uninteresting documents in the representation. Their profiles comprise *two classes*; "interesting" and "not interesting" [Mladenic D., 1996] and [Casasola E., 1998]. For [Mladenic D., 1996], if the user decides the status of the document as interesting or not interesting (either explicitly or implicitly), the document is associated to the equivalent class. In [Casasola E., 1998], documents are stored as weighted keyword vectors, and for both classes, every term is assigned a weight representing its membership to "its" class.

Other systems tried to differentiate between different interests by preserving set of disjoint classes (no relationship between them and they don't overlap) represents the interests of the user. For example, In Personal Wall Street Journal (www.wsj.com) and [Chesnais P., 1995] classes are companies that the user has a share in it, see [Pretschner A., 1999]. In addition, the classes in [Pazzani M., 1996] represent or describe the content of the index page. Within each class, the profile consists of Boolean keyword vector. In [Rucker J., 1997] and [Montebello M., 1998] systems, the profile is bookmark directories. The directories in the bookmark represent the user's classes of interest. In [Montebello M., 1998] and [Liu F., 2004], documents contained in these classes are stored as weighted keyword vectors.

In PSUN system [Sorensen H., 1995], recurring words in the documents of relevance feedback are stored by means of n-grams, and the n-grams are stored in a network of mutually attracting or repelling words. The degree of attraction is determined by the degree of co-occurrences. Different user profiles are then stored in a way similar to Minsky's K-lines, connecting n-grams of different weights. Each user has multiple profiles that compete via a generic algorithm as in [Sheth B., 1994], see [Pretschner A., 1999]. Furthermore, neural networks are used to represent the user in WAWA [Eliassi-Rad T., 2001].

3.1.2. Data Acquisition

Section 2.2.3 of chapter 2 has discussed the three features that characterize data acquisition in the system; item data vs. relevancy data, implicit data vs. explicit data and for batch learning data vs. data for frequent updating. This section discusses the various systems that use such methods. Each system is going to be examined for the three features.

Personal Wall Street Journal provides explicit relevance feedback technique; see [Pretschner A., 1999] by clicking on Categories of Interest, it also provides implicit feedback using the user's stock portfolio to propose links to follow or articles to read that are related to the shares contained in the portfolio.

Explicit relevance feedback is also provided by giving a list of keywords to choose among them as in [Sheth B., 1993], [Sheth B., 1994] [Kurki T., 1999], [Armstrong R., 1995], or deciding that a document is relevant or irrelevant manually [Sheth B., 1993], [Sheth B., 1994] [Moukas A., 1996].

Queries and snippets in [Speretta M., 2005] are the items data that are used to build and update the profile, which are provided by search engines. The system considers the relevancy of snippets as the user follow their links. In [KIM H., 2003], [KIM H., 2008], [Sieg A., 2007a] and [Parent S., 2001], Web pages are the source of item data. In [Sieg A., 2007a] a web page is deemed according to the frequency of visiting it. [KIM H., 2003] and [KIM H., 2008] uses web pages in the user's bookmarks. In [Parent S., 2001] and [Chirita P., 2005], the user provides categories or topics of interest. [Parent S., 2001] considers clicking on or writing categories of interest.

In [Liu F., 2004], the item data is the user's search history and must be provided in a particular tree structure of three levels. This history data is used for off-line learning. The root of the tree structure, level 0, is the query submitted by the user, level 1 is the prospective categories that query may belong to. Level 2 is the web pages that visited by the user and belong to one of the categories in level 1.

System in [Meng X., 1999] used the user entire file system to learn its web personalization system. [Challam V., 2004] used MSN Messenger Documents and MS-Office documents. In [Zhu D., 2008], desktop computer information is used, indexed by *Google Desktop Search SDK* to initialize the user profile.

The time that user spent on a page is used by [Konstan J., 1997] and [Zhu D., 2008] to indicate relevancy since they show that there is a strong correlation between the time spent on a page and the actual user interest. This approach is also investigated in [Morita M., 1994] [Nichols D., 1997] [Oard D., 1996], [Parent S., 2001], [Sieg A., 2007a], with some modifications, and is also implemented in [Pretschner A., 1999a] and [Pretschner A., 1999b].

[Stefani A., 1998] tracks down the user's browsing behavior (e.g. following links) and trying to anticipate what documents in the site could be interesting for the user.

In [Speretta M., 2005], a wrapper is used around the Google search interface to monitor user's activities to collect information such as queries submitted, results returned (titles and snippets), and results selected by user.

In implicit feedback, monitoring user behavior is the most dominant way. For instance, when the user selects a Web page in an attempt to satisfy his/her information need from search engine results [Sugiyama K., 2004]. In addition, the user may access more Web pages by following the hyperlinks [Mladenic D., 1996] [Sugiyama K., 2004] on his/her selected Web page and continue to browse [Lieberman H., 1995], [Lieberman H., 1997], [Stefani A., 1998].

The user behavior is tracked also while he reads activities like scrolling, peeking at, maximizing, opening articles in new windows, or saving them to a scrapbook probably mean a user is interested in that article [Kamba T., 1995], [Sakagami H., 1997], or spending sometime on a page [Konstan J., 1997] [Morita M., 1994] [Nichols D., 1997] [Oard D., 1996]. Also [Trajkova J., 2004] have used the time the user spent in a web page as a judgmental behavior of the relevancy. The web page is considered relevant if the user spends 5 seconds at least on the page and the page size is 1 KB or more.

Bookmarking a page also means this page is interesting [Lieberman H., 1995], [Lieberman H., 1997], [Rucker J., 1997], [Thomas C., 1997] and [McGowan J., 2002]. [Rucker J., 1997] uses both the structure of the bookmark as well as the web pages inside to build the system.

As (Western) users tend to read from the top left corner to the right bottom corner, links that are omitted during the reading process might express disinterest in the referenced document [Lieberman H., 1995; Lieberman H., 1997].

3.1.3. User Profile Construction

This subsection discusses the different techniques used by web personalization systems to build user profiles using the offline data prior to system operation.

Techniques that are used to update the profile during operation are discussed subsection 3.1.4.

3.1.3.1. Building Concept Hierarchy Profiles

The techniques that have been used to build concept hierarchy profiles could be classified according to several criteria. It could build user profile using a pre-exist reference ontology such as ODP, WordNet or Wikipedia, or without using one. Another criterion is whether the nodes reflect general concepts or personally identified concepts. Examples of techniques that does not use pre-exist reference ontology are [Chaffee J., 2000], [Ramanathan K, 2008], [KIM H., 2003], and [KIM H., 2008]. Examples of system that generate personally identified concepts are [KIM H., 2003], [KIM H., 2008]. Examples of systems that have used an existing reference ontology to build their profiles are [Pretschner A., 1999a] [Pretschner A., 1999b] [Trajkova J., 2004], [Speretta M., 2005], [Chirita P., 2005], [Sieg A., 2007a], and [Zhu D., 2008]. Usually, the user profile nodes of the corresponding existing reference ontology reflect general concepts. The following is a detailed explanation of the used techniques.

- In [Ramanathan K, 2008], the hierarchy is built from the Web pages and other documents returned as feedback as follows. Web pages (and other documents) are mapped to a set of Wikipedia concepts. Then a hierarchical profile is constructed from these concepts. Finally, the concepts in the profile are tagged in two ways. One tag describes whether the concept is of transactional or recreational interest. The other tag is a measure of how recent is the user interest in that concept.

- In [Chaffee J., 2000], the user manually constructs his own personal ontology (or uses already existing bookmark structure as one which has been built manually also), and then collects documents he feels they belong to each concept (in case he opted the bookmark structure, he is exempted from this step). Then, they try to determine a mapping between reference ontology and the personal ontology.

- In [Kim H., 2003] and [Kim H., 2008], a set of web pages visited by a user is the input of the profile construction process. [Kim H., 2003] devises a divisive hierarchical clustering (DHC) algorithm to group terms (topics) into a hierarchy where more general interests are represented by a larger set of words. The relations between terms are calculated based on the co-occurrence in the same web page. Each web page can then be assigned to nodes for further processing in learning and predicting interests. The resulting hierarchy is used to build *Page Interest Estimator* (PIE) as well as providing a context. For each cluster in [Kim H., 2003; Kim H., 2008], the associated documents are used as positive examples for learning a PIE.

Another way to build concept hierarchy profiles is to use an existing general reference ontology, tree or taxonomy structure to be as the user's view of world. There are many examples of existing hierarchy of concepts that is suitable to this mission such as DMOZ, WordNet and Wikipedia. Usually the hierarchy is mirrored from a publically accessible browsing hierarchy as in [Pretschner A., 1999a] [Pretschner A., 1999b] [Trajkova J., 2004], [Speretta M., 2005], [Chirita P., 2005], [Sieg A., 2007a], and [Zhu D., 2008].

- The system in [Pretschner A., 1999a] and [Pretschner A., 1999b] extracts its profile from the Magellan hierarchy. The nodes of the ontology are labeled with names of the nodes in the browsing hierarchy. The semantics of the edges of this hierarchy are not specified; in most cases, they corresponds to a specialization relation. Each concept in the profile consists of a weighted vector of keywords constructed from only the documents contained in the node of the browsing hierarchy equivalent to the concept of the profile.

- In [Trajkova J., 2004], after choosing the concepts to be in the profile from the Open Directory Project (ODP), all the web pages that are associated to an individual concept are merged together to form a single super-document. All the super-documents of all the concepts go through an indexing process to calculate and save vector for each concept.

- The system in [Speretta M., 2005], is similar to the system in [Trajkova J., 2004] in that it uses the same way of constructing the profile and the same source of reference ontology; ODP. However, instead of using web pages to train the user profile, it uses search results' titles and snippets. The concept weights are assigned by classifying textual content collected from the user into the appropriate concepts using a vector space classifier and the k-neighbor algorithm. The weights assigned by the classifier are accumulated over the text submitted.

- [Sieg A., 2007a] have used ODP as reference ontology. They utilize the web pages as training data for the representation of the concepts. Using the web pages for a certain node they learn a vector of terms that semantically describes the concept corresponding to the node. In addition to the individual web pages to train a concept, the vectors of all concepts that relate directly with the current concept with specialization relationships are used as well; the current concept is considered the parent. The initial user profile is essentially an annotated instance of the reference ontology. Each concept in the user profile is annotated with an *interest score* which has initial value of one.

Some systems such as [Sieg A., 2005] used a *hybrid approach* that combines both approaches above to build their users' profiles. The user profile in [Sieg A., 2005] uses Yahoo! Concept hierarchy as the already existing reference ontology. To create the aggregate representation of the user profile, a weighted term vector is computed for each concept in the Yahoo concept hierarchy. In Yahoo, each concept contains a collection of documents and a set of sub-concepts. To compute the concept vector first we compute a term vector for each document. Then, the document vectors are summed and added to the summation of the vectors of sub-concepts of the current concept.

3.1.3.2. Build Semantic Networks Profiles

The INFOrmer [O'RIORDAN A., 1995; SORENSEN H., 1997] has a profile represented as a semantic network. The [O'RIORDAN A., 1995; SORENSEN H.,

1997] system uses initial set of interesting documents to learn the user profile. The networks are constructed by first doing the pre-processing. Initially, each sentence is viewed as a chain of nodes linked by edges. Terms that occur in the same article more than once are merged into the same node if the words around them satisfy some measure of similarity. This similarity judgment is necessary because of the problem of homographs (words with the same spelling, but different meanings). The links have a certain fixed initial value (held by a system parameter). These values are later adjusted during the profile adaptation phase.

3.1.3.3. Other Users Profiles Construction Methods

Data to construct the user profile in [Liu F., 2004] is provided in a tree structure as illustrated in section (3.1.2); queries are the roots, level 1 are categories, and level 2 are the documents. Three matrices are constructed from these three levels and the links between them. These matrices are *DT* (from relations between Documents and their terms), *DC* (from the relations between the Documents and their categories), and *M* (built from DT and DC using machine learning) which the final user profile. DT is built from each document words by first removing stop words. Then, documents are stemmed using Porter stemmer. Then, terms within five-term window before and after query terms are selected. After that, tf-idf weights are calculated for each distinct term. DC is built by placing 1 if the document is member to the category and 0 if the document is not.

In ARCH system [Parent S., 2001], once enough documents have been gathered to build the profile which is a set of vectors each representing one category, the system first clusters the documents into semantically related categories. Each document is represented as a term vector, with term weights derived using the tf-idf measure. Clustering algorithms, such as k-means, partition a document set into groups of similar documents based on a measure of vector similarity. Individual profiles are computed based on the centroid of the document clusters. Each individual profile represents a topic category.

3.1.4. Update User Profiles

As user interests and knowledge are changing over time, some web personalization systems try to mimic this human behavior by detecting the user interest shifting property. The following are some techniques that are used to help the system response to the interest shifting:

3.1.4.1. Updating Concept Hierarchy Profiles

In [Ramanathan K, 2008], the method just reflects the recency of the user interest in a particular concept. The recency is based on the age of the pages supporting the concept. He just represents the interest of the user in some category by the recency of its pages. See equation 1.

$$\mathrm{Re}cency = \sum_{pagesOfCategory} \frac{1}{e^{current\_date - access\_date\_of\_the\_page}}$$

Equation 3.1: The recency of a Category in [Ramanthan, 2008]

In [Pretschner A., 1999b], the surfed pages are "characterized" –their categories are determined- using vector space model. The result of the characterization process yields the nodes (or the concepts) of the profile hierarchy that the page belong to. The best 5 categories are updated in terms of time spent on a given page and amount to which they describe a page. The weights in the profile nodes are updated constantly, thus allowing for detection of shifting user interests. The system represents the interest by just the similarity value between the surfed pages and the categories weighted by the percentage of relevancy assurance which is calculated by dividing the time the user spends in reading the page by the number of characters in the page (the author tried different combinations).

In [Trajkova J., 2004], after the system decides the relevancy of a web page, the web page is mapped into a weighted term vector. Then, the highest weighted 20 words are used to represent the content of the page. The page is classified by comparing the page vector with each concept's vector using cosine function. The web is finally classified into the *top-matching concept*

and the similarity weight is added into the concept weight. The concept weight is calculated as a sum of its children's weights and its own weight.

In [Speretta M., 2005], each query or snippet (that already judged as relevant) is classified, resulting in pairs of concepts and corresponding weights in decreasing order of weight. For queries, the top 4 concepts are going to be added into the concepts' weights. However, for snippets top 5 concepts are going to be added into the concepts' weights.

The update function in [Zhu D., 2008] system is applied using a time window for searches. It uses slide time window of 400 searches width.

In [Sieg A., 2005], Based on user behavior, a specific context in the user profile can be updated or a new context can be added. The user profiles are utilized to provide the user with a domain ontology that is more consistent with their view of the world. One approach is to use a similarity measure to compare the pair of term vectors that make up the user's short-term context with the term vectors for each context in the long-term user profile. The term vector that represents positive evidence in the user's short-term interest can be compared to the term vectors that represent positive evidence in the user profile contexts. Similarly, the term vector that represents negative evidence can be compared with the term vectors that represent negative evidence in the contexts that make up the user profile. If the similarity between two term vectors exceeds a certain threshold, those term vectors can be aggregated into a single term vector using vector operations. If the similarity between the term vectors for the user's short-term context and the term vectors in the user profile is below the threshold, a new context is added to the user profile. Note that a new context is added only if the similarity comparison for both the vectors representing positive evidence and the vectors representing negative evidence results below the determined threshold.

As a result of interaction of the user with the system [Sieg A., 2007a; Sieg A., 2007b; Sieg A., 2007c], a vector that represents the local context of the user, a query in this system, is provided as the input for the updating process. The interest scores that are associated with concepts are updated with *spreading*

activation[31] using the input term vector. The first step to update the user profile is to calculate the concepts' activation values for the concepts related to[32] the input vector and their neighbor concepts to-be propagated. The second step is concerned with updating the user profile by the activation values. In this step, the resulting activation value is added to the existing interest score. The interest scores for all concepts are then treated as a vector, which is normalized to a unit length using a pre-defined constant, k, as the length of the vector. Rather than gradually increasing the interest scores, they utilize normalization so that the interest scores can get decremented as well as getting incremented. The concepts in the ontological user profile are updated with the normalized interest scores.

3.1.4.2. Other Methods of User Profiles Updating

Updating user profile in [Sugiyama K., 2004] system does not need any additional function to keep the user profile up to date. The profile in this system is a time window over the web search sessions and web pages visited by the user which is definitely updating itself over time.

In [Liu F., 2004], the user profile is updated by just adding or deleting one of the classes in profile. There is no adapting for weights of already existing classes in the system.

In case of several user profiles such as [Sorensen H., 1995], [Moukas A., 1996], [Moukas A., 1997], a user's interests could be updated using genetic algorithms. The usual operations in genetic algorithms then eventually lead to a generation of profiles that represent the user's interests accurately. In [Sheth B., 1993], [Sheth B., 1994], several instances of the user profile (called agents) compete with each other, and an agent is rewarded when the user liked a suggested document. The common techniques of crossover and mutation yield a generation of agents that eventually represent a user's interest suitably.

[31] The ontological user profile is treated here as the semantic network and the interest scores are updated based on activation values.

[32] Related to is determined by calculating the cosine similarity between the input vector and each concept vector in the profile and finally choose concepts with similarity values greater than zero

3.1.5. Personalization Techniques

This subsection focuses mainly on web search personalization techniques of personalized search and news filtering. There are three common approaches to address the search personalization techniques; re-ranking, filtering, and query Expansion [Pretschner A., 1999a].

3.1.5.1. Personalization by Ranking

Many web personalization systems such as [Pretschner A., 1999a] and [Pretschner A., 1999b], [Meng X., 1999], [Sugiyama K., 2004] and [Speretta M., 2005], [Chirita P., 2005], [Kim H., 2006], [Sieg A., 2007a], [Li L., 2007], [Zhu D., 2008], used to improve and personalize search results by trying to rank again the results. The following are the detailed illustration of the techniques use for ranking results.

When the user submits a query in [Sugiyama K., 2004], the search results are adapted based on his profile. After the subjects submit these queries to Google, the system reorders the search results according to each user's profile. The profile is mapped into just a vector of weights of words. The profile data is divided into three parts each part receives a different treatment according to its importance (the importance is a result of the recency). Each part is converted into a vector of weights using the same method; however, each vector is multiplied to a weight that determines its importance. The three parts are the current session, today's sessions except current session, and the previous days. After constructing the weight vector, each web page of the results of the search engine is also converted into weight vectors using the same way. Then, a similarity between the profile vector and each web page vector is calculated. The web pages are ranked according to these values of similarity.

The system in [Meng X., 1999] re-ranks search results rather than filtering them. For all documents (URLs) that were returned by a search engine, every word contained in them is looked up in the profile. If it exists in the user

profile, its weight in the retrieved document is added to the *URLs score*. This yields a new personalized ranking.

The system described in [Pretschner A., 1999a] and [Pretschner A., 1999b] also ranks again the web search results. Pages that are returned by some search engine are categorized with respect to the aforementioned hierarchy (multiple cosine similarity in the vector space model). The system chooses only four categories with highest similarity values. Then, it calculates the average of these four similarities to obtain the value of user interest in that page. Then, it ranks again the result according to the following formula:

$$\varrho(d_j) = w(d_j).(0.5 + \frac{1}{4}\sum_{i=1}^{4} \iota_i(d_j))$$

Equation 3.2: Calculating new Weights for the Search Engine Result

Where $w(d_j)$is original rank assigned by search engine to rank the document or the page and $\iota_i(d_j)$ is the value of interest to the page with respect to the category i which calculated by multiplying the similarity value, between the document and the category, and interest value of the category.

The system in [Chirita P., 2005] uses pre-categorized search engines such as Google Directory or ODP Search. At run-time, the output given by a search service (from Google Directory, ODP Search, etc.) is re-sorted using a calculated distance from the user profile to each output URL. The distance can be defined as the minimum distance between all pairs of nodes given by the Cartesian product between all the nodes representing the user profile and those associated with the URL from the result. The system also proposed a formula to score URLs that has no topics.

The system of [Zhu D., 2008] organizes the Web snippets (of search results returned from the meta-search engine) into a hierarchy by comparing the similarities between the semantics of each ODP category and the Web snippets. Meanwhile, the Web snippets are clustered to boost the quality of the

categorization. Then according to the descending order of the categories' interest weights of the user profile the search results are displayed to the user.

The application in [Kim H., 2006] uses the user model built in [Kim H., 2003]. To modify the general rank of the search engine, a page score is being calculated. The new rank of the results depends on both the *personal scoring* and the *general rank* of a page. The personal score is a sum of term scores. The term score (which is the importance of the term to the user) is calculated based on four features; deepest level of a node where a term belongs to, length of a term, frequency of a term and emphasis of a term. Finally, the rank order returned by Google is used as the public score. The personal score and public score could be weighted to reflect the more important score. [Hu J., 2008] used both words in the web page and the attributes' values *src*, *name*, and *alt* of the *img* tags after stemming and removing stopwords. Also, the term features are modified to term frequency, term span, term specificity, and node specificity. The scoring function is also modified to normalize documents length using pivoted normalization.

After calculating a weight term vector for each document in the web result, [Sieg A., 2007a] compute the similarities between the documents' vectors and the query. Then, the similarity of the document with each concept in the user profile is computed to identify the best matching concept. Once the best matching concept is identified, a rank score is assigned to the document by multiplying the interest score for the concept, the similarity of the document to the query, and the similarity of the specific concept to the query. Once all documents have been processed, the search results are sorted in descending order with respect to this new rank score.

3.1.5.2. Personalization by Filtering

Examples of such systems are [Pazzani M., 1996], [Montebello M., 1998], [Casasola E., 1998], [Widyantoro D., 1999a], [Widyantoro D., 1999b], [Widyantoro D., 2001], [Pretschner A., 1999a] and [Pretschner A., 1999b].

In [Pretschner A., 1999b], filtering is done by using either one of the ranking functions ϱ_1 - ϱ_5. The idea is straightforward: All weights of the personalized

rankings are normalized between 0 and 1. Then, a threshold is introduced which classifies the relevant documents from the non-relevant ones.

The system filters the results returned by the ProFusion meta-search engine [Gauch S., 1996]. The system in [Casasola E., 1998] decides which result to present to the user and which to discard. For each term in the retrieved documents (or rather their summaries), its weight in the irrelevant set and in the relevant set are used to assess how interesting the document is. This is done by calculating the similarities with the two classes in the vector space model.

Search results are annotated with symbols reflecting the assumed interest (good, okay, don't know, poor) [Pazzani M., 1996]. In [Montebello M., 1998], the system is similar to [Pazzani M., 1996] in that search results are augmented with icons indicating a possible interest of the user. The suggestions are restricted to links that already exist on a page, and if the system considers them interesting, these links are highlighted [Mladenic D., 1996]

A user's browsing history within one particular site is monitored and used to determine the best links to follow which is done by comparison with other users who previously accessed that site [Han E., 1998]. In other words, the idea is to use (potentially global) access patterns of Web usage to recommend links at a particular site by comparing a (probably short) user's browsing history (within that site) with other users' browsing histories [Cooley R., 1999]. The results of these comparisons are then used to point users to interesting links, where interesting links are determined as extrapolation of an individual user's surfing history.

In INFOrmer system [O'Riordan A., 1995; Sorensen H., 1997], where semantic networks are used to represent both the user profile and the documents, the personalization involves the localized matching of structural similarity between the profile network and incoming article networks, using profile weights to influence this comparison.

3.1.5.3. Personalization by Expanding Query

In [Liu F., 2004], the system automatically deduces a small set of categories for the query submitted by the user based on his search history. Then, the system uses the set of categories to augment the query to conduct the web search. The web pages are retrieved by merging (fusion) multiple lists of web pages from multiple query submissions (query without any augmentation and query augmented with each category alone).

The WebMate system [Chen L., 1998] uses user profiles to refine user queries. Also, the system named Watson [Budzik J., 1999] refines queries using a local context but doesn't update the user profile. Inquirus 2 [Glover E., 2001] uses users' preferences to choose data sources and refine queries but it does not have user profiles, and requires the users to provide their preferences of categories, see [Liu F., 2004].

The work in [Parent S., 2001] shows that ARCH system assists the user in formulating his query in three phases. First, the user writes his keywords to perform a web search. Second, using a modular concept classification hierarchy, the system modifies the user original query to suit a specific topic(s) by manually selecting or "deselecting" categories from the concept tree. Once the enhance query, Q2, is derived from the concept hierarchy, each profile can be compared to the query vector for similarity. Those profiles which satisfy a similarity threshold are then used to further expand the query, resulting in a new query, Q3. Note that, as in the case of query expansion based on the concept hierarchy, the new query is computed as a weighted sum of the term vector representing Q2 and the normalized sum of term vectors representing the matching profiles. Third, using learned user profile, the system modifies again the second-phase query to reflect the user interests. In [Sieg A., 2004b] they tried to eliminate the need for the explicit user feedback (the manual selection and/or deselection of categories), by using the user profiles. The user profiles are used to automatically selecting categories.

A new idea has risen by [Sieg A., 2005], after calculating the user context the term vector for the new search query is computed by getting the difference between the positive evidence and the negative evidence.

3.2. Processing Arabic Documents

Different techniques have been developed to overcome the difficulties for the matching process including normalization process, stemming process, morphological analysis process, n-gram for words, using ontologies, etc. This section discusses some of them. Normalization process is used to address the problem of habits of writing. Normalization removes the diacritics so that words without diacritics match with words that have diacritics and normalize the use of HAMZA and TAA MARBOUTA in words, it also could remove Kashida. Usually normalization is used in conjunction with the aforementioned techniques. It is performed at the beginning of the information retrieval process after tokenizing the query and the documents. Another technique is stemming the words; it just removes the most frequent prefixes and suffixes of the word to obtain its stem [Aljlayl M., 2002], [Larkey L., 2002]. Stemming technique gives the highest performance so far. It overcomes word n-gram and morphological analysis techniques [Larkey L., 2007]. However, multiple synonyms, language morphology and polysemy problem still exist. Some systems use ontologies to help understand the queries and documents to improve the performance [Bhogal J., 2007]. Some systems use ontologies to handle the query clarification process by regarding the spatial information that the query and documents may have or by involving the different relations of the ontology to solve this kind of problems [Fu G., 2005]. Many techniques have been used for beating the problem of information retrieval for the Arabic language. At the very beginning, researchers tried to use dictionaries of roots and stems, built manually, for each word to be indexed. The roots and stems extracted from a very small collection of text [Al-Kharashi I., 1994]. This method is not suitable especially when the collection is very big. People tried to use Arabic morphological Analyzers to

obtain the roots of the words automatically to be indexed. A lot of analyzers currently exist and are used and getting evaluated; such as Khoja Morphological Analyzer [Khoja S., 1999], Tim Buckwalter morphological analyzer 1.0[33], ALPNET morphological analyzer [Beesley K., 1996], and Sebawai [Darwish K., 2002a].

A controversial issue at that time is whether to use roots or stems as terms for indexing. Many studies have claimed that roots outperform stems [Al-Kharashi I., 1994], [Hmeidi I., 1997], [Abu-Salem H., 1999] and [Darwish, 2001]. However, more recent studies found that using stems as index terms outperform roots; [Aljlayl M., 2002], [Larkey L., 2002], [Darwish K., 2002b], [Larkey L., 2007], [Taghva K., 2005], [Darwish K., 2005]. The reason that the former researchers, that found the root better than stems for IR tasks, have done their experiments on small collections of text which is not enough for judgment.

TREC 2001 and TREC 2002[34] Conferences [TREC, 2001; TREC, 2002] help a lot for improving the performance of Arabic information retrieval systems. They also helped in evaluating the different techniques for handling Arabic language, in the cross-language Information retrieval tracks. They provided, with help from Linguistic Data Consortium LDC[35], a potentially large text collection to be used in evaluation. This helped in deciding which is more appropriate for use as index term in Arabic information retrieval systems.

Using the TREC-2001 Arabic corpus[36], experiments reveal that roots are not suitable because Arabic consists of few thousands of roots. Analyzing each word to its root would conflate many words of different meaning to the same

[33] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2002L49 [last accessed: Dec, 2011]

[34] http://trec.nist.gov/ [last accessed: Dec, 2011]

[35] http://www.ldc.upenn.edu/ [last accessed: Dec, 2011]

[36] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2001T55 [last accessed: Dec, 2011]

class. For example, the Arabic words for office, book, library, writer, and letter have the same root.

After TREC Arabic cross-language information retrieval tracks (CLIR) [Gey F., 2001], researchers have directed their research to use stems as index terms. They developed a lot of stemmers to handle Arabic Language in IR context. Many studies have been conducted in stemming techniques; [Darwish K., 2002b], [Aljlayl M., 2002], [Larkey L., 2002], [Chen A., 2002], [Larkey L., 2007], [Al-Ameed K., 2005], [Nwesri A., 2005], [Kadri Y., 2006], [Nwesri A., 2007], and [El-Beltagy S., 2009]. Appendix (A) explains nine stemming techniques in details.

3.2.1. Stemming Definition and Stemmers Classification

Stemming has multiple definitions. Shereen Khoja's definition [Khoja S., 2001] limits stemming for Arabic language to the root extraction process. She has defined the stemming process as "…Stemming is the process of removing all of a word's affixes to produce the stem or root. In Arabic this means the removal of prefixes, suffixes and infixes. The stemming component is the rule-based part…". However, Leah Larkey [Larkey L., 2002] was more general in her definition. She could fetch more techniques under the stemming umbrella. She defined stemming processes as "…we use the term *stemming* to refer to any process which conflates related forms or groups forms into equivalence classes, including but not restricted to suffix stripping ….". This definition considers more stemmers than Khoja definition. For example, light stemmers and statistical n-gram methods to conflate words to same class are considered stemmers as well as stemmers that extract roots. A close definition to Larkey's one is [Al-Sughaiyer I., 2003] definition. They defined the stemming as, "… *Stemming* is a method of word standardization used to match some morphologically related words. The *stemming algorithm* is a computational process that gathers all words that share the same stem and have some semantic relation …"

There were also many attempts to classify the existing stemmers. Abdusalam Nwesri [Nwesri A., 2005] has classified the stemmer into heavy stemming, or

root-based stemming, and light stemming. Heavy stemming usually starts by removing well-known prefixes and suffixes. It aims to return the actual root of a word. Light stemming stops after removing prefixes and suffixes, and does not attempt to identify the actual root. He further categorizes the light stemmers into three categories according to the way in which existing stemmers deal with particles; conjunctions and prepositions. He also mentioned that a stemmer can combine between any of these approaches:

- Match and Truncate (MT): the beginning of a word is removed if a match happens and the remaining words more than 3 letters length.
- Remove and Check (RC): the beginning of a word is removed if a match happens and the remaining word exists in the document collection.
- Remove With Other Letters (RW): removing a combination of particles and the definite article ال like فال , وال, كال, and بال

Larkey [Larkey L., 2002] divided the Arabic stemmers into four classes:

- Manually constructed dictionaries
- Algorithmic light stemmers; which remove prefixes and suffixes
- Morphological analyses which attempt to find roots
- Statistical stemmers, which group word variants using clustering techniques.
    - new statistical methods involving parallel corpora

Other classifications are done by [Al-Sughaiyer I., 2003] and [Al-Hajjar A., 2009].

Summary

In this chapter, we have provided a comprehensive review of intelligent techniques for Web personalization. We have described the various explicit and implicit data sources availability along with the typical approaches used to transform this data into useful user profiles that can be used to personalize the Web experiences. We have also described various approaches to filter or recommend different items (such as web pages of search results or news articles, or web links) on behalf of the user of the system.

Commonly, used user profiles are still simple. They are representing the user as a list of keywords. This could be attributed to the system doesn't know what to represent.

Recently research has begun to explore user profiles that are based on ontological information which showed promise in comparison to systems that limit user models to a vector or keyword list.

User interests and needs change over time. Identifying these changes and adapting to them is very important to represent. However, very little research effort has been conducted on this topic.

Arabic language has a very rich morphology system that is used to form the various forms of words which depends on templates. Different techniques have been developed to overcome the difficulties for the matching process including normalization, stemming, morphological analysis, n-gram for words, using ontologies, etc. Normalization removes short vowels and Kashida, normalize the use of HAMZA and TAA MARBOUTA in words.

We use the term *stemming* to refer to any process which conflates related forms or groups forms into equivalence classes, including but not restricted to suffix stripping The stemming technique gives the highest performance so far. It overcomes word n-gram and morphological analysis techniques [Larkey L., 2007]. There were also a lot of attempts to classify the existing stemmers.

CHAPTER 4 : ‘ARABAGENT’ APPLICATION ARCHITECTURE AND FRAMEWORK

'ArabAgent' is a web personalization system used to personalize search and filter news articles. ArabAgent is mainly specialized in Arabic language. The ArabAgent system uses pure content-based approach to provide recommendations to users.

This chapter and the following three chapters are dedicated mainly for our system; ArabAgent. This chapter discusses the framework of the ArabAgent system, the application architecture, and each component of the framework in details. Chapter 5 discusses the techniques used for each component in details (except for the text processing component). Chapter 6 is fully dedicated for text processing techniques, and the evaluation of the ArabAgent system is presented in chapter 7.

Section one discusses the ArabAgent from intelligent Agent view point. The section examines important definitions for intelligent agent and tries to apply the aspects of the definitions to the case of ArabAgent system. Sections two and three discuss the framework of ArabAgent and the components in brief. Section four discusses the application architecture of the ArabAgent system and distribution of the components over the layers and tiers of an internet-based architecture. The components of ArabAgent framework are discussed in more details in sections 4.5, 4.6, 4.7, 4.8, and 4.9. Sections 4.5, 4.6, and 4.7 discuss ArabAgent user interface, query customizer and web wrapper components, respectively. Sections 4.8 and 4.9 discuss the

ranking and filtering component, and the modeler and profiler component, respectively.

4.1. ArabAgent as an Intelligent Agent

ArabAgent system is inherently an intelligent agent. We next examine some of the important definitions and characteristics of intelligent agents introduced by prominent researchers in the field and try to apply the aspects of the definitions to the case of the ArabAgent system.

In light of Russell and Norvig [Russell J., 1995] definition, ArabAgent perceives its environment, which is the user web browser and predefined news websites, through its sensors, which is the changing parameters of the browser through event listeners such as loading web page or a user click, and acting in that browser by amending the web page and search results of certain search engines for the user and send the relevant news articles to the user's e-mail.

According to Maes [Maes P., 1995] definition, the ArabAgent could be seen as inhabiting the web browser and the personalizing server as the "complex dynamic environment". ArabAgent senses autonomously by implementing event listeners to sense user events as they happen on the web pages and sense news article as they are available in the news websites. ArabAgent acts autonomously by modifying the user query, modifying search results and recommend news articles without the user intervention. ArabAgent (by monitoring user behaviors, and customizing the results of search and filtering news article) realizes the main goal of individualizing the access to the web to its user for which they are designed.

ArabAgent could be viewed as assisting people and acting on their behalf in locating and filtering data and saving time by making decisions about what is relevant to the user. Owning and installing the

ArabAgent "software" could be implicitly interpreted as a delegation of filtering news articles and picking relevant search results to the ArabAgent system. Furthermore, ArabAgent "repeatedly" filters news articles and ranks search result as the user search again and again, "learns" user interests and preferences, and "recommends" news articles.

In case of ArabAgent, the tasks are filtering news articles and tailoring web search results in order to satisfy the user needs using the user feedback of the relevancy of previous web pages and search results as gleaned information from the environment. We could deem the event that happens within the browser such as web page loading and user visiting certain web pages gleaned information as well. ArabAgent acts in a suitable manner while filtering news articles and tagging or ranking search results so it helps the user to individualize its view of the web. ArabAgent adapts itself as the user's interests change (this change is realized by monitoring the users behaviors and choices while accessing the internet).

To summarize, ArabAgent system is inherently an intelligent agent. One thing helped in that is that information overload problem in the internet is a typical problem to be solved using intelligent Agents. Information overload required the user "repeatedly" to spend a lot of time filtering information on the internet, wading through hundreds or thousands of web search results, and modifying queries submitted to search engines to improve search results. As a result, ArabAgent is used to assist its user and ("or acts on behalf of its user") to tailor web search results, modify user query, and filter news articles. ArabAgent accepts user's relevance feedback to adapt itself to the new interests of the user. See the ArabAgent as a black box in Figure 4.1. Next section describes the ArabAgent framework.

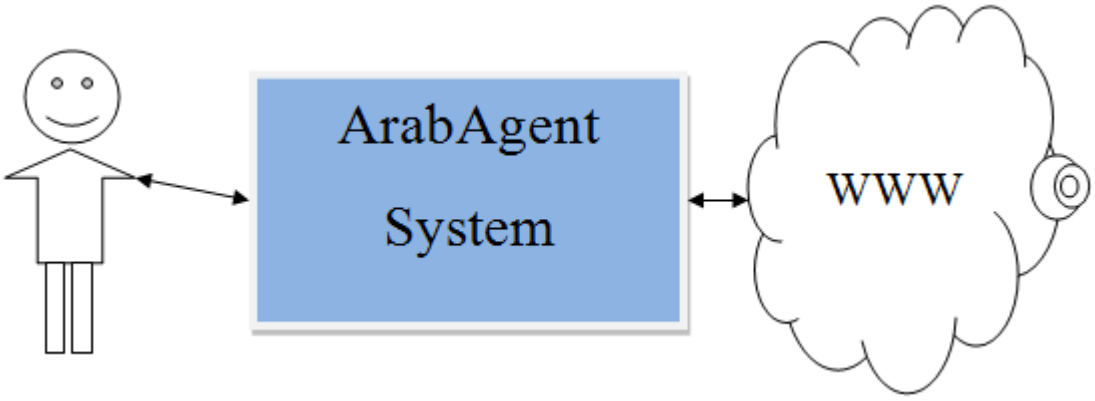

Figure 4.1: ArabAgent System as a black box

4.2. ArabAgent Framework

Figure 4.2 shows the ArabAgent framework. The framework could be used as a general framework of agent used for web personalization regardless of its application (e.g. personalized search, navigation assistant, content personalization, etc). It contains the main component needed by most web personalization systems. The framework could accommodate more components and sub-components of services, functions, or processes as needed.

The ArabAgent system consists of seven components; user profile, user interface, query customizer, web wrapper, ranking and filtering component, user modeler and profiler, and text processing component.

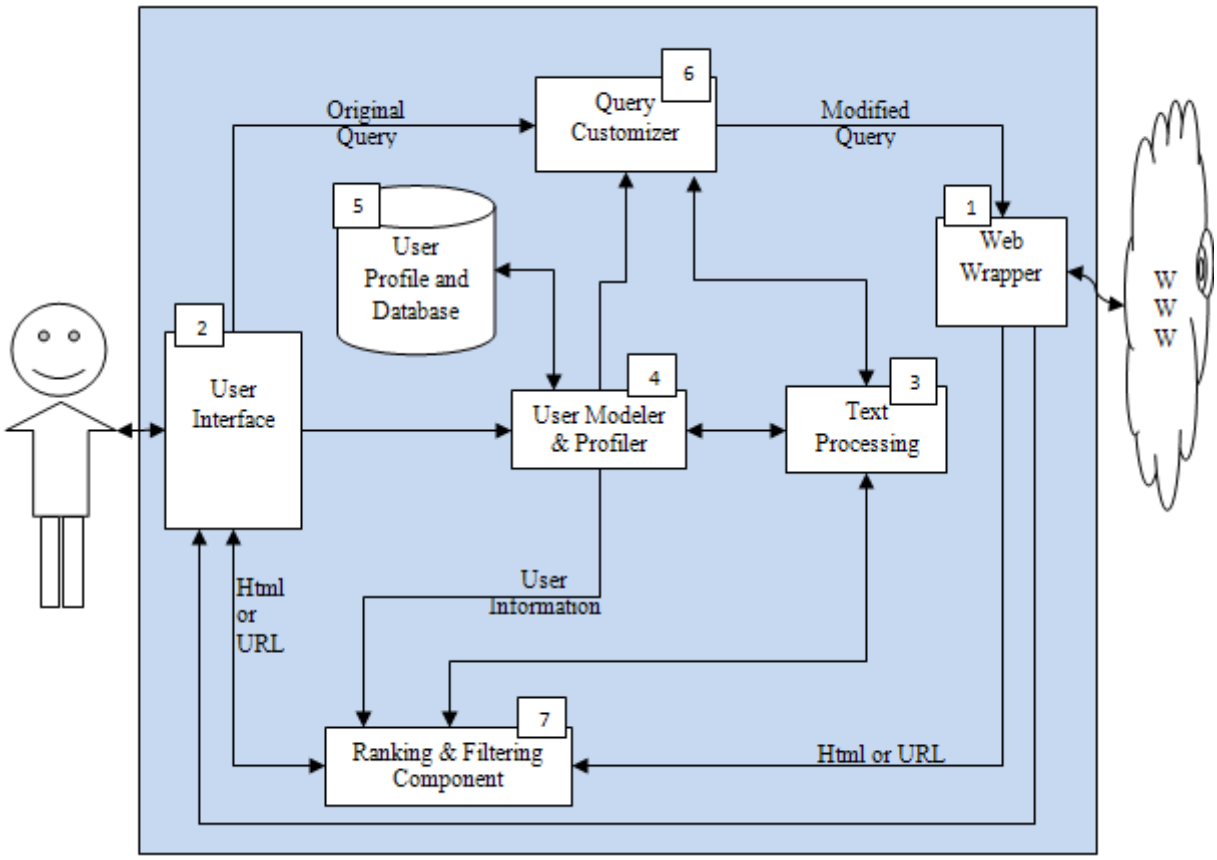

Figure 4.2: ArabAgent System Framework

4.3. ArabAgent Components

In this section we briefly discuss the components of the ArabAgent framework. Sections 4.5, 4.6, 4.7, 4.8 and 4.9 provide more illustration for the components. Chapter 6 provides thorough illustration for the text processing component and its techniques.

The web wrapper is the only interface with the Web in the system. The web wrapper is responsible of monitoring the web. The wrapper takes actions according to certain cases. For example, it periodically checks for the availability of news articles in specific news websites, and chooses to either send them to the user's email or directly to his interface.

In addition, the wrapper forms the data submitted to the internet to suit certain websites. For example, it is responsible for formatting the

(modified) user query to suit the search engine. It also extracts links and web pages from certain web pages such as search engines' pages.

The web browser is an old experience that every internet user is familiar with. To avoid annoying the user with installing and learning a new software application, we opt to use the web browser as the user interface to ArabAgent. It is easy to add an extension (such as a toolbar or a button) that modifies the behavior of the web browser in certain cases. To fully leverage ArabAgent potentials the user has to install the ArabAgent extension for his web browser. The extension helps the system to monitor the user activities and to log these activities in a special file that could be sent to the personalization server to update the user profile.

In addition to monitoring and logging, the user interface accepts external feedback from the user for web pages and search results.

The ArabAgent user interface supports user feedback in many ways. The extension contains a button that could be pressed if the user is interested in the web page currently viewed by the web browser. After wrapping the search results with feedback controls, the user interface presents the search results in a way the user could provide feedback for these search results.

If the user interface detected that the user is submitting a search query to one of the search engines it monitors, the user interface sends the query to the query customizer to modify it before sending it to the search engine.

The main goal of the text processing component is identifying the important concepts that represent the documents; either these documents are web pages from the user relevance feedback or web pages returned as a result from search engines. It also identifies terms (not concepts) as part of the user query processing task.

Identifying concepts from documents considers two steps. The first step is to detect phrases that seems to be concepts and assign for each of them the corresponding prospective concept(s) id(s) (in case of polysemy, a term can have several concept ids), and second step is to disambiguate between concepts for phrases that have multiple concepts.

The text processing component depends essentially on data and statistics extracted from the Arabic Wikipedia project. Thorough illustration on techniques and heuristics used to elicit the data from the dump of Arabic Wikipedia project is in chapter 6.

User modeler and profiler is the component that is responsible for creating user profile (if not exist), loading user model, and updating the user profile. The user modeler and profiler component is the only component that directly deals with user profile.

The user modeler and profiler component accepts web pages of feedback from the user interface. Then, it sends these web pages to text processing component. The text processing component sends back the concepts identified in the web pages. The user modeler and profiler component updates the profile with the concepts and connections returned from text processing component.

The user modeler and profiler component also provides user model for both query customizer component and ranking & filtering component so they can personalize their content.

As a personalization system, one of the key aspects is to maintain the user's interests and preferences. Accordingly, a user profile is an integral part of most personalization systems. The data about the user is stored in the user profile.

In ArabAgent, the user profile is stored as an XML file, containing data of the user.

Most of the systems (that are content-based approach) try to represent user interests through representing the keywords of the documents that the user showed interest in, and they claim that they represent user interests. However, in ArabAgent system, the matter is tackled differently. According to Lawrence Page [Page L., 1998], “The importance of a Web page is an inherently subjective matter, which depends on the reader’s interests, knowledge and attitudes”. Accordingly, ArabAgent attempts to represent the user's interests and knowledge to define the importance of a web page to the user as a way to individualize the access to the web.

The user model is represented using two structures; semantic networks and hierarchy of categories.

The query customizer is responsible for modifying the user query to suit his/her need. It could expand the user query or alter some keywords or search. The query customizer could narrow the search by limiting it to specific websites (for example websites the user likes to visit).

The ArabAgent system personalizes the user experience using different ways depending on the task the user performs. If the user is searching the web using one of the monitored search engines (such as Google), the ArabAgent system (using ranking and filtering component) tag the results returned by the web search engine by signs to indicate the likelihood of a webpage in a search result to be relevant to the user.

Because the previous method may make the user sift through loads of links in the result set to satisfy all his/her needs, we support the previous method by additional service that ranking the result again. The new rank of the result is supported in a sidebar (so we make the user feel that we don't interfere in his decision which makes him gradually trusts the system).

Furthermore, the ArabAgent system provides filtering techniques for some news websites. The ArabAgent checks for news availability in the websites every five minutes. The ranking and filtering component filters the new news article and if the article is likely to be relevant it sends it to the user email.

4.4. ArabAgent Application Architecture

The ArabAgent system has been divided into two parts; the first part is in the client side and the other exists in a server side. Although, ArabAgent intended to be a single agent that applies a content-based approach to recommend items to the users, the framework and architecture of the ArabAgent system is flexible in design. Since there is part of the system that resides in a separate server, the ArabAgent could serve as multi-agent collaborative recommendation system with a minor modification to the framework.

[Whitten J., 2004] states that there are five general layers that can be distributed differently according to the chosen system architecture. These five layers are the presentation layer, presentation logic layer, the application logic layer, data manipulation layer, and finally the data layer. In addition, there are three system architectures that can contain these layers; file server architecture, client/server architecture, and Internet-based architecture.

Since the user interface of the ArabAgent system and the user profile are communicating through the internet and since each user has its own account, the internet-based architecture is most suitable for the ArabAgent System.

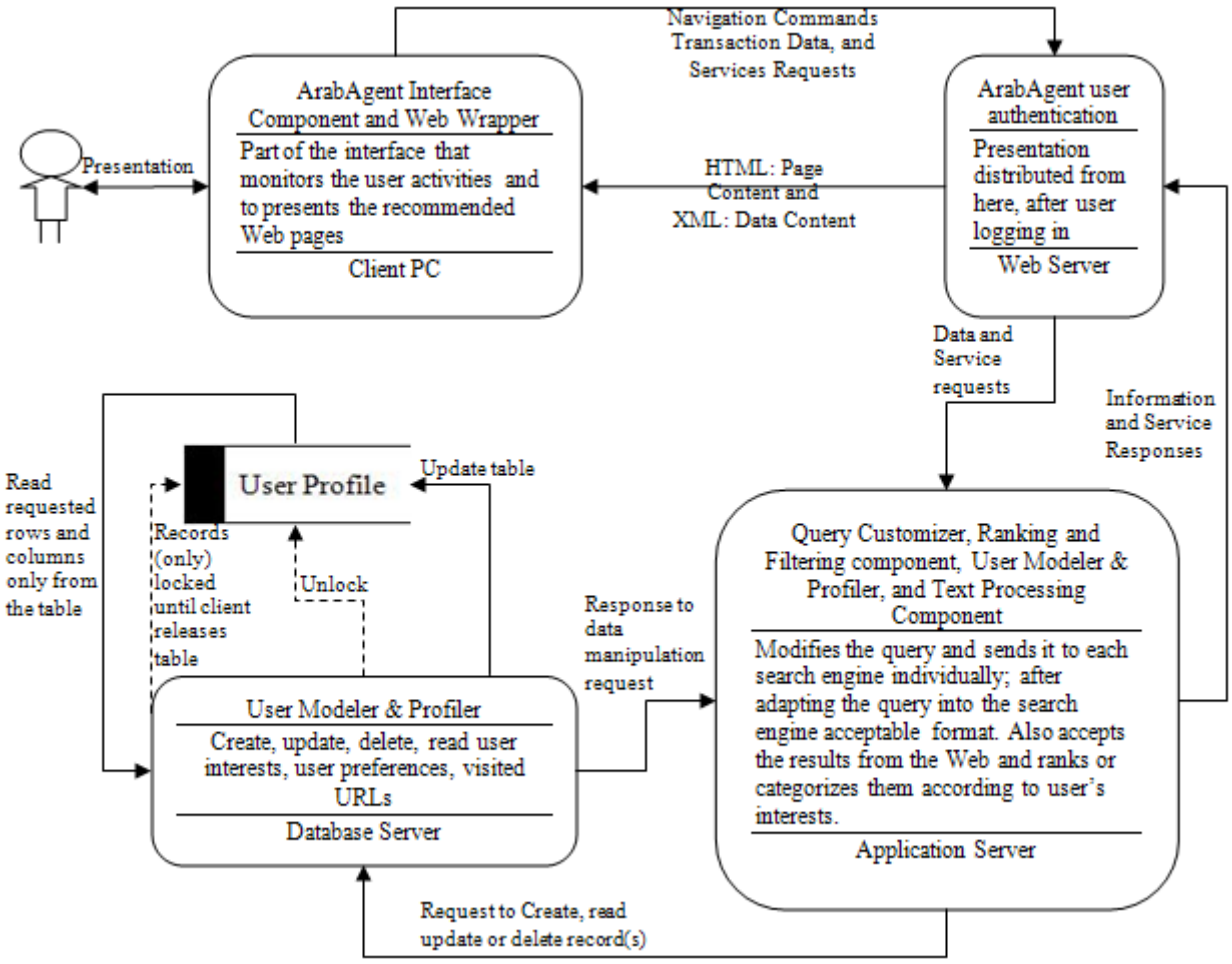

Figure 4.3: The Application Architecture of the ArabAgent system as Internet based Architecture

The first layer of the ArabAgent system is the presentation layer. It consists of what is shown to the user in the monitor. It could be a web page, or the Web pages that are displayed by the Web browser. These web pages are formatted and presented using the next layer which is the presentation logic layer. The presentation logic layer of the ArabAgent system is the part that monitors the user's activity and, collects user relevance feedback and useful data that represents his interests. This part also formats and presents the recommended web pages for the user. Here, the presentation layer and the presentation logic layer represent the user interface of the ArabAgent system.

The user data that has been collected by the presentation logic layer is sent to the application logic layer, or shortly the logic layer, the data is

then manipulated and used to update the user profile. Also, the logic layer modifies the user query using the data previously saved and collected about the user. The logic layer then sends the modified query to each search engine individually; after adapting the query into the search engine acceptable format. The logic layer then accepts the results from the Web browser and ranks or categorizes them according to user's interests. The application logic layer consists of the following components; query customizer component, the web wrapper, ranking and filtering component, and text processing component.

The data manipulation layer includes the user modeler and profiler. Data manipulation layer is used to access the user profile. Other components can't retrieve or restore data into the user profile except through this layer. User profile is the data layer. It may contain user preferences such as search engine preference degrees, URLs visited by the user, the user interests and others.

The ArabAgent system implements the Internet-based Architecture; sometimes called multi-tier architecture. See Figure 4.3. Each rectangle represents a tier. The first tier, that contains the presentation layer and presentation logic layer, is implemented at the client-side.

4.5. ArabAgent User Interface

The user interface is considered a very important part of any system especially those where directly face non-technical non-specialist users of the system such as the majority of internet users. One of the main reasons of the success of Google search engine is its simple user interface. The interface of Google search engine is very simple consisting merely of text box and a button.

In ArabAgent, we consider this simplicity in the user interface. We also consider not facing the user with new features and properties that he/she is not familiar with and needed to be learned. As a result, the ArabAgent user interface is chosen to be an extension for most of the

prevalent web browsers such as Firefox, Internet Explorer, and Google Chrome.

The extension of ArabAgent could appear as toolbar or a button in the web browser. In case of toolbar, the extension has only two buttons; one is labeled “options” and the other as “feedback”. Using the “option” button, the user can adjust few parameters of the ArabAgent system such as the search engines he or she needs to track, news websites he or she is interested in and from which news articles are going to be sent to his email. While browsing, when the user finds useful web pages, the “feedback” button is used to indicate the relevancy of these pages. Furthermore, as the user marks an item in the result set as relevant, the user interface sends the user query and the items's URL to the updater component.

As the user writes his query and presses the search button, the user interface sends the user query to the query customization component before sending it to the search engine.

Using this extension allows to automatically append feedback controls to search engine results. After detecting a result page of any tracked search engine, the system adds two radio buttons for each item in the search result; one labeled "relevant" and the other labeled "irrelevant". In case the user finds the item relevant he could choose relevant. The extension could be improved in the future to support implicit user feedback (feedback without user intervention) while user browsing, reading, bookmarking, saving, and printing web pages.

The user interface tags the items in the search result with color symbols that indicate the relevancy likelihoods of the items. The symbols take color between red (which means far away from the user interest) and green (which means strongly recommended for the user). Each item in the search result page is tagged with the symbol during page loading process. Each symbol sends the URL of the item to the

ranking and filtering component to asynchronously load its color using the Ajax web technology.

4.6. Query Customizer Component

The query customizer component is responsible for individualizing the query submitted to search engines that are monitored by the ArabAgent system. The query customizer is one component used for personalizing the user access to the web. The other component is the ranking and filtering component.

Personalizing the user experience using query processing techniques has several advantages over other techniques that handle (rank or tag) search results. Although, techniques that deal merely with search result set try to improve the effectiveness of the system by improving its precision at particular rank, they totally ignore the recall measure. The reason behind that is that the system is confined only to the set of results returned by a certain query. This could decline the overall system performance in case where the user submits bad query keywords (the result set contains no relevant data at all). Accordingly, some systems opt to personalize the search by modifying the user original query to avoid these drawbacks. Furthermore, it is difficult to rank again all the result of the search engines. The user can't tolerate the relatively long time the system takes to rank all the results.

In the other hand, one of the main disadvantages of the keyword search, which has been adopted by the majority of search engines such as Google, is that the result is very sensitive for keywords used in web search. Different keywords yield different results for most search engines; even if they have the same meaning. Accordingly, modifying the user query is tricky and difficult. You cannot augment new keywords to the user original query unless you are hundred percent sure they are going to improve the result. The keywords could take the

query to another direction causing the result to decline in terms of both precision and recall.

Therefore, in ArabAgent, we introduced a novel technique to modify the user query. Instead of adding just the synonyms for the query, a technique to reduce the ambiguity of the query is used. This technique is suitable only for the informational kind of queries [Broder A., 2002] -queries that cover a broad topic (e.g., neural networks, ancient Greece) for which there may be thousands of relevant results and usually used for learning about the thing you are searching for. Using this technique with other kinds of queries such as transactional and navigational queries would harm them.

Figure 4.4 depicts the sub-component of the query customizer. The technique is summarized in the following algorithm:

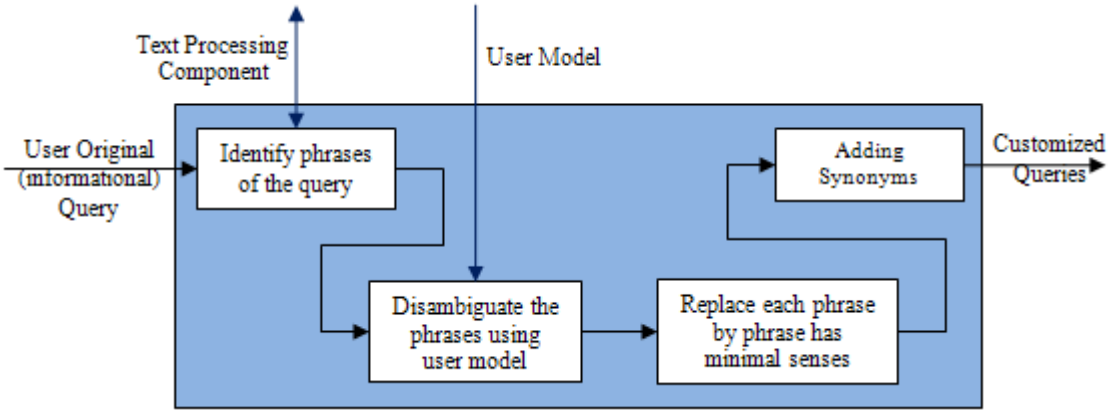

Figure 4.4: Query Customizer Component

- Identify just the phrases (not the concept) in the query using the phrase detection sub-component of the text processing component, then
- Use the user model to disambiguate the phrases in query with most interesting concept to user.
- Replace each phrase in the query by another phrase for the same concept, but with minimum senses or concepts. (this could only

done for informational queries, however other queries such navigational and transactional queries is not suitable)

- The component could (optional) limit the search of the search engine to certain websites.
- The component could (optional) add synonyms and acronyms for each phrase in the query (this is only provided for informational queries)

4.7. Web Wrapper of ArabAgent

The Web Wrapper of ArabAgent system is the component that deals directly with the Web matters such as submitting queries for the search engine, receiving result from the search engine, extracting information from results, extracting news article from web pages etc.

After Query customizer component modify the user query, it sends the query to the web wrapper. The web wrapper adapts the query to suit the search engine sending to. The web wrapper receives the result from the search engine and sends it to the user interface to show it. In case a new news article available in the news website the web wrapper sends it to ranking and filtering component before send it to user email.

Although, most of the search engines accept free text search queries, each search engine has its own set of operators. For example, in addition to the well-known widely used operators such as OR, AND, and NOT, there are a lot of specific operators to Google search engine, for example, such as allinanchor:, allintext:, allintitle:, allinurl:, cache:, define:, filetype:, id:, inanchor:, info:, intext:, intitle:, inurl:, link:, phonebook:, related:, site:, and more. The Web wrapper forms each query to suit the search engine submitted to.

In addition to search engines' operators, there are search result URL parameters. The web wrapper extracts the values of these parameters

or changes them in order to obtain the results of a search. These parameters differ from search engine to another. Examples of such parameters for Google search URL are "q=", "as_epq=", "as_oq=", and "as_eq=".

4.8. Ranking and Filtering Component

Ranking and filtering component is the other component used in personalizing user access into the web. The ranking and filtering component is mainly responsible for weighting web pages. It is used to filter news articles as they are available in their websites as well as tagging search result during search process. It produces a weight representing the relevancy likelihood of the document in respect to user interests and knowledge. See Figure 4.5.

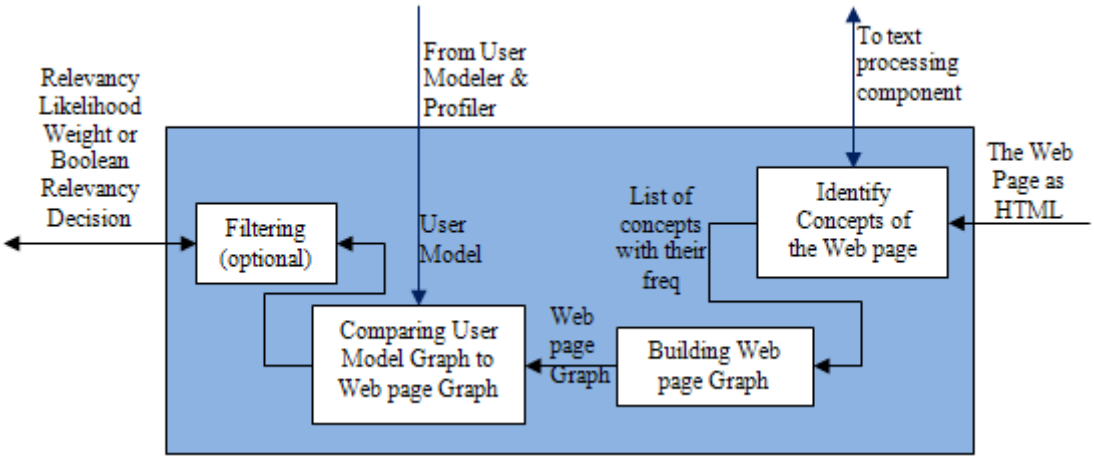

Figure 4.5: Ranking and Filtering Component

After loading the search result in the user interface, the interface communicates with the ranking and filtering component asynchronously to either rank or tag the result set. The asynchronous property prevents the user from feeling the relatively long time the ranking and filtering component takes to weight each item in the result set.

Before the ranking and filtering component calculates the weight of a web page, it sends the web page to the text processing component. The text processing component identifies the concepts of the web page and

sends the concepts back to the ranking and filtering component. The ranking and filtering component then builds a semantic network of the concepts and their frequencies. The semantic network of the Web page is then compared to the semantic network of the user model to compute the relevancy likelihood weight of the web page.

In case of filtering news articles, an additional step is performed after computing the relevancy likelihood weight of the news article. The ranking and filtering component has to decide the relevancy of the article, whether relevant or irrelevant. Accordingly, a decision is made whether to send the article to the user's email or not.

4.9. The Modeler and profiler component

The modeler and profiler component is the only component that directly manipulates the user profile. The modeler and profiler component updates the user profile, loads the user model from the user profile, and provides the user model for all other components for personalization.

The data extracted from the user feedback is stored in a user profile. The modeler and profiler component sends the user relevancy feedback (after receiving it from the user interface) to the text processing component to identify the concepts in the web pages. After identifying concepts, the modeler and profiler component loads the user profile into memory and updates it with semantic network of the documents. The modeler and profiler component saves the profile after updating.

Although the user profile stores the user data, ArabAgent doesn't use the user profile directly in personalization tasks (i.e. tagging, ranking, or modifying query). The system has to generate the more convenient form of user model for personalization tasks. The user model is the system view of the user. The modeler and profiler component

generates the user model and provides it to query customizer component and ranking and filtering component.

Summary

Our approach is to use *Intelligent Information Agent* as a general approach and framework to overcome the problem of overloading. ArabAgent is used to assist its user and ("or acts on behalf of its user") to tailor web search results, modify user query, and filter news articles. ArabAgent accepts user's relevance feedback to adapt itself to the new interests of the user.

The ArabAgent system consists of seven components; user profile, user interface, query customizer, web wrapper, ranking and filtering component, user modeler and profiler, and text processing component.

The interface of the ArabAgent system consists of the web browser with an extension that monitors user activities and logs them in a special file that could be sent to the personalization server to update the user profile.

The ArabAgent system has been divided into parts; the first part is in the client side and the other side exists in a server side. This architecture provides dynamic in design, since it is suitable for single content-based Agent as well as Multi-Agent Collaborative recommendation system.

The ArabAgent system implements the Internet-based Architecture; sometimes called multitier architecture. Each rectangle represents a tier. The first tier, that contains the presentation layer and presentation logic layer, is implemented at the client-side. Intuitively, this tier represents the user interface component.

The text processing step is fully or partially required for the following: user profile updating Task, filtering and customization Task, and query modifying task.

The main goal of the text processing component is identifying concepts from documents. Identifying concepts from documents considers two steps. The first step is to detect phrases that seems to be concepts and assign for each of them the corresponding prospective concept(s) id(s) (in case of polysemy, a term can have several concept ids), and second step is to disambiguate between concepts for phrases that have multiple concepts.

The Arabic Wikipedia project has been chosen to be a source to provide data and statistics to build the user profile. Furthermore, Arabic Wikipedia project has facilitated building a word sense disambiguation technique based on Wikipedia link structure to detect and disambiguate between concepts to use in the text processing component.

Although the great advantages of multi-agent systems, our system, ArabAgent, is a stand-alone agent that act on behalf of his user to pursue his/her goal while accessing information on the Web. ArabAgent is an Autonomous intelligent agent which is capable of learning and adapting to represent his user. It represents his user interests and knowledge while surfing and searching the World Wide Web.

CHAPTER 5 : PERSONALIZATION APPROACHES AND TECHNIQUES USED IN ARABAGENT

This chapter and the next chapter discuss and illustrate the techniques used to apply functions of the components of *ArabAgent* framework that has been mentioned in the previous chapter. While the next chapter (chapter 6) is fully dedicated for text processing techniques, this chapter comprises the rest of the approaches and techniques of ArabAgent.

As the user uses the interface of the system through a web browser, the system does one of two things, either monitors user to acquire feedback or involves in a personalization task (i.e. filtering or ranking). Both of these tasks requires processing web pages: in feedback, the system has to download the web page and identifies its concepts before updating the profile. Also, in personalization each web page to be ranked or filtered has to be downloaded and its concepts have to be identified before deciding its relevancy likelihood.

Therefore, this chapter discusses the techniques used in filtering web pages, ranking web pages, representing user in user model, creating user profile, updating user profile, user feedback techniques and other techniques used to implement the components of the ArabAgent framework. Techniques used to process Arabic text and identifying the concepts of the web documents are illustrated in chapter 6.

This chapter discusses user relevance feedback in section 5.1. The ArabAgent system maintains its point of view of user to use in the personalization tasks, this point of view is the user model which is generated from the user profile as discussed in section 5.4. In section 5.3, we discuss user profile structure and how to calculate its attributes (i.e. concept importance and connection importance). Updating user profile task is discussed in section 5.4. Three

techniques for personalization are discussed in section 5.5. The comparison of the graph generated from a web page and the user model graph to compute the relevancy likelihood of the web page is in section 5.6.

5.1. User Relevance Feedback

As Manning [Manning C., 2008] mentioned, there are three types of user feedback techniques; (1) explicit user relevance feedback, (2) implicit user relevance feedback and (3) blind relevant feedback. Explicit user relevance feedback, or shortly explicit feedback, is the type of feedback the user provides manually by, for example, choosing between relevant and irrelevant set of documents or by providing rate of relevancy to a document. Explicit feedback is most trusted and reliable source of feedback. However, explicit feedback constitutes a burden to the user and could sidetrack him from his search goals.

Implicit user relevance feedback, or shortly implicit feedback, is type of feedback that is used to collect data and information about the user navigation and search behavior without the intervention of the user. The liger time[37], opened web pages, etc. Although this type of feedback overcomes the problem of explicit feedback, the certainty of its relevancy is not guaranteed.

In search context, blind feedback assumes that top (n) documents are relevant. According to that, it extracts (m) keywords with highest frequencies of occurrences from those (n) documents to expand the user query. Although this assumption is not always true, the results of expansion have higher performance than those without blind relevance feedback [Manning C., 2008]. However, the relevancy of such feedback is less certainty even from implicit feedback.

Although, using implicit techniques of the user relevance feedback techniques for web personalization systems are very beneficial since they keep the user focus in his goal without sidetracking or distraction, ArabAgent system applies explicit techniques to focus mainly on the task of user modeling and

[37] The time that the user spends on a web page

expressing user context without being concerned by the efficiency and accuracy of the implicit techniques used to judge on the relevancy of an item or object the user inspect.

ArabAgent provides two ways for explicit feedback; while searching and while browsing the Web. As the user browses the Web he or she could tag a Web page as relevant. In Addition, for each search result of a search engine, he could assign a relevancy status since there is a feedback input associated with every item in the result.

5.2. ArabAgent User Model

As a personalization system, one of the key aspects is to maintain the user's interests and preferences. Accordingly, a user profile is an integral part of most personalization systems. The data about the user is stored in the user profile and administrated by a user modeling system [Wahlster W., 1989] as stated by Heckmann [Heckmann D., 2005]. The user profile is used later to generate a so-called the user model which is used in personalization tasks such as ranking search results filtering news article and modifying user queries.

In ArabAgent, the user profile is stored in as an XML file containing partially analyzed data of the user (this level of analysis is just for reducing modeling time). The user model comprises of semantic networks that represents user's knowledge, long-term interests and short term-interests. We first address the user model in this section. User profile of ArabAgent is discussed in the next section (5.3).

User model is the system view of its user. Koch [Koch N., 2000] describes a user model as the representation of the system's beliefs about the user; see [Heckmann D., 2005].

Web personalization systems that use merely taxonomy or a hierarchy of concepts alone to represent the user's interest suffer from a serious flaw. This flaw is inherent in the idea that although the user is usually concerned with a specific topic (the context) while he or she is searching or navigating, most of

the documents that attract the user are usually related to many concepts that are related to each other. And because these hierarchies or ontologies just represent the concepts of a knowledge domain, these systems poorly represent the user's interests and knowledge.

To make the user model reflect the above vision, two structures have been combined together to comprise the user model; a hierarchy of categories and semantic networks. See Figure 5.1.

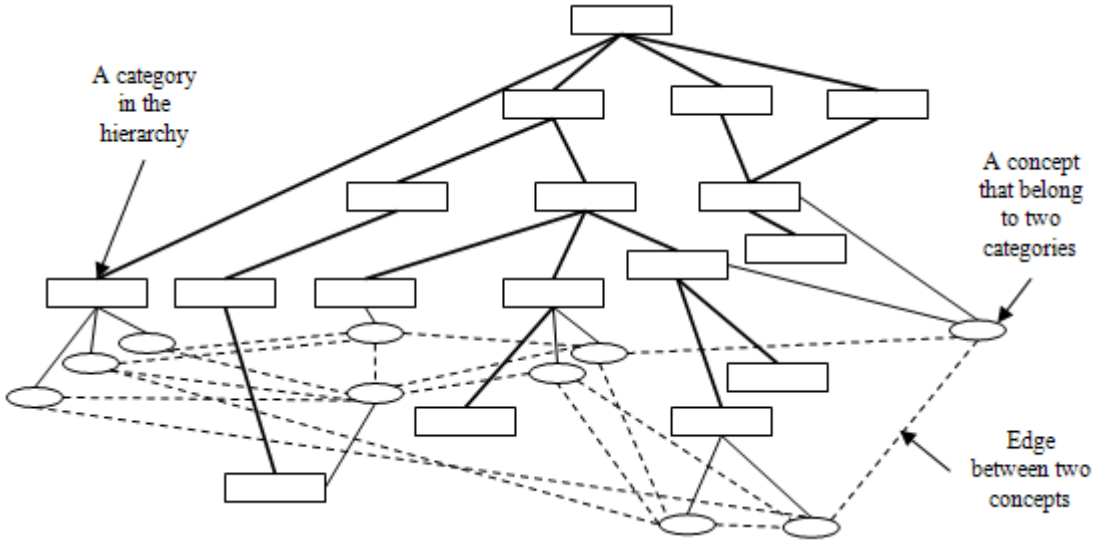

Figure 5.1: Sample User Model

Each time a new interest appears; new node is added to the semantic networks to represent both his knowledge and interest values. While the user interests may change (decay) over time, user knowledge is maintained persistent. This has been represented using the structure itself of the semantic network and two attributes tagging the nodes and the edges of the network. The first attribute is named "lt-interest", and represents long-term interests and the second attribute named "st-interest" and represents the short-term interests of the user.

Our user modeling provides the following:

- General knowledge through a hierarchy of categories; the general knowledge provides a source of knowledge domain to help anticipate new topic that may concern the user but he doesn't know about its existence. This solves the problem of serendipity through a comparison process between the user knowledge and the general knowledge,

- User Knowledge through the semantic networks connecting nodes, which represent concepts the user was (and still) interested in, and the edges represent concepts that user like to co-occur in the same document. Weights of both nodes and edges represent the amount of experience the user has in a particular topic and how often two or more concepts co-occur,

- Representation of user interests through "st-interest" attribute. The user interest is represented as an attribute tagging each node and edge in the semantic network. The attribute value is decayed over time as the concept for a corresponding node stop occurring in user's interested pages or concepts stop co-occur together.

- The capability of representing short-term (transient or ephemeral information needs) and long-term (persistent information needs) interests through hierarchy of categories. Long-term interests usually are the broad topics of the short-term interests. This does not mean that we consider every broad topic of each instant short-term interest. However, for a topic to be a long-term interest its subcategories should be mentioned frequently. A topic of a leaf category could be a long-term interest as well, if it has been mentioned frequently for a long period of time,

- The capability of representing short-term and long-term interests through weights of edges connecting concepts in the semantic Networks. Long-term interests are represented through the semantic networks by determining the lowest condition probability to be the long-term interests and the high condition probability to be the short-term interest.

The INFOrmer system [O'Riordan A., 1995; Sorensen H., 1997], ifWeb system [Asnicar F., 1997], and SiteIF system [Stefani A., 1998] use semantic networks to represent the documents that the user interested in. However, they represent the user differently; while they connect tokens and words exist in the same sentence only (and the order of the words in the sentence determines the direction of the edge

between the two words), ArabAgent connects all the concepts that exist in a web page to each other. The semantic networks in the ArabAgent are weighted undirected graph and weights are represented differently.

In all the above systems the nodes of the networks represent only single token or word. However, in ArabAgent each node is corresponding to a concept that could represents a single or compound word, several morphological variants, synonyms, acronyms, or any set of characters that are equivalent in meaning.

5.2.1. ArabAgent Hierarchy of Categories

As we mentioned above, the user model consists of hierarchy of categories and semantic networks. Each category in the hierarchy embraces one or more nodes. The hierarchy is constructed once and forever (but the attributes are updated frequently). The hierarchy of categories is generated using the Arabic Wikipedia project. Before the initial installation of ArabAgent system this hierarchy of categories and their concept nodes need to be generated.

The Arabic Wikipedia project is used to be the source of both hierarchy of categories and their concept nodes for many reasons. Furthermore, The Arabic Wikipedia project is used to elicit data and statistics needed for phrase detection and phrase sense disambiguation tasks. See chapter 6.

5.2.2. ArabAgent Semantic Networks

The nodes of the semantic networks are the concepts of the categories. The Edges connecting between two nodes are undirected and tagged with two weights; one represents the long-term interests and the other represents the short-term interests. Edges of the semantic network connect between two concepts if they co-occur in the same document. Each concept and edge is tagged with two attributes; one represents the short-term interest of the user toward this concept or edge and the other represents the long-term interest.

The semantic networks are built gradually and updated frequently as user provides relevance feedback to update the user profile and as time pass away.

As profile is updated, edges in the semantic networks are created to connect between the concepts as they co-occur in the relevant documents and the attributes values of the concepts (occurrence and interest attributes) are updated as well. The calculation of the attributes of both short-term interest and long-term attributes is done through the process of updating the user profile.

5.2.3. Building Short-term Interests Calculating Formula

In this section we build two formulas that calculate the weights of short-terms for both concepts and edges. We gradually build these two formulas to help the reader understand the justification of building them.

5.2.3.1. Building Short-term Interests Calculating Formula for Concepts

The function that is going to be built is computed each time the system updates the user profile since new information is available and should be considered in the user model. The function or the formula represents the interest of the user over 30 days. It takes into consideration number of times that each concept is updated, the recency of days where updates occur, and the distribution of updating incidence for each concept and connection over time.

As the formula takes into consideration the number of times a concept updated, or for short *CUF*, stands for *concept updating frequency*, this present in the initial form of Equation 5.1. Note that the system performs a single update operation for each document.

$$st\_interest(c) = \sum_{i=1}^{30} NoOfDocumentsContainTheConcept(c, i)$$

Equation 5.1: Considering only number of Documents in Calculating Short-term Interests in the last 30 days for Concepts

Where (c) is the concept we need to compute the user interest for and (i) is an iteration index for the last 30 days of the profile.

However, concept updating frequency is not completely sufficient. Consider the two cases when two users have mentioned a particular concept in 20

documents. However, in the first case each document of the 20 has mentioned the concept only 1 time whereas in the other case each document has mentioned the concept about one fifth of the total occurrences of the concepts in the document. Using the number of documents that present a certain concept is not enough to represent the importance of a concept for the user. Therefore, Equation 5.1 needs to adapt to Equation 5.2:

$$st\_interest(c) = \sum_{i=1}^{30} \sum_{DocCount(c)} \frac{ConceptCount(DOC_c)}{AllConceptCount(DOC_c)}$$

Equation 5.2: Considering both the number of the documents and Importance of the Concept for the last 30 days in Calculating short-term interest for Concepts

Another reasonable factor to consider is the positions of days in day frequency; day recency. Suppose that the two users have considered relevant the same set of documents for instance over five days, but the first user considered them in the beginning of the 30 days and the other user considered them in the end of the 30 days. It seems that the second user should be more interested in the concept. To overcome this problem, each day of the 30 days has been given a weight to represent its recency. The weight is computed by calculating the difference between the current date and the date the document considered relevant in, and then subtract the difference from 30. Then, the outcome is divided by 30 as in the next formula; Equation 5.3.

$$dayImportance(date) = \frac{30 - (currentDate - date)}{30}$$

Equation 5.3: Calculating the Importance of a Day

Combining this weight with Equation 5.2 yields the final Equation 5.4:

$$st\_interest(c) = \sum_{i=1}^{30} \left(dayImportance(Date(i)) * \sum_{DocCount(c)} \frac{ConceptCount(DOC_c)}{AllConceptCount(DOC_c)} \right)$$

Equation 5.4: Considering documents number, Concept Importance and Day Recency for the last 30 days in Calculating short-term interest for Concepts

The above equation considers the following:

- Document weight in the calculation.
- Documents count in one day in the calculation.
- Differentiate between the days by considering day recency.
- The number of days

5.2.3.2. Building Short-term Interests Calculating Formula for Edges

The edge weight is used mainly to show how closely two concepts are related in respect to the user, in other words, the extent the user is interested in documents possess both concepts. A first intuition is to consider the number of the documents mentioned both the concepts against the documents that mentioned at least one of the concepts. This yields Equation 5.5:

$$st\_interest(C_1, C_2) = \frac{|documents_{C1,C2}|}{|documents_{C1}| + |documents_{C2}| - |documents_{C1,C2}|}$$

Equation 5.5: Considering only number of Documents in Calculating Short-term Interests for Connections

Where C_1 and C_2 are the two parties of the edge, |documentsC_1|, |documentsC_2| are the number of documents that mentioned concepts C_1 and concept C_2,

respectively, and |documents$_{C1,C2}$| are the number of documents have both the concepts.

The importance of an edge is related to the importance of both its concepts. If concepts of an edge don't interest the user, the edge itself (which means the connection between these two concepts) will not interest the user. However, if the two concepts highly interest the user, the edge definitely will concern him a lot. As a result, one cannot treat all edges as same. The edge is represented by the minimum importance of the concepts participating in the edge in a

$$st\_interest(C_1, C_2) = \frac{\sum_{|docs_{C1,C2}|} MIN(IMP(C_1), IMP(C_2))}{\sum_{|docs_{C1}|} IMP(C_1) + \sum_{|docs_{C2}|} IMP(C_2) - \sum_{|docs_{C1,C2}|} MIN(IMP(C_1), IMP(C_2))}$$

Equation 5.6: Considering number of Documents and Concepts Importance in Calculating Short-term Interests for Connections

single document. The new formula (Equation 5.6) is as follows:

Where IMP(C_1) is the importance of the concept in the document, and MIN(IMP(C_1), IMP(C_2)) is the minimum of the importance of C_1 and C_2 (MIN function choose the lowest importance).

The previous formula seams perfect, however, the formula does not consider the decay of user interests over time. Accordingly, the importance of the day is considered as in the concept interest formula before. So, the formula is going to be as follows:

$$st\_interest(C_1, C_2) = \sum_{i=1}^{30} (dayImportance(Date(i)) * \frac{\sum_{|docs_{C1,C2}(i)|} MIN(IMP(C_1), IMP(C_2))}{\sum_{|docs_{C1}(i)|} IMP(C_1) + \sum_{|docs_{C2}(i)|} IMP(C_2) - \sum_{|docs_{C1,C2}(i)|} MIN(IMP(C_1), IMP(C_2))})$$

Equation 5.7: Considering documents number, Concepts Importance and Day Recency for the last 30 days in Calculating short-term interest for Connections

The previous formulas aren't calculated directly to load the user model. However, part of the above calculation is done while updating user profile and the rest of the calculation is done in loading from user profile to user model task (just the importance of the concept in the day is calculated for the user profile and assigned as value to the attribute SCISD, also the sum of the minimum participant of edges for that day is calculated and assigned to SMP attribute).

The next section discusses the part of calculations done while modeling the user profile.

5.3. ArabAgent User Profile

To obtain the user model, the user profile has to store the sufficient data of the user. One could suggest to store all the data while user accessing the web so we guarantee that we have all the data we need. However, this is not efficient since the profile will get bigger and bigger and the system will consume a lot of resources in storing and processing such data. A more efficient way is to store just data with a sufficient degree of analysis.

In ArabAgent, we have maintained an XML file to store user data. The XML file stores list of concepts (representing nodes of the semantic network) and list of connections (representing edges between nodes in the semantic network). Each concept in the list of concepts contains the days that the concept accessed in. Each day has three attributes; the first attribute, named "date", is the date of the day. The second attribute is the number of web pages that contain this concept in that day, named "*docCount*". The third attribute is the sum of the importance of the concept in each web page contains the concept in that day and named "SCISD".

For connection list, each connection contains the days the connection appeared in. Each connection has two attributes; "from" and "to", to hold the concept ids of the two parties of the connection. Each day in the connection has three attributes as well. The first attribute is the date of the day as in the concept and called "date". The second attribute is the number of documents the two parties

co-occur in and called "*docCount*". The third attribute is the sum of the connection importance for the documents and named "SMP".

These attribute are updated frequently as we will see soon in section 5.3.3.

5.3.1. Calculating the Importance of the Concepts and Connections for A Web Document

Calculating the importance of a concept in a web page is simple. After identifying concepts of the web page, we count the occurrences of the concepts in the web page. The importance of a concept is the concept occurrence in that page divided by all occurrences of all concepts. This importance is similar to term frequency that used in information retrieval to compute the weight of the term. However, instead of considering all terms, we consider only concepts of the page. See Equation 5.8.

$$conceptImportance(c,p) = \frac{occurrenceOfConceptInThePage(c,p)}{occurrencesOfAllConceptInThePage(p)}$$

Equation 5.8: Compute the Importance of a Concept

Where c is the concept we need to compute the importance for, and p is the web page.

The importance of a connection between two concepts in a web page depends on the concept Importance of both concepts in that page. See Equation 5.9.

$$connectionImportance(c_1,c_2,p) = MIN\left(conceptImportance(c_1,p), conceptImportance(c_2,p)\right)$$

Equation 5.9: Compute the Importance of a Connection between two Concepts within a Document

Where c_1 is the first part of the connection and c_2 is the second part of the connection, and p is the web page.

5.4. Updating User Profile

The user profile is updated frequently to adapt to the frequently changing user interests and needs. The calculation of the attributes of both short-term interest and long-term attributes is done through the process of updating the user profile. New concepts may be added to the user profile while updating task as well.

As user provides his relevant set of documents, the user profile is updated accordingly. More importantly, as certain concepts did not get mentioned any more in the set of relevant documents, the system assumes that the user interests of these concepts decline. As a result, a time window technique of size 30 days is used to adapt the user interest attributes automatically to mimic user interest behavior. The interest attribute value gets decayed over time until it reaches value zero i.e. the user has no interest in a topic anymore; although he may have a great knowledge about it.

5.4.1. Preparing Documents for Updating

Updating user profile task starts by creating a graph for the feedback document. To create the graph of the set of the feedback documents, several pre-processing steps to documents need to be accomplished. After downloading and eliciting text form Web page, the concepts should be identified from the documents. The concept identification process isn't direct. Terms that may compose concepts in the future are detected first. Then, a disambiguation process is needed to reveal ambiguous terms i.e. terms with multiple meaning. The pre-process step is explained in great detail in chapter 6.

5.4.2. Generating Feedback Document Graph

After identifying the concepts of each document, each document in the set is then represented as an undirected unweighted graph where just frequency of occurrence attributes values of only concepts is computed. For each document, the occurrence attribute values for the concepts are the numbers of occurrence of the concepts in the document. Each identified concept is connected with all

other concepts within particular document. The system uses the graph of the documents to update the user profile.

5.4.3. Updating the Profile

The updating process starts after preparing the document that is considered relevant and generating the document semantic network. The updating process starts with updating the concepts first. In case a concept does not exist, it adds a new concept to the profile. In the same manner, the system updates the existing edges first, if the edge does not exist in the profile, it adds a new edge with profile. The following discusses the steps of updating the profile in detail.

5.4.3.1. Updating Existing Concept

For each document in the feedback set, update the user profile with the graph of that document. Each concept in the document graph is checked for its existence in the user profile. In case the concept exists in the profile, the existing concept in the profile is checked if it has been visited today. If the concept has been visited today we increase the *docCount* attribute by 1 and add the concept importance value of the current concept to the old value of SCISD attribute (the concept importance value equals the concept occurrence frequency in the document divided be the sum of concept occurrence frequency of all concepts as in Equation 5.8). In case the concept hasn't been visited today, a new day element is added for the concept with date value equals today date, *docCount* value equals 1 and SCISD value equals the value of concept importance.

5.4.3.2. Adding New Concept

In case the concept does not exist in the user profile, (the concept is totally new), a new concept element is added to the profile with a new day representing today where the value of date attribute equals today's date, the value of *docCount* attribute equals 1 and SCISD value equals the value of concept importance.

5.4.3.3. Updating Existing Edges

After adding all the concepts to the profile we add the edges of the graph into profile. Each edge in the document graph is checked for its existence in the user profile. In case the edge exists in the profile, the existing edge in the profile is checked if it has been visited today. If the edge has been visited today we increase the *docCount* attribute of last day element (which is today) by 1 and add the minimum concept importance value of concepts participating in the edge to the value of SMP attribute in the profile. In case the edge hasn't been visited today, a new day element is added for the edge with date value equals today date, *docCount* value equals 1 and SMP value equals to the minimum concept importance value of concepts participating in the edge.

5.4.3.4. Adding New Edges

In case the edge is not existing in the user profile, (the concepts could exist but the edge is totally new), a new edge element is added to the profile. The attributes "to" and "from" are equal to the concepts participating in the edge. A new day representing today is added to the edge element where the value of date attribute equals today's date, the value of *docCount* attribute equals 1 and SMP value equals to the minimum concept importance value of concepts participating in the edge.

5.4.4. Loading User Model from the User Profile

These are the steps for loading the user model:

1. For each concept (c) in the profile do the following:
    a. make a corresponding node in the semantic network
    b. conceptImportance = 0
    c. for each day (d) for the concept (c) do the following:
        i. conceptImportance = conceptImportance + (importanceOfDay(d) * SCISD)

2. For each connection (e) in the profile do the following:
    a. add two edges; edge(e.to, e.from) and edge(e.from, e.to) to the semantic network because the edges are undirected
    b. edgeImportance = 0
    c. for each day (d) for the connection (e) do the following:
        i. edgeImportance = edgeImportance + (importanceOfDay(d) * (SMP/(conceptImportance(e.to, d) + (conceptImportance(e.from, d) - SMP))

The importance of day is computed as the difference between the day of today and the day of the date in the "date" attribute of the concept or connection.

5.5. ArabAgent Personalization Techniques

The ArabAgent Agent provides three forms of personalization; news filtering, tailoring web search results, and modifying search queries.

5.5.1. Personalized News Filtering

The ArabAgent system provides filtering techniques for some news websites. The ArabAgent checks for news availability in the websites every five minutes. The ranking and filtering component filters the new news article and if the article is likely to be relevant, it sends the article to the user email.

5.5.2. Personalized Search

Most of the web personalization systems that deal with Web search as its application have conquer the problem by either treating web search result or the user original query. The systems that deal with search result use variety of methods such rank again, or so-called re-ranking, filter, or tag the result set of the search engine.

Due to time constraints, it is difficult to ranking again all the result items of the search engines. The user can't tolerate the relatively long time the system takes to rank all the results. Furthermore, the results are going to be restricted to the result set which the search engine show to you, in response for user query which may poorly represent the user needs missing a great opportunity to find other good web pages that may satisfy the user needs.

Two systems treated the problem differently; Syskill & Webert [Pazzani M., 1996] and PEA [Montebello M., 1998]. They tagged each item of a search engine result by a tag to indicate its relevancy likelihood to a user. The user gradually gains trust on the system as he/she search again and again, as the system's tags meet the user's expectations, (since he could evaluate the tagged results while he wade across the result set). However, later, the user finds himself wades across hundreds of result links pursuing the ones tagged with high relevancy likelihoods.

To resolve this problem in ArabAgent, we have combined both techniques (results tagging and result re-ranking) but with a different way. As user submits his query to the search engine, the system shows the original order and rank of the search engine result. However, the result links are tagged with the relevancy likelihoods to user. In addition to tagging, we provide the ArabAgent special rank of the result in a sidebar that the user could show or hide.

In the sidebar, the links of a result are shown immediately as they are retrieved. There is no necessity to wait for all the links to be ranked to show them all as a bunch. The links appear to the user in one by one manner, as they are weighted by the ranking and filtering component. As new result link is given a weight, the order of the result changes immediately, so the user notices that some result items' positions are changing and others are subsiding (some sink and the others emerge to the top of the result list). This technique of displaying the result prevents the user from feeling the tardy arrival of the

result. Furthermore, this technique prevents user wading across hundreds of result links pursuing the ones tagged with high relevancy likelihood.

5.5.2.1. Query Personalization

The methods and techniques that deal merely with search result set has great disadvantages. Although, it tries to improve the effectiveness of the system by improving its precision at particular rank, it totally ignores the recall measure. The reason behind that is that the system is confined only to the set of results returned by a certain query. This could cause decline in the overall system performance in case the user submits bad query keywords (the result sets contains no relevant data at all). Accordingly, some systems opt to personalize the search by modifying the user original query to avoid these drawbacks.

In ArabAgent system, the query customizer is the responsible component for modifying user query to suit his/her user need. It resolves the previous problem by introducing modified query to the search engine. It could expand user query or alter some keywords or search. The query customizer could narrow the search by limiting it to specific websites (for example websites the user likes to visit).

It seems that Modifying user query is good solution to the disadvantages of techniques handle only search result. However, modifying user query is tricky and difficult. You cannot augment new keywords to the user original query unless you are hundred percent sure they are going to improve the result. The keywords could take the query to another direction causing the result to decline in terms of both precision and recall.

5.6. Computing Document Relevancy Likelihood Weight

A graph comparison algorithm is used to compare the semantic network of a web page against the semantic network of the user model. The algorithm is adopted from [Sorensen H., 1997]. The used algorithm gives advantage of concepts that co-occur over individual concepts by comparing between the

structures of semantic networks of the Web page and the user profile. The algorithm takes into consideration the frequency of occurrence of certain concepts within the web page, the frequency of co-occurrence of certain concepts within the web page, and the user interests of these co-occurrences.

Although the user model represent both short-term and long-term interests, our algorithm utilizes only the short-term interest in comparing between the semantic network of the user and the semantic network of the document.

The comparison utilizes a spreading activation algorithm to highlight portions of graphs common to both the user model and the web page. The algorithm consists of setting the activity levels of nodes in the local web page graph that are common to both web page and user profile graph. The activity is then *leaked* out to its neighboring nodes, depending on these neighboring nodes' occurrences (and their position and weight) in the user model graph. The leaking of the activity is controlled in such a manner that priority is given to co-occurrences of concepts that are common to both. A formal illustration of the algorithm is as following.

Identify the concepts of the article to be concepts and initialize all nodes of the article network each to the corresponding concept importance.

If $\mathcal{P}(v, \varepsilon)$, $\mathcal{A}(\acute{v}, \acute{\varepsilon})$ are profile graph and article graph, respectively, then we can write this as:

$$\forall n_i, n_i \in \acute{V}\colon (Wt_A(n_i) \leftarrow conceptImportance(n_i)).$$

Equation 5.10: Initializing Nodes with Concept Importance

Set the activity level of all nodes that occur in both article graph and the user model to the weight of that same node in the user model multiplied by the article node weight as in Equation 5.11:

$$\forall n_i, \Big(\big(n_i \in \acute{V}\big) \wedge (n_i \in V)\Big)\colon (Wt_A(n_i) \leftarrow [Wt_P(n_i) * Wt_A(n_i)])$$

Equation 5.11: Activity Level Considers Concept Importance and User Interest

Next, spread the weights out to the neighboring nodes in the article graph using the *spreading activity mechanism.* This helps increase the relevancy of an article containing such co-occurrence. Note that nodes with no connection at all in the user model will remain with the same values. This guarantee that nodes that don't co-occur with each other remain with a positive weight but are not increased. We perform this as in Equation 5.12:

$$\forall n_i, \left((n_i \in \acute{V}) \wedge \left((n_i, n_j) \in \acute{\mathcal{E}}\right) \wedge \left((n_i, n_j) \in \mathcal{E}\right)\right) : \left(Wt_A(n_j) \leftarrow \left(Wt_A(n_j)\right) + \left(\left(Wt_A(n_j)\right) \times \left(Wt_P(n_i, n_j)\right)\right)\right)$$

Equation 5.12: Spreading Activity of the Node

The similarity measure between the graph representation of the profile and the article is calculated as in Equation 5.13:

$$sim\left(\mathcal{P}(v, \varepsilon), \mathcal{A}(\acute{v}, \acute{\varepsilon})\right) = \frac{\text{sum of the weights of the common nodes}}{\text{sum of the weights of all node in the article graph}}$$

Equation 5.13: Similarity Measure between the Article and the User Profile

Summary

This chapter discusses and illustrates the techniques used to apply the functions of the components of *ArabAgent* framework that has been mentioned in the previous chapter.

ArabAgent system applies explicit techniques to focus mainly on the task of user modeling and expressing user context without being concerned by the efficiency and accuracy of the implicit techniques used to judge on the relevancy of an item or object the user inspect. ArabAgent provides two ways for explicit feedback; while searching and while browsing the Web.

In ArabAgent, the user profile is stored in as an XML file containing partially analyzed data of the user (this level of analysis is just for reducing modeling time). The user model comprises of semantic networks that represents user's knowledge, long-term interests and short term-interests.
The user profile is updated frequently to adapt to the frequently changing user interests and needs. A time window technique of size 30 days is used to adapt the user interest attributes automatically to mimic user interest behavior. The interest attribute value is get decay over time until it reaches value zero i.e. the user has no interest in a topic anymore; although he may has a great knowledge about the topic.

ArabAgent attempts to represent the user's interests, knowledge and attitudes to define the importance of a web page to the user as a way to individualize the access to the web. To make the user model reflect the above envision, two structures have been combined together to comprise the user model; a hierarchy of categories and semantic networks. The updating formula considers the following:

- Document weight in the calculation.
- Documents count in one day in the calculation.
- Differentiate between the days by considering day recency.
- The number of days

The ArabAgent Agent provides three forms of personalization; news filtering, tailoring web search results, and modifies search queries. ArabAgent combines two forms of personalization search results; results tagging and result re-ranking.

A graph comparison algorithm, adopted from [Sorensen H., 1997], is used to compare the semantic network of a web page against the semantic network of the user profile.

ArabAgent depends on the “meta” tag to identify the encoding of the web page. Then, the concepts are identified using the text processing component.

CHAPTER 6 : TEXT PROCESSING OF ARABAGENT

The text processing component is a key component in the ArabAgent. This component uses novel techniques in treating the Arabic text. This chapter illustrate in great details the mechanism of techniques used to processing the Arabic text.

The techniques used to process text depend on controlled vocabulary extracted from the Arabic Wikipedia project as well as data extracted from the link structure. In section (6.1) we show the methods and heuristics used to extract both controlled vocabulary and the statistical data extracted from the internal link structure of Arabic Wikipedia project. Sections (6.2) and (6.3) demonstrate the techniques used to process the Arabic text.

6.1. Preparing Data from Wikipedia Database Dump

Before phrase detection and phrase sense disambiguation tasks could be supported and user profile can be built, the data and statistics that are used in both tasks should be elicited and prepared from Arabic Wikipedia dump. The aim of this process is to prepare the files used by the text processing component in order to perform its tasks.

The data and statistics that are extracted are used in both text processing task and for building the user model. The user model consists of two structures a graph that connects between concepts that the user is interested in and a hierarchy of categories that embraces the concepts interested in by the user. We first define what Arabic Wikipedia database dump is. Then, we demonstrate the output of the task of preparation. The output of the preparation task is a set of files that have the needed data and statistics for both text processing and modeling user. Section (6.1.3), explains how we

extract these data and statistics from the database dump. The section first explains how we extract the concepts then the relatedness statistics and finally how to build the hierarchy of categories and bind the concepts with their categories.

6.1.1. The Arabic Wikipedia Dump

Wikipedia offers free copies of all available content to interested users. These databases can be used for mirroring, personal use, informal backups, offline use or database queries. All text content is multi-licensed under the Creative Commons Attribution-ShareAlike 3.0 License (CC-BY-SA) and the GNU Free Documentation License (GFDL). Images and other files are available under different terms, as detailed on their description pages.

A database dump contains a record of the table structure and/or the data from a database and is usually in the form of a list of SQL statements. A database dump is most often used for backing up a database so that its contents can be restored in the event of data loss. Corrupted databases can often be recovered by analysis of the dump. Database dumps are often published by free software and free content projects, to allow reuse or forking of the database. Dumps from any Wikimedia Foundation project found here; http://download.wikimedia.org/

The Arabic Wikipedia project dump has several files; about 30 files. Most of these files are SQL command that could be used later to generate tables structure and populate them with data. Other files contain statistical data about the project. Example of such files is "pages-articles.xml.bz2" compressed file which contains current versions of article content wrapped in some XML. Another example is "page.sql.gz" compressed file which consists of a list of SQL statements that could be used to generate both table schema and table data. However, in our preparation process we don't use all the 30 files of the dump; we only use the following files:

- pages-articles.xml.bz2
- page.sql.gz

- pagelinks.sql.gz
- redirect.sql.gz
- category.sql.gz
- categorylinks.sql.gz
- site_stats.sql.gz

The first file, mentioned above, contains current versions of article content wrapped in some XML. Each of the remaining files is representing a database table of the MediaWiki database schema (the software that runs the Wikipedia projects). See the database schema in Figure 6.1.

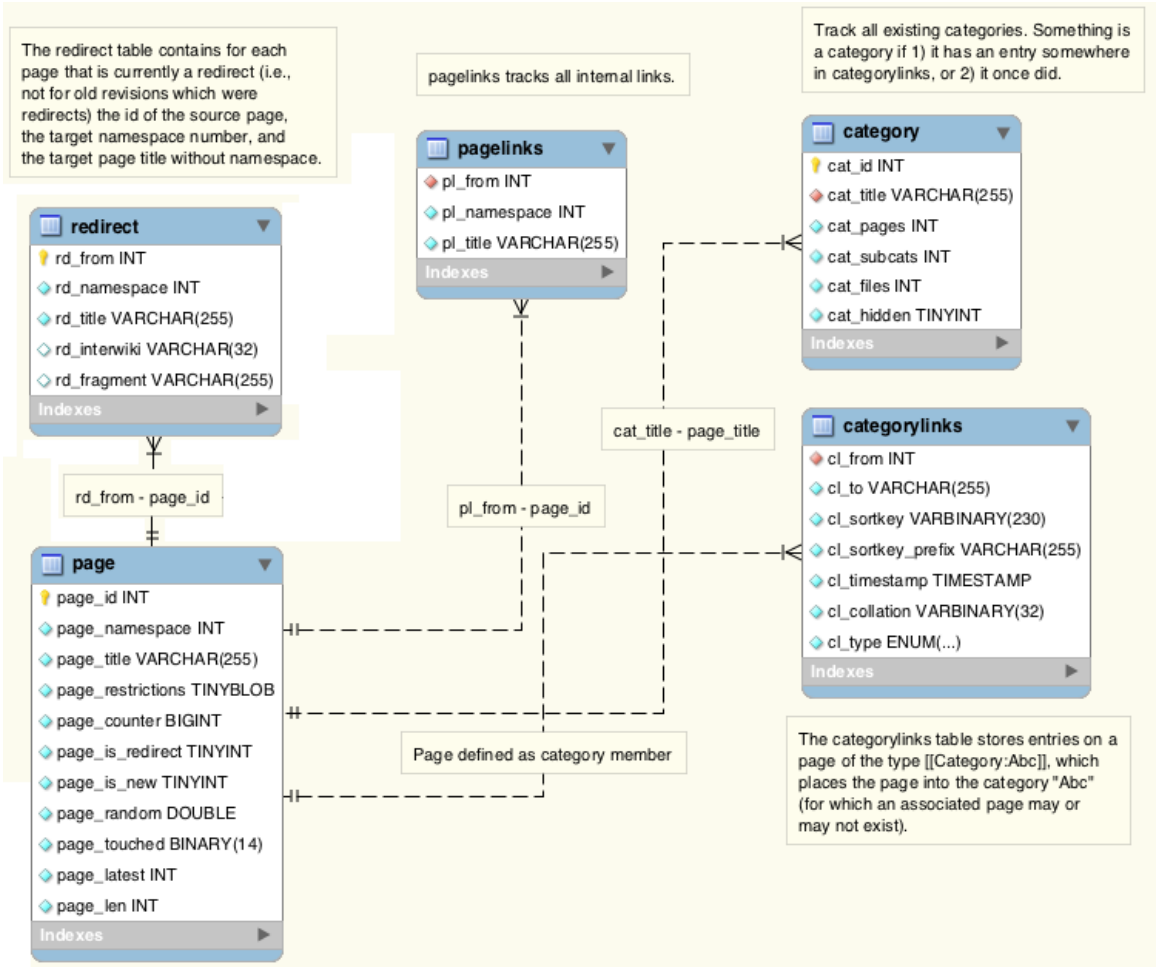

Figure 6.1: Part of Database schema diagram for MediaWiki as of version 1.17

The "page.sql.gz" compressed file contains all the pages that exist in Wikipedia. The pages are not all articles; there are other kinds of pages such as user pages, discussion pages, Wikipedia help pages and others. Each page type is called a namespace and given an id. The namespace id of the articles in the Wikipedia is 0. The table has the following fields (only the important fields are described):

- page_id: Uniquely identifying primary key.
- page_namespace: This field contains the number of the page's namespace.
- page_title: The sanitized page title, without the title of its namespace. It is stored as text, with spaces replaced by underscores. The real title showed in articles is just this title with underscores (_) converted to spaces ( ).
- page_restrictions
- page_counter
- page_is_redirect: A value of 1 here indicates the article is a redirect; it is 0 in all other cases.
- page_is_new
- page_random
- page_touched
- page_latest
- page_len
- page_no_title_convert

The compressed file "pagelinks.sql.gz" contains all internal links in Wikipedia. Each entry contains the source page's ID; "pl_from" field, and the namespace (number); "pl_namespace" field, and article name (in text); "pl_to" field, that is being linked to within that source page. There may be many instances of the source page's ID, as many as the internal links within it, but there can be only one entry per internal link for any page ID. Note that the target page may or may not exist, and due to renames and deletions may refer to different page records as time goes by.

In Wikipedia project, a redirect page is a page that automatically redirects the reader's browser to a specified target page. Redirects are used to help users locate information and keep wikis organized, so that multiple names, abbreviations, misspellings, or related topics can all point to the same page. The "redirect.sql.gz" compressed file contains for each page that is currently a redirect. The table generated from this file contains the fields "rd_from" which the id of the source page, "rd_namespace" which is the target namespace number, and "rd_title" which is the target page title without namespace. Note that the target page may or may not exist.

All the categories of Wikipedia project is contained in the compressed file "category.sql.gz". The table produced from the file contains the following fields:

- cat_id: Uniquely identifying primary key.
- cat_title: Name of the category, in the same form as page_title (with underscores).
- cat_pages: Number of pages in the category
- cat_subcats: Number of sub-categories in the category
- cat_files
- cat_hidden

The table that generated from the compressed file "categorylinks.sql.gz" links the pages to their categories. The table also connects sub-categories to their categories. The table has the following fields:

- cl_from: Stores the page_id of the article where the link was placed.
- cl_to: Stores the name (excluding namespace prefix) of the desired category. Spaces are replaced by underscores (_)
- cl_sortkey
- cl_timestamp

- cl_sortkey_prefix
- cl_collation
- cl_type: identify the type of page (page, subcategory, or a file) associated with the category; takes the values 'page', 'subcat' or 'file'.

Finally the file "site_stats.sql.gz" contains some statistics about the dump itself such as number of articles, number of views, number of edits, number of page total pages number of users, total pages number of admins, total pages number of images, and finally total pages number of active users. In our preparing process we only need the number of articles.

6.1.2. The Output Files

The process of preparing data and statistics from Wikipedia database dump results in the following files as its output. These files are used directly by the text processing component:

- concept_inlinkCount.txt; this file maintains the concepts extracted from Wikipedia and the number of articles back-linking their corresponding articles. The concepts were essentially articles in the Wikipedia so normally they have back-links from other articles.
- concept_outlinkCount.txt; this file contains the concepts and number of hyperlinks in their corresponding article in Wikipedia.
- concept_inConceptList.txt; this file contains the concept ids given to their corresponding Wikipedia articles and the list of concepts corresponding to the list of articles connecting to these articles.
- concept_outConceptList.txt; this file contains concept ids given to the articles and the live wikilinks of the corresponding article
- term_conceptsList.txt; this file contains all the terms of the Wikipedia and their potential concepts.

- concepts.xml; this file contains all the concept ids and their alternative synonyms.

6.1.3. Preparing Process

This subsection illustrates the algorithm of eliciting and preparing these data and statistics from Wikipedia dump database files. This section will show you how the mentioned files above are prepared.

6.1.3.1. Eliciting Concepts form Wikipedia

The idea behind extracting concepts from Wikipedia is based on the fact that each Wikipedia article usually describes a single topic or entity. The article titles in Wikipedia are unique. Here, the Wikipedia's articles are used as concepts; each article in the Arabic Wikipedia project is corresponding to one concept that represents it.

Each Wikipedia article usually describes a single topic or entity. The article titles in Wikipedia are unique. Here, the Wikipedia's articles are used as concepts; each article in the Arabic Wikipedia project is corresponding to one concept that represents it. Each concept takes an identifier, concept id, which is used later in the text processing task. Another kind of pages in Wikipedia project is called redirect pages. These redirect pages represent other names (synonyms) that the article could take; each redirect page represent another different name of the article. Each single article page could have many redirect pages and each redirect page points to only one article page. The redirect pages have been used to form part of the synonyms of the concepts. An article may mention many other articles' names throughout its text. Also, an article could be referred to by many articles. Since the context that a referenced article could vary from one refereeing article to another, the referenced article's name could take different morphological forms and/or have different affixes. These forms are considered the rest of the forms to represent the concept underlying the referenced article.

After uncompressing the files, the first step is to prepare the files by extracting the useful data (from our system point of view) from these files which has the

form of SQL commands. The first file to prepare is the "page.sql.gz". The data is usually extracted from the "INSERT INTO" SQL statement (this is true also for all other files in the form of SQL commands such as pagelinks, redirect, category, categorylinks, and sitestats). The preparation step involves removing all the commands of the SQL and keeping only the data.

After preparing the data to be used, the system creates the concepts (senses). The creation of the concepts first involves removing namespaces other than articles namespace which has the id of 0. The namespace 0 does not only have the article pages but also has redirect pages and disambiguation pages (these are pages that help user disambiguate his term and direct him to the right sense of his term, during search on Wikipedia). Therefore, the system removes the disambiguation pages as well since they don't represent any entity or idea; they merely used for clarifying the meaning of user query. The redirect pages are removed too for the same reason, so the remaining pages are just the article pages.

Since redirect pages are used to provide multiple names, abbreviations, acronyms, misspellings, and synonyms to be pointing to the same article, we uses this redirect pages to be the various synonyms for the concept. The redirect pages exist in the "page.sql.gz" file and the links between the redirect pages and article exist in the "redirect.sql.gz". Before using them directly as synonyms, the file (redirect.sql) needs more preparation. For example, there are some redirect pages pointing to other redirect pages, also there are some redirect pages pointing to themselves and some redirect pages redirect to pages belong to other namespaces than 0.

The preparation steps of the redirect pages start with removing all redirects that pointing to pages of namespaces other than namespace 0. Then the system removes redirect pages that pointing to themselves. For redirect pages that pointing to other redirect pages, we follow these links, if they lead us to an article we keep them all by replacing the target of each link by the article name in the page file; else (lead us to page not in namespace 0 for example) we remove them.

Next, after preparing the redirect links to a page, we consider the names of the redirect pages as synonyms for the article names they point to.

Sometimes when you edit an article in Wikipedia and you want to refer to another article, you want to refer to them by the same name or by another name more suitable to the context of sentence you are writing. For example, the article named "مصر" -Egypt in Arabic- in the Arabic Wikipedia has the phrase "للبحر الأحمر" as a wikilink (as inner link between Wikipedia articles) in the first paragraph pointing to the article "البحر الأحمر". The context in the article "مصر" has forced the article name "البحر الأحمر" to change to "للبحر الأحمر" to suit the context of the sentence. Two ways to do that in Wikipedia; the first one is by creating a redirect page as we mentioned above, and the second way is just by adding the pipe "|" divider followed by the alternative name in order to link to an article, but display some other text for the link to the reader.

We have taken advantage of such properties to gather more morphological forms of the articles' names. This property provides us with different morphological forms of articles' names, more synonyms, acronyms, and more different forms agglutinated with prefixes and suffixes. Accordingly, we have parsed the "pages-articles.xml" file to collect theses forms and consider them as second set of synonyms.

After collecting the synonyms using the above two methods, we normalize all the synonyms using the unified normalization technique by removing all short vowels, Kashida, punctuation and non-letters. It replaces the TAA MARBOUTA "ة" with HAA "ه" and ALEF MAKSOURA "ى" with YAA "ي", and replaces "أ،آ،إ" with "ا". Finally, the concepts and their synonyms are stored as XML file named "concepts.xml" (as one of the output file).

6.1.3.2. Extracting the Relatedness Statistics between Concepts

The analysis of Wikipedia link structure starts by building a table that shows the links between two concepts. This table is built from the file "pagelinks.sql" (the file that holds the links between all articles). First, we remove all the links that have their targets not belong to namespace 0. Then we remove the targets

that have their page title not exist as synonyms. After that we replace the page titles of targets by their page ids. Finally, we replace the page ids of both sources and targets of links by the concept ids from the "concepts.xml" file.

After replacing the page ids of both sources and targets of links by the concept ids, remove any duplicates of links. Then, we assure that each concept has a link to itself. For each distinct target concept of a link in the concepts' links table, we generate a list of its source concepts and save two output files; the "concept_inConceptList.txt" file which contains all the distinct concepts that appear as target concepts in concept links table each with its source concepts, and the second file is "concept_inlinkCount.txt" which contains each concept appears as a target concept in the concept links table associated with the number of times it appears as target.

The files "concept_outConceptList.txt" and "concept_outlinkCount.txt" are similar to the previous two files "concept_inConceptList.txt" and "concept_inlinkCount.txt", but instead considering the target concepts they consider the source concepts. For each concept appears as a source side of the links in concept links table, consider all the concepts appear as target for them. The file "concept_outConceptList.txt" consists of all distinct source concepts with their list of all concepts appear as targets, and the file "concept_outlinkCount.txt" consists of all distinct source concepts with their number of concepts appear as targets for each.

After generating the statistical data of the internal link structure, which used later in text processing component for disambiguation task, build the inverse file of the "concept.xml" file. The "concept.xml" file consists of concept ids each with its different synonyms. However, while phrase detection process, the senses or concepts need to be determined for each detected phrase. Accordingly, we built the "term_conceptsList.txt" file which includes for each phrase it’s a list of all possible senses found in Wikipedia for this phrase.

6.1.3.3. Building User Hierarchy of Categories

The Wikipedia project allows each page (whether it is article or not) to belong to one or more categories. Each category could embrace one or more pages.

Some categories may exist for organization reasons (i.e. this makes some categories don't have pages). The categories of Wikipedia projects form an acyclic directed graph. Each categories could have one or more super categories (or parent), and may have one or more sub-categories. We use the file "category.sql" to extract the categories. The file "categorylinks.sql" is used to link concepts to their categories.

Since the graph is "acyclic directed graph", we use the term hierarchy for sake of simplicity.

Since we need only categories that classify article pages and their super categories until the root of the hierarchy, we need to eliminate other categories that used for other pages. Accordingly, since we know the pages that are used as concepts, we maintain only the categories that classify these pages. Using the "categorylinks.sql" file we keep only those links that their "cl_from" attributes belong to the set of page ids of articles that used as concepts. We extract the categories names from the field "cl_to" of these links. To determine the parent categories of categories in hand a new iteration is conducted to determine their parent using the page id of these categories. We conduct more iterations until there is no more categories found.

Here are the steps in a compact form:

1. Prepare the needed Arabic Wikipedia files by
    a. Preparing “page.sql” file
    b. Preparing “pagelinks.sql” file
    c. Preparing “redirects.sql” file
    d. Preparing “categorylinks.sql” file
2. Then create concepts by
    a. Remove namespaces other than namespce 0
    b. Remove disambiguation pages
    c. Remove redirect pages
    d. Prepares in redirects
    e. Add synonyms to concepts file
        i. Add synonyms from redirects

ii. Add synonyms from articles

f. Normalize synonyms

3. create links statistics
   a. Building Concept Concept Link Table
   b. remove duplicate same concept link
   c. create cc_inlink_count.txt file
   d. create concept_inconcept list index
   e. create cc_outlink count
   f. create concept_outconcept list index
4. generate_term_concepts_list_index

6.2. Web Pages Processing

This component does the following:

1. After downloading the feedback document, encode the page
2. Remove tags and unwanted scripts from the text of Web page
3. Identify the concepts of the text using the text processing component.

See the following figure.

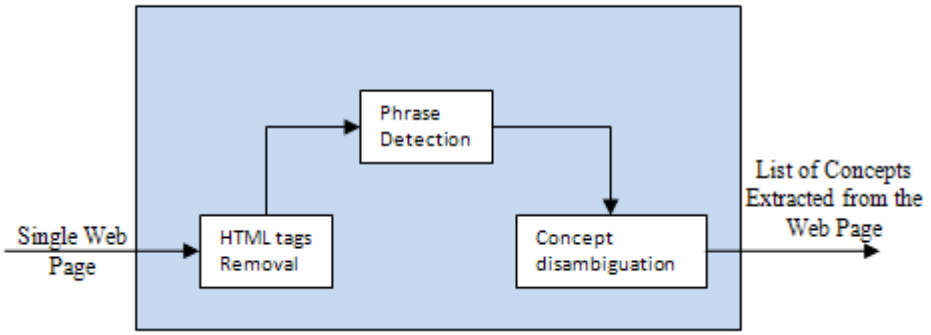

Figure 6.2: Text Processing Component of ArabAgent System

6.2.1. Downloading Web Page

One of the main problems is to identify the encoding of document that is feed to the system. Several approaches have been used. However, the most successful one is the method that depends on machine learning to identify the encoding [Manning C., 2008]. In ArabAgent we have used a simple and effective method other than machine learning to identify the encoding of the web page. Since the documents feed in the ArabAgent system are web pages. The ArabAgent system takes advantage of the feature of html “meta” tag presented in the beginning of most of the Web pages. We first download the web page encoded in utf-8 encoding. Then, we check its "*charset*" attribute in the "meta" tag, if exist it is value is considered as the page encoding. If it does not exist, the page is encoded utf-8 encoding. This is because there is a convention that a page that doesn’t have a charset attribute always be encoded in utf-8 format by the browser.

6.2.2. Removing Tags and Scripts

Once the html string is available, we remove the all tags from the string as well as the JavaScript string and CSS string. The keywords inside the "meta" tag are added to the text again.

After preparing the text of the web page, the text is sent to text processing component to identify its concepts for later either update user profile or personalize the user experience.

6.3. Text Processing of the ArabAgent

This section describes the steps of processing Arabic text in great detail; starting from term detection in subsection 6.3.1 and ending with the disambiguation senses or concepts of the phrases task in subsection 6.3.2.

The technique could be used as an additional step for processing text within the information retrieval systems. It could be used with normalization and stemming techniques. It handles synonymy, polysemy and the other aforementioned problems in section (3.2).

6.3.1. Term Detection

The term detection step goes as follows: after tokenizing the document, the tokens are normalized; using the unified normalization [Eldesouki M., 2009]. The document then is processed to generate word n-grams. The n-gram generation process differs from the usual way of producing n-gram[38]; the concept ids are assigned during the n-gram generation process. See the algorithm in Figure 6.3. While the system generates n-grams, it tries to match the n-gram to the synonyms of each concept and assign the concept id(s) to the term in case a match occurs. Terms or n-grams of several concept ids are saved for disambiguation in the second step. The size of the n-gram, n, is equal to length of synonym with maximum length. Although, there is little likelihood to produce wrong phrases, the customized method for generating n-gram has the advantage of reducing ambiguity by trying to produce longer phrases first.

The stopwords removal process begins after the detection process, and the reason for that is some phrases may contain stopwords, which will not be matched if we remove the stopwords before the n-gram process. For example, the phrase "مهرجان كان" if we remove stopwords before n-grams process the word "كان" is going to be removed, but if we removed stopwords after n-gram process nothing going to be removed. (In Wikipedia, stopwords don't have corresponding articles, so stopwords don't exist in the synonym-concept_ids dictionary or syndic as it is mentioned in Figure 6.3).

[38] instead of constructing n-gram incrementally (by considering one word as n-gram then two words then three words till n words as n-gram), we construct n-gram decrementally (by considering the n words as n-gram first, then n-1 words as n-gram till using 1 word). This way gives the chance to detect longer phrases (which means less ambiguities) first even if the there could be short terms.

Input: **TokensQ** (queue of all document tokens), **synDic** (synonym-concept_ids dictionary which is named in section 6.1.2 as term_conceptsList.txt file)

Output: list of terms, each with its prospective concept(s)

Algorithm:

1) If TokensQ size = 0, then return;
2) Else If TokensQ size >= n, Choose first **n** (size of n-gram) tokens from the TokensQ into a list; named **nList**.
3) Else, choose **all** tokens from the TokensQ into **nList**.
4) Constitute a term by concatenating all the tokens in **nList.**
5) Try to find a corresponding **variant** for the term in **synDic**.
6) If (a variant found in synDic)
   a) Assign the **concept_id(s)** to the term.
   b) Empty **nList** and dequeue the tokens of the phrase from the TokenQ.
   c) Go to step 1.
7) Else (the term has no corresponding variant)
   a) Then remove one token from the end of **nList**.
   b) Check the size of **nList** after removal
      i) If number of tokens that exist in **nList** = 0, then dequeue the last removed token from TokenQ and go to step 1.
      ii) If number of tokens that exist in **nList** > 0, then go to step 4.

Figure 6.3: Algorithm of generating n-grams each with its prospective concept(s)

Many terms may refer to the same concept, so the appearance of one of them in the document makes it subject to be replaced by the equivalent concept id; this solves the problem of synonyms. In case of polysemy problem, phrases might lie under several concepts. This requires a technique to disambiguate between the several concepts and choose the right one. The techniques are illustrated in the next section. As a result unwilling match is prevented with the same spelling but with different in meanings phrase.

After replacing all the phrases and terms by their right concept id, we treat the document as if it is “bag of words”; however, it is actually a page of concepts. The rest of the information retrieval steps remain the same. An extra step is to

go through the previous steps manually for substituting the query's phrases by the intending concepts ids.

6.3.2. Phrase Sense Disambiguation

If an n-gram has multiple concepts, then a concept disambiguation is going to happen. The disambiguation process starts after the phrase detecting process is completed for the document since the disambiguation process depends on all the phrases which posses a single concept of document; context terms. Multiple techniques have been used for evaluation and comparison purposes. All techniques used in experiments depend on the Arabic Wikipedia statistics such as in-links and out-links of an article, number of out-links of an article and others. The disambiguation techniques that have been tried are inspired from Milne and Witten [Milne D., 2008a; Milne D., 2008b] and Cilibrasi and Vitanyi [Cilibrasi R., 2007]; however, neither machine learning techniques nor "link probability" are used.

The techniques used to disambiguate could be categorized in to two sets of methods, the first set contains one technique which chooses the most common sense among the senses of a phrase, the other category or set (which includes three techniques) depends on so-called semantic relatedness. The semantic relatedness could be computed based on in-links, out-links, or both the in-links and out-links of a Wikipedia article.

6.3.2.1. Most Common Sense

Different techniques have been examined to disambiguate between concepts and each has been evaluated. First technique is choosing the most common concept among all concepts. The commonness measure depends on number of articles refereeing to the article that representing the concept and is calculated by dividing the number of in-coming links to the page representing that concept divided by the sum of numbers in-coming links of all concepts being disambiguated.

6.3.2.2. Semantic Relatedness

The disambiguation techniques depend on so-called *semantic relatedness* between concepts. Different techniques have been examined to calculate the semantic relatedness between concepts. To disambiguate between different senses of a phrase, we used the following algorithm:

1. After detecting all the phrases of the document, all the context terms are identified

2. Then, the weights of the context terms are identified (the weight of the context term show its contribution in the context of the document i.e. how much it represent the document), see the next subsection for more detail on calculating the weight.

3. Then, for each non-context term disambiguate it as follows

    a. set the AWR_Max = 0 and sense = null

    b. for each candidate sense do the following:

        i. calculate its 'average of weighted relatedness' (AWR) with all the context terms

        ii. if the AWR > AWR_Max, then set AWR_Max = AWR and set sense = s, else continue to next sense of the phrase

    c. return sense (the sense with greatest relatedness value)

This algorithm is the same for all techniques that use semantic relatedness to disambiguate senses of the phrases. The difference between them is in the way of calculating the semantic relatedness for each technique. Subsection 6.3.3 explains in great detail the different ways of calculating the semantic relatedness.

6.3.2.2.1. Computing the context term weight

Context terms are not the same; some of them are quite expressive than others. Accordingly, it is unfair to make them contribute with the same effect. As a result, to maintain a degree of consistency with the main context of text, a weight is calculated for each context term depends on its degree of closeness to the context.

The weight is calculated by dividing the sum of the semantic relatedness with other context terms by the number of context terms – 1, as in Equation 6.1.

$$contextTermWeight(ct) = \frac{\left(\sum_{i=0}^{|contextTerm|} semanticRelatedness(ct, contextTerm(i))\right) - 1}{|contextTerm| - 1}$$

Equation 6.1: Computing the context term weight

6.3.2.2.2. Calculating the 'Average of Weighted Relatedness' for a Prospective sense

To decide which sense is the right sense to choose, the system calculates the so-called "Average of Weighted Relatedness" for each sense of the phrase and then chooses the sense with the greatest value. See the algorithm in section 6.3.2.2.

This average is calculated by aggregating the product of the semantic relatedness between the sense and the context term and weight of this context term for each context term. As stated by the Equation 6.2.

$$AverageOfWeightedRelatedness(S) = \frac{\sum_{i=0}^{|contextTerm|} contextTermWeight(i) * semanticRelatedness(S, contextTerm(i))}{|contextTerm|}$$

Equation 6.2: Average of Weighted Relatedness

Where S is a sense of several senses for the phrase

There are three other techniques used for disambiguation depends on the semantic relatedness between the candidate concept and the surrounding context terms. A general theme is to choose the concept with highest average of semantic relatedness with context terms. Since the context terms are not the same in their representation of the context of a document, the semantic relatedness of the context terms are weighted. The weight expresses the importance of the context term to the document by averaging the semantic relatedness between the desired context term and all other context terms; for more details and examples about weighting the semantic relatedness with context terms please refer to [Milne D., 2008b].

The later three techniques have the same calculation theme. However, they differ on the way of calculating the semantic relatedness. The first depends on the in-links counts, the second depends on out-links count and the third depends on the average between both the first and the second ways; review [Milne D., 2008a].

6.3.3. Semantic Relatedness

Semantic Relatedness is a technique to compute the relatedness or similarity between two concepts or senses. The semantic relatedness used as part of the phrase sense disambiguation process in the ArabAgent system. In ArabAgent, the semantic relatedness is applied using several techniques. All the techniques discussed here are taken from Milne and Witten [Milne D., 2008a; Milne D., 2008b]. The ArabAgent computes the semantic relatedness using "most closely related pair using in-links", and "most closely related pair using out-links" techniques. The ArabAgent could also combine between these two techniques to leverage both in-links and out-links of the Wikipedia articles.

6.3.3.1. Semantic Relatedness, What does it Mean?

Since concepts are Wikipedia articles in the first place, the semantic relatedness between two concepts is essentially a task of finding relation between their two Wikipedia articles. Two techniques used to find this relation between the two articles; the first depends on the hyperlinks (called the wikilink) found within the two article of Wikipedia, or so-called out-links. This technique compares between the two set of wikilinks found within both articles and as the percentage of the shared number of wikilinks increases as the relatedness of the two articles (and the two concepts in turn) increase.

The other technique to compute the semantic relatedness is similar to the one above but instead of using the wikilinks we use the back links, or so-called in-links. Also, here the technique compares between the in-links and as the percentage of the shared number of in-links increases as the relatedness of the two articles (and the two concepts in turn) increase.

Another difference between the technique uses out-links and the technique uses the in-links is the way to compare between the lists of the out-links and in-links of both articles to compute their relatedness. Following subsections describes this difference in detail.

6.3.3.2. Semantic Relatedness using Out-links

This measure is defined by the cosine similarity between the two out-links vectors of both articles that their relatedness is needed. The weight scheme is almost similar to tf-idf weight scheme. The only difference is that weight is computed as the link occurrence counts weighted by the probability of each link occurring instead of weighted by the probability of term occurring in the Wikipedia articles.

Calculating semantic relatedness using out-links requires the following data:

- The list of distinct links inside an article to represent out-links,

- The occurrence count of each distinct wikilink inside the article,
- count of articles that contain each link in the distinct wikilinks inside both articles, and
- The number of all Wikipedia articles, |W|.

Suppose that s and t are the source and target articles. T is the set of all articles that link to t, then the probability of the occurrence of this link is computed by Equation 6.3:

$$\omega(s \longrightarrow t) = log\left(\frac{|W|}{|T|}\right) \quad \text{if s} \in T,\ 0\ otherwise$$

Equation 6.3: The Probability of the Occurrence of a Link

The set of features of the two vectors is the union of all distinct links made from both of the two articles.

6.3.3.3. Semantic Relatedness using In-links

To compute the semantic relatedness using in-links technique, the system must maintain the following data:

- The set of in-links of each article in Wikipedia, and
- The number of all Wikipedia articles, |W|.

This technique is originally inspired by distance measure of Cilibrasi and Vitanyi [Cilibrasi R., 2007]; named Normalized Google Distance; since they used Google search engine to measure the distance between two terms. Later, Milne and Witten [Milne D., 2008a] used the same measure but based on Wikipedia structure and named "*Wikipedia Link-based Measure*" or *WLM*. The distance measure is as in Equation 6.4:

$$sr(a,b) = \frac{log\,(max(|A|,|B|)) - log(|A \cap B|)}{log(|W|) - log\,(min(|A|,|B|))}$$

Equation 6.4: Wikipedia Link-based Measure or WLM

Where *a* and *b* are the two articles of interest, A and B are the sets of all articles that link to *a* and *b* respectively.

Furthermore, semantic relatedness could be computed using both the in-links and out-links. This technique showed great performance over either using in-links or out-links alone to compute semantic relatedness [Milne D., 2008a]. The technique is computed by simply calculate the average between the two previous measures.

Computing the semantic relatedness is not the only way in ArabAgent system to disambiguate between senses. ArabAgent system uses most common sense technique as another technique for disambiguation.

Summary

Before phrase detection and phrase sense disambiguation tasks could be supported and user profile can be built, the data and statistics that used in both tasks should be elicited and prepared from Arabic Wikipedia dump. The aim of this process is to prepare the files used by the text processing component in order to perform its tasks.

The text processing component is a key component in the ArabAgent. This component uses novel techniques in treating the Arabic text. The ArabAgent depends on concepts (phrases, names, terms) extracted from the text rather than words to represent that text.

CHAPTER 7 : EVALUATION AND RESULTS

The evaluation has been taken in several levels; overall system evaluation, text processing component evaluation, and stemming technique Evaluation.

The overall system evaluation assesses the stability of the user profile; time the user profile takes to be capable to represent the user interests and knowledge.

As we discussed in chapter 6, text processing comprises of the following steps, tokenization, normalization, stemming, phrase detection, stopwords removal, and phrase sense disambiguation. To choose the best stemming techniques for our text processing component, several leading stemming techniques have been evaluated in terms of information retrieval effectiveness.

Furthermore, the text processing component has been evaluated as a unit in terms of information retrieval effectiveness as well. The evaluation of text processing component considers the evaluation:

- Experiments are conducted with and without normalization technique for phrases that have been detected.
- Experiments are conducted with and without stemming technique for phrases that have been detected
- Experiments are conducted with data extracted from various versions of Arabic Wikipedia database dumps,
- Experiments are conducted with different phrase sense disambiguation techniques

The best performance of the text processing component is compared to the stemming technique with best performance.

We begin the evaluation by first evaluating the stemming techniques. After that, the stemming technique with the best performance is used within the text

processing component that is going to be evaluated. The text processing component with the combination gives the best performance is used within the ArabAgent system that is going to be evaluated.

7.1. Evaluating the stemming techniques

The following stemming technique is going to be evaluated; Al-Stem for Kareem Darwish [Darwish K., 2002b], [Aljlayl M., 2002], Light8 for Leah Larkey [Larkey L., 2002], Berkeley Light Stemmer [Chen C., 2002], Light10 [Larkey L., 2005], SP_WOAL Light Stemmer [Al Ameed, 2005], Restrict Stemmer [Nwesri A., 2005], [Nwesri A., 2007], linguistic-based stemmer [Kadri Y., 2006], and Domain-Specific Stemmer [El-Beltagy S., 2009].

A normalization step is important for Arabic text. Normalization is used as a complement task for stemming. They are both used as techniques for helping matching desired words. While stemming is used as a technique of grouping words by considering the different morphology of the words, normalization looks for different writing habits of people. For example, normalization task is taking into consideration whether or not people are neglecting Hamza in writing Alef due to speed, drawing diacritics in seeking meaning or uttering accuracy, using Kashida for decoration, interchanging in using ى and ي, and interchanging in using ة and ه. The normalization steps have to be consistent with the stemming technique so they complement each other. For example, if a stemmer removes only letter ة from end of the word, the normalization has to replace ه with ة so the stemmer removes both.

In our experiments, if we would like to evaluate the stemming techniques we have to unify the environment of the experiments and change only the stemming technique to be used. The normalization step is one of the factors that have to be unified for all the experiments. However, for most of the stemmers in this comparison, each stemmer has its own normalization steps. The question is which normalization steps to consider. Unfortunately, a normalization task of one stemmer could not be suitable for another stemmer; due to the *conflict* which may happen between the normalization task and the

stemming mechanism. For example, some normalization replaces ة with ه other do the opposite which make this conflict. A normalization conflict happens when a normalization step affects the working mechanism of the stemming algorithm. For example, if a normalization step affects the removal of an affix by stemmer.

To unify the normalization steps we have considered the following method: First, normalization steps that are shared among all the stemmers are chosen. Then, for each remaining normalization step check whether it makes any conflict with the stemmers that don't use them. If no conflict happens with all the stemmers consider it, otherwise neglect the step.

According to the previous analysis we have unified the normalization steps for all experiments and for all stemmers. The normalization step goes as follows (hint: the order of the normalization steps has to be considered):

- Text was broken up into words at any white space or punctuation characters,
- Remove punctuation, diacritics and Kashida
- Remove non letters (consider only alphanumeric letters)
- Replace آ، إ، أ with ا
- Replace final ى with ي
- Replace final ة with ه, the step made some changes in the stemming algorithms of some stemmers; Aljlayl stemmer is modified to remove ه instead of ة and Berkeley stemmer is modified to remove يه instead of ية and stop removing ة from the end of the words. Beltagy stemmer also has some minor modifications.
- Replace the sequence ى and ء from the end of the word to ئ.

7.1.1. Experiments setup for evaluating stemming techniques

The TREC-2001 Arabic corpus, also called the AFP_ARB corpus, consists of 383,872 newspaper articles in Arabic from Agence France Presse. This fills up

almost a gigabyte in UTF-8 encoding as distributed by the Linguistic Data Consortium. There were 25 and 50 topics used in 2001 and 2002 respectively with relevance judgments, available in Arabic, French, and English, with Title, Description, and Narrative fields. We used the Arabic titles and descriptions as queries of the 75 topics in the experiments.

For all the experiments, we used the Lemur language modeling toolkit*, which was configured to use Okapi BM-25 term weighting with default parameters and with and without blind relevance feedback (the top 50 terms from the top 10 retrieved documents were used for blind relevance feedback). To observe the effect of alternate indexing terms, mean average precision was used as the measure of retrieval effectiveness. To determine if the difference between results was statistically significant, a paired t-test [Hull D., 1993] and Wilcoxon sign test [Wonnacott R., 1990] have been used with p values less than 0.05 as indication for significance.

As a requirement for Arabic text to be indexed with Lemur toolkit, corpus and topics have been first converted to CP1256 encoding. Then a normalization step was performed. The encoding conversion and normalization steps were conducted on both text collection and the topic where queries were extracted.

7.1.2. Results and discussion of Stemming techniques Evaluation

The following table, Table 7.1, shows the mean average precision of the stemmers. The stemmers are in descending order according to their mean average precisions (MAP) for expanded experiments:

Table 7.1: Mean Average Precisions for each Stemmer with and without Relevance Feedback (the best Performances are shown in bold)

Stemmer	Unexpanded	Expanded
Aljlayl-3	0.3332	**0.4003**
Light10	**0.3490**	0.3982
Light8	0.3375	0.3930
Aljlayl-1	0.3425	0.3923

Aljlayl-2	0.3411	0.3881
Beltagy_R	0.3320	0.3841
Beltagy_LR	0.3311	0.3830
Restrict	0.3119	0.3774
Al-Stem	0.3188	0.3715
Berkeley	0.3262	0.3656
Kadri	0.3078	0.3594
SP_WOAL	0.2843	0.3549
normalized	0.2478	0.3057
raw	0.2056	0.2645

The first experiment was conducted on the Arabic News corpus without performing any normalization steps or stop words removal and was called *raw*. After performing normalization steps and stop word removal we have conducted an experiment called *normalized*. The *normalized* experiment was used as a baseline experiment for all other stemming techniques' experiments. We have stemmed the corpus using each stemmer described above. The stemmers used to stem both the corpus and the topics. For Samhaa El-Beltagy stemming experiments, we have used the stemmer as a general-purpose stemmer. We haven't generated any stem list. Two experiments have been conducted; first one, Beltagy_R, used the restricted version of the stemmer by checking the existence of the possible stems in the collection and considering the one which exists only and the second experiment, Beltagy_LR, used the less restriction version of the stemmer which considers the stem whether or not it exists in the collection. We have used Aljlayl stemming algorithm [Aljlayl M., 2002] in three experiments using different affixes list. We have used light10 definite articles for all the experiments. For the first experiment, *Aljlayl-1*, we have used Al-Stem prefixes list and Light10 suffixes list. For experiment *Aljlayl-2*, we have used Al-Stem prefixes and suffixes lists. *Aljlayl-3* experiment have used SP_WOAL prefixes and suffixes lists. Although Nwesri [Nwesri; 2005, 2007] have used Microsoft Office 2003 proof toolkit in his research, we have used in our experiment for his stemmer

Restrict, the Microsoft Office 2007 proof toolkit. The rest of the stemmers have been developed as they mentioned in their studies.

Table 7.2: t-test and Wilcoxon statistical measures for non-expanded experiments

Normalized	SP_WOAL	Kadri	Restrict	Al-Stem	Berkeley	Beltagy_LR	Beltagy_R	Aljlayl-3	Light8	Aljlayl-2	Aljlayl-1	Light10	
0.000 0.000	0.003 0.002	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	**Raw**
	0.073 0.052	0.002 0.001	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.001	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	**Normalized**
		0.059 0.052	0.032 0.018	0.013 0.082	0.024 0.032	0.002 0.082	0.003 0.032	0.001 0.032	0.001 0.018	0.001 0.032	0.001 0.018	0.000 0.002	**SP_WOAL**
			0.348 0.322	0.167 0.032	0.106 0.052	0.009 0.082	0.009 0.082	0.005 0.082	0.007 0.002	0.001 0.010	0.001 0.018	0.000 0.001	**Kadri**
				0.220 0.082	0.134 0.124	0.020 0.244	0.024 0.322	0.011 0.244	0.007 0.322	0.002 0.082	0.002 0.052	0.000 0.005	**Restrict**
					0.307 0.322	0.104 0.408	0.098 0.244	0.096 0.408	0.041 0.591	0.017 0.408	0.012 0.322	0.001 0.018	**Al-Stem**
						0.351 0.917	0.327 0.947	0.231 0.677	0.161 0.408	0.043 0.322	0.031 0.408	0.017 0.082	**Berkeley**
							0.324 0.952	0.386 0.408	0.215 0.032	0.093 0.322	0.064 0.244	0.004 0.005	**Beltagy_LR**
								0.436 0.677	0.252 0.052	0.114 0.052	0.077 0.052	0.005 0.005	**Beltagy_R**
									0.281 0.177	0.040 0.023	0.035 0.082	0.003 0.018	**Aljlayl-3**
										0.268 0.591	0.194 0.244	0.014 0.040	**Light8**
											0.241 0.032	0.010 0.000	**Aljlayl-2**
												0.010 0.052	**Aljlayl-1**

The Mean Average precision measure is sometimes not enough to measure the stemmers' performance. Statistical measures are used to show the possibilities that the differences between stemmers performance occurred by chance. In our study, we used two statistical measures; paired t-test and Wilcoxon sign test. Table 7.2 and Table 7.3 show both measures to stemmers for non-expanded and expanded experiments respectively. Each cell in the two tables has two values; the upper one reflects the t-test probability and the lower is the Wilcoxon test probability. The Dark cells indicate significant differences between stemmers' results according to t-test measure.

From Experiments we have found that affixes lists to be removed from the words could affect significantly the performance of one stemmer. In our experiments Aljlayl-1, Aljlayl-2 and Aljlayl-3, in each of them we have tried different affixes list for the same algorithm and found the performance of the stemmer has improved significantly when using Al-Stem affixes lists, 03411, against SP_WOAL affixes lists, 0.3332, and the difference is statistically significant with p values of 0.040 and 0.023 for t-test and sign test respectively for the case without relevance feedback. However, in the case of relevance

feedback the performance of the stemmer when using SP_WOAL affixes lists outperforms its performance when using Al-Stem affixes lists and the difference is not statistically significant.

Table 7.3: t-test and Wilcoxon statistical measures for expanded experiments

Normalized	Kadri	SP_WOAL	Al-Stem	Berkeley	Restrict	Beltagy_LR	Beltagy_R	Aljlayl-2	Light8	Aljlayl-1	Light10	Aljlayl-3	
0.002 0.000	0.000 0.000	0.001 0.003	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	**Raw**
	0.014 0.010	0.021 0.007	0.003 0.000	0.000 0.000	0.003 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	**Normalized**
		0.605 0.852	0.166 0.208	0.361 0.244	0.138 0.363	0.022 0.147	0.022 0.100	0.008 0.023	0.003 0.003	0.003 0.001	0.000 0.003	0.006 0.023	**Kadri**
			0.122 0.208	0.300 0.322	0.109 0.147	0.035 0.065	0.037 0.208	0.080 0.453	0.012 0.013	0.011 0.065	0.022 0.147	0.010 0.007	**SP_WOAL**
				0.633 0.177	0.331 0.852	0.118 0.852	0.087 0.719	0.072 0.322	0.020 0.280	0.030 0.280	0.005 0.363	0.047 0.280	**Al-Stem**
					0.291 0.591	0.136 0.500	0.123 0.719	0.252 0.792	0.039 0.124	0.035 0.082	0.019 0.052	0.036 0.124	**Berkeley**
						0.356 0.453	0.330 0.280	0.252 0.792	0.163 0.065	0.187 0.147	0.099 0.065	0.047 0.208	**Restrict**
							0.384 0.636	0.270 0.546	0.113 0.040	0.156 0.100	0.026 0.013	0.125 0.280	**Beltagy_LR**
								0.325 0.363	0.129 0.065	0.188 0.065	0.035 0.000	0.151 0.363	**Beltagy_R**
									0.224 0.363	0.210 0.407	0.047 0.174	0.149 0.546	**Aljlayl-2**
										0.554 0.453	0.081 0.453	0.312 0.852	**Light8**
											0.088 0.079	0.279 0.792	**Aljlayl-1**
												0.441 0.792	**Light10**

7.2. Text Processing Component Evaluation

The text processing component is going to be evaluated in terms of information retrieval effectiveness as the stemming techniques above. The experiments have the same setup as in subsection (7.1.1). The evaluation uses the same test collection as in the evaluation of the stemming techniques above. The text is normalized at first using the unified normalization technique mentioned in (7.1). Then, concepts are identified for each document before indexed using the lemur toolkit.

Six experiments have been conducted. Experiments one, two, three, and four have used version3 of Wikipedia database dumb (the latest version at that time). Experiment one used system that disambiguate between concepts using "*most common sense*" disambiguation

technique; by choosing the most common concept of the candidate ones. Experiment two uses the disambiguation technique that depends on in-link concepts. Experiment three uses the disambiguation technique that depends on out-link concepts. Experiment four uses both in-links and out-links concepts for disambiguation. Experiment five and six uses version1 and version2 respectively with the disambiguation technique that give highest performance through experiments one, two, three and four.

One parameter has been used to adjust the disambiguation techniques for the system speed purpose. The parameter is used to reduce the number of candidate concepts to be disambiguated for an n-gram which reduces the overall computation time. The parameter calculates the percentage of appearance of a concept as out-link relevant to the sum of appearance of all the candidate concepts and neglecting concept under certain threshold. This threshold is set to 0.02 as in [Milne D., 2008b].

7.2.1. Results and discussion of Text Processing Component Evaluation

The following table, Table 7.4, shows the mean average precision of the different experiments, as appeared in [Eldesouki M., 2011]. The table contains three extra experiments conducted in [Eldesouki M., 2009] for comparison's sake[39]. In addition to experiment *raw* which was conducted on the Arabic News corpus without performing any normalization steps or stop words removal there are two other experiments, one conducted after normalization and stopwords removal process, called *normalized*, and the other after stemming the corpus using the light10 stemmer, called *Light10*. The stemmers are in

[39] We justify comparing our technique of text processing with stemming techniques by two reasons, the first one is that both techniques are used here as text processing techniques to boost the effectiveness of Information Retrieval systems. The other reason is that stemming techniques so far outperforms other techniques.

descending order according to their mean average precisions (MAP) for experiments *expanded* by blind relevance feedback:

Table 7.4: Mean Average Precisions for the different Experiments

Experiment	Unexpanded	Expanded
raw	0.2056	0.2645
normalized	0.2478	0.3057
Exp5 (version1 + out-link disambiguation)	0.3111	0.3312
Exp6 (version2 + out-link disambiguation)	0.3120	0.3550
Exp1 (version3 + common concepts)	0.3252	0.3561
Exp4 (version3 + both in-link and out-link)	0.3225	0.3721
Exp2 (version3 + in-link disambiguation)	0.3290	0.3759
Exp3 (version3 + out-link disambiguation)	0.3301	0.3801
Light10	**0.3490**	**0.3982**

Experiments Exp3, Exp5 and Exp6 use the same setting and parameters for their experiments except that each one uses different version of Wikipedia database dump. Exp6 uses version of Wikipedia that is most recent than for Exp5. Also Exp3 uses a version that is most recent than for Exp6.

As you can notice from experiments Exp5, Exp6 and Exp3, the mean average precisions gradually increase. This indicates that as Wikipedia continually grows and develops the performance improves. Thus, even if the current retrieval system that uses the light stemming outperforms our system; there is likelihood that using Wikipedia could outperform retrieval using light stemming technique over time.

7.3. Overall System Evaluation

The overall system evaluation assesses the system in terms of the stability of the user profile; the time profile takes to have the capability to represent the user.

7.3.1. Experiments Setup for Overall System Evaluation

The ArabAgent system uses a disambiguation technique that leverage similarity relatedness that takes advantage of out-links of the Wikipedia articles since it has the best performance in the experiments of section (7.2).

For overall system evaluation, news articles of 40 days from *Ahram Gate* news website –"بوابة الاهرام"- have been downloaded. The average number of articles per day is about 328 with total number of news articles is 13124. Table 7.5 illustrates the dates of the (40) days and the numbers of articles per day. The table also shows the numbers of relevant articles of each day with respect to the author needs.

Table 7.5: Dates and numbers of Ahram Gate News Articles Used in Overall System Evaluation

Day No.	Date	Number of Articles in the day	Number of Relevant Articles
1	10-05-2011	345	19
2	11-05-2011	270	30
3	12-05-2011	311	30
4	13-05-2011	353	35
5	14-05-2011	298	20
6	15-05-2011	360	32
7	16-05-2011	319	27
8	17-05-2011	353	32
9	18-05-2011	367	41
10	19-05-2011	325	35
11	20-05-2011	268	24
12	21-05-2011	293	26
13	22-05-2011	311	26
14	23-05-2011	342	20
15	24-05-2011	348	30
16	25-05-2011	336	30
17	26-05-2011	351	30
18	27-05-2011	316	21
19	28-05-2011	328	17
20	29-05-2011	279	25
21	30-05-2011	338	21
22	31-05-2011	294	20
23	01-06-2011	324	20
24	02-06-2011	356	21
25	03-06-2011	328	20
26	04-06-2011	347	20
27	05-06-2011	341	20
28	06-06-2011	359	20
29	07-06-2011	329	20
30	08-06-2011	324	22
31	09-06-2011	288	80

32	10-06-2011	318	77
33	11-06-2011	374	83
34	12-06-2011	337	63
35	13-06-2011	371	76
36	14-06-2011	234	79
37	15-06-2011	363	91
38	16-06-2011	351	72
39	17-06-2011	357	59
40	18-06-2011	318	51

The last 10 days of the 40 days are used to test the system; starting from 09-06-2011 and ending with 18-06-2011. The 30 days that precede each day of the last 10 days are used to train the system (the 30 days are not the same; the movement of the 30 days let me test a new day that may have new topics not in the previous day). For each day of 10 testing days, 30 user profiles (or UPs for short) are built and used to filter the news articles in the day of testing. The user profile 1 (or UP 1) of the test day 09-06-2011, has been built using the relevant articles of the day 08-06-2011. The user profile 2 (or UP 2) of the test day 09-06-2011, has been built using the relevant articles of the days 08-06 and 07-06-2011. The user profile 3 (or UP 3) of the test day 09-06-2011, has been built using the relevant articles of the days 08-06, 07-06 and 06-06-2011. This process continues until we finish the 30 profiles. Figure 7.1 depicts building the 30 user profiles of the first day of the 10 days of testing. Each day of the days that train the system has an average of 25 training article.

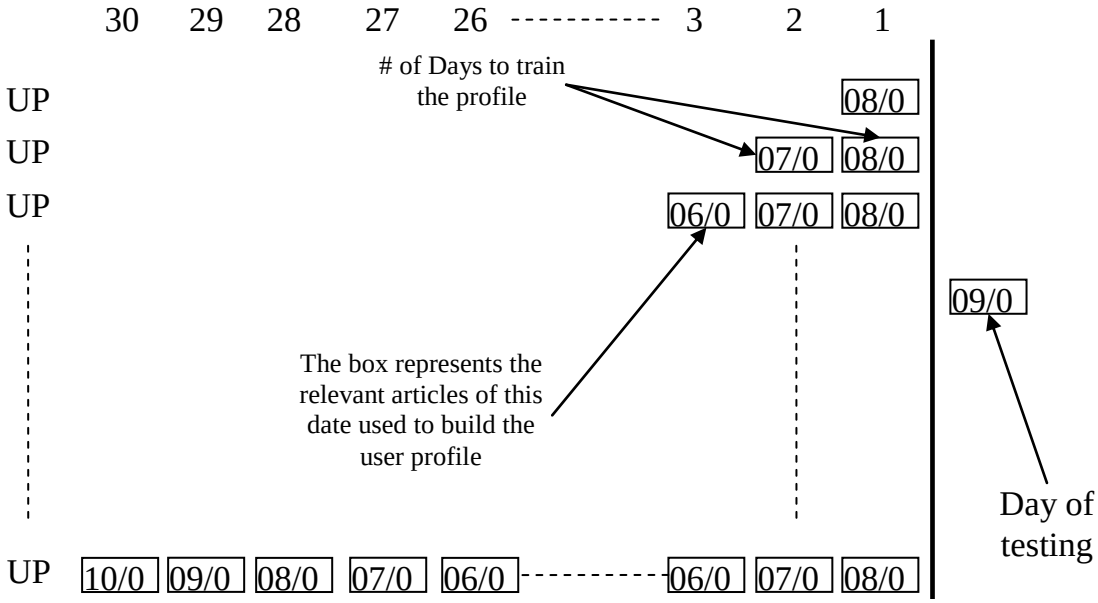

Figure 7.1: Building the 30 User Profiles of the First Day of Testing (Day 09/06/2011)

Each user profile of the 30 is used to filter the news article in the test day. The performance of each user profile is measured using the accuracy measure. This experiment has been done for the remaining 9 days that have been dedicated for testing.

Figure 7.2 shows the results of accuracy of the 30 user profiles of the first testing day (09/06/2011). Notice that the performance increases as user profiles maintain more days. Notice that, the increase in accuracy is faster in the first few user profiles and the performance stabilizes as it reaches user profile 30.

Figure 7.3, Figure 7.4, Figure 7.5, Figure 7.6, Figure 7.7, Figure 7.8, Figure 7.9, Figure 7.10, and Figure 7.11 depict the accuracy of the 30 user profiles of the remaining 9 day of testing. Notice that, the accuracy curve starts stabilize after a while.

Figure 7.12 depicts all the curves of all test days for comparison. Notice that, the accuracy differs from one test day to another this could be attributed to for (1) appearance of articles with new topics, that interest the user, among articles of the test day that (2) the difference in representation between the different topics of interest i.e. one topic could have enough training articles to represent the topic while others don't have.

Figure 7.13 depicts the average curve of the ten days of testing. The average accuracy curve of the user profile takes about 15 days to converge. We consider only an average of 25 relevant news articles (chosen randomly) of an average of 80 relevant articles. The convergence, definitely, takes less to converge if more than 25 relevant news articles are chosen and more time if less relevant articles are chosen.

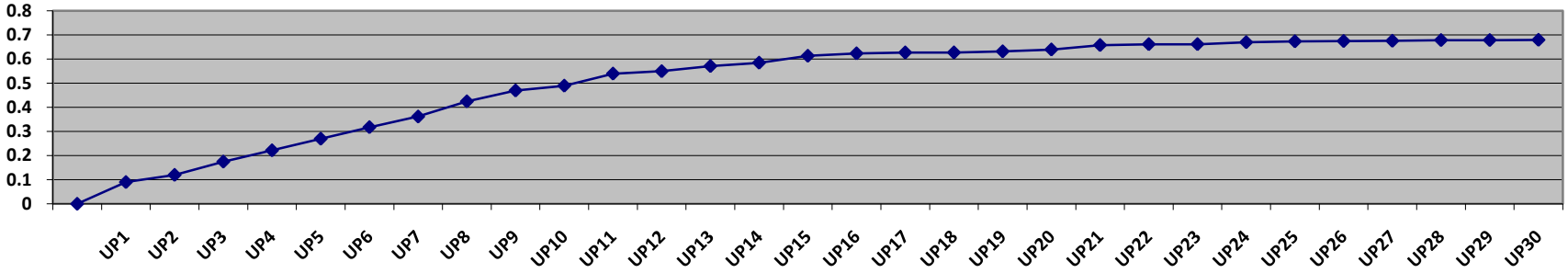

Figure 7.2: Accuracy of the 30 user profiles of the first day of testing (09-06-2011)

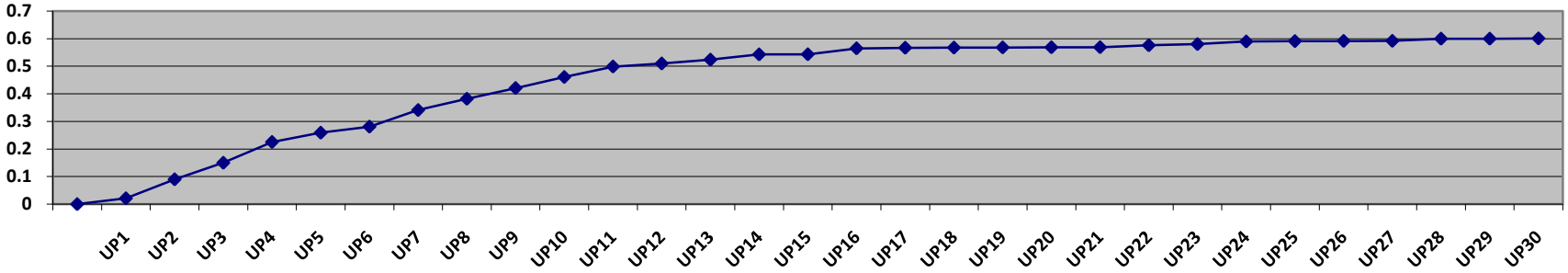

Figure 7.3: Accuracy of the 30 user profiles of the second day of testing (10-06-2011)

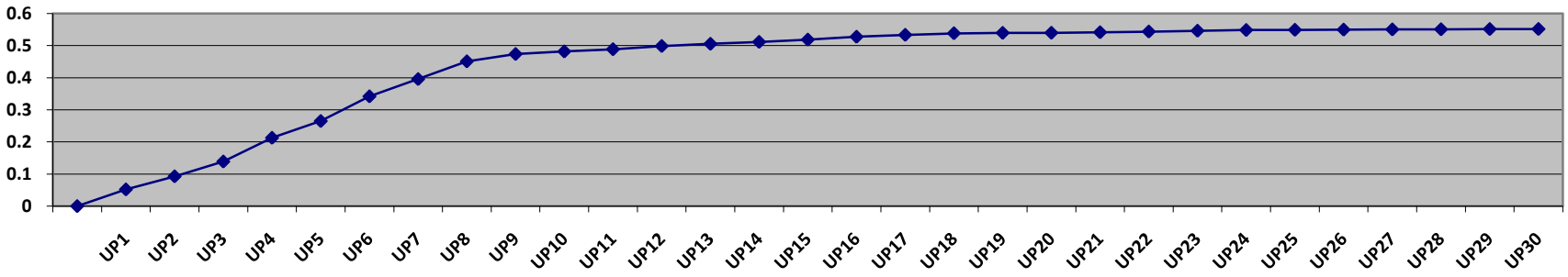

Figure 7.4: Accuracy of the 30 user profiles of the third day of testing (11-06-2011)

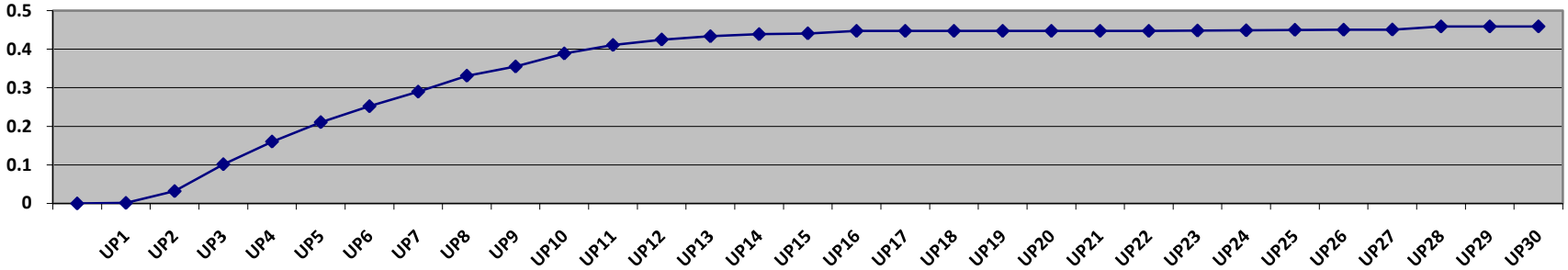

Figure 7.5: Accuracy of the 30 user profiles of the fourth day of testing (12-06-2011)

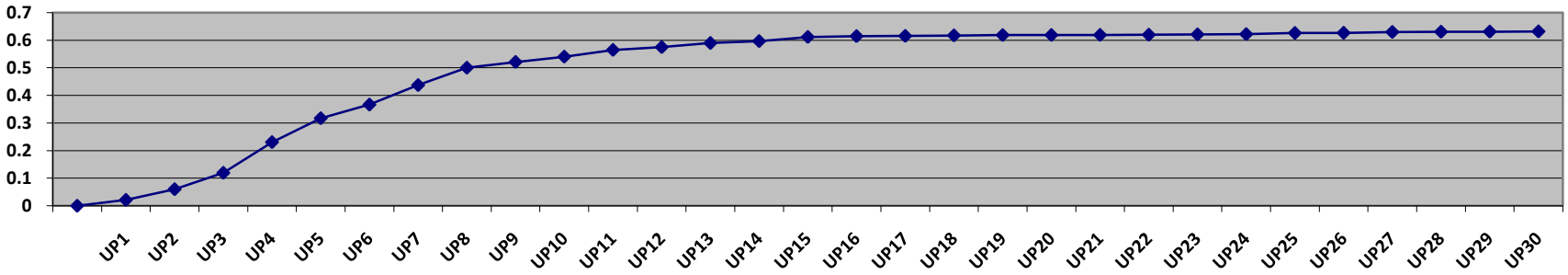

Figure 7.6: Accuracy of the 30 user profiles of the fifth day of testing (13-06-2011)

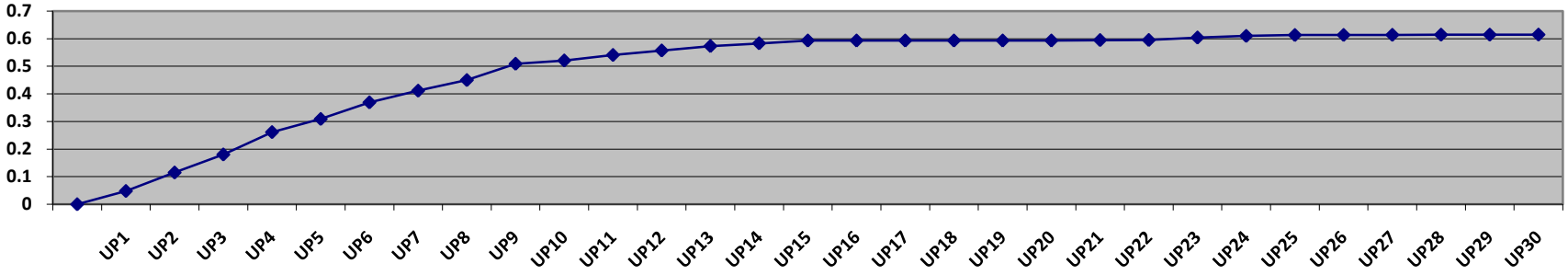

Figure 7.7: Accuracy of the 30 user profiles of the sixth day of testing (14-06-2011)

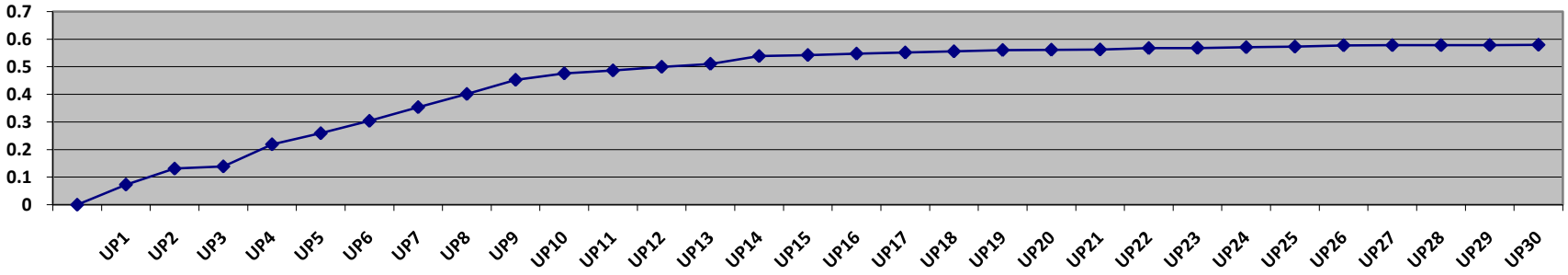

Figure 7.8: Accuracy of the 30 user profiles of the seventh day of testing (15-06-2011)

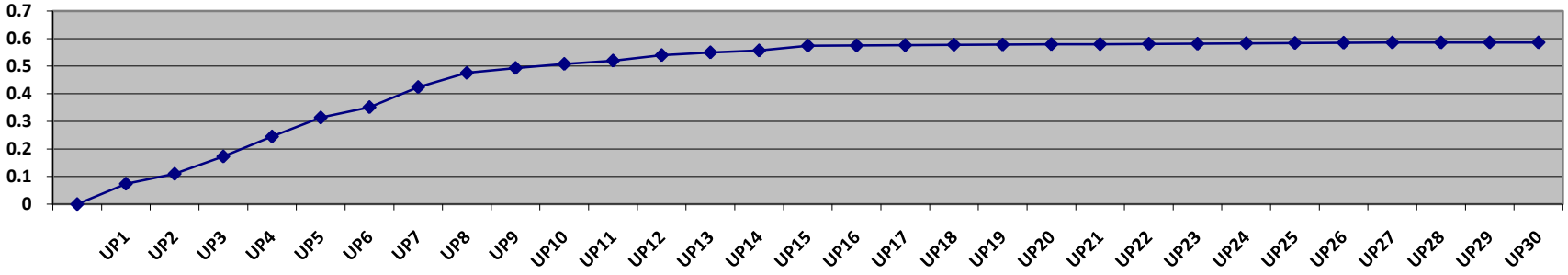

Figure 7.9: Accuracy of the 30 user profiles of the eighth day of testing (16-06-2011)

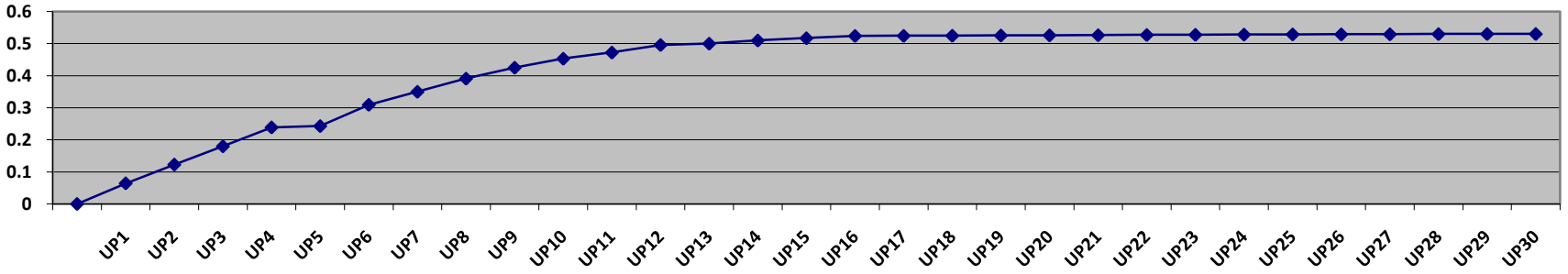

Figure 7.10: Accuracy of the 30 user profiles of the ninth day of testing (17-06-2011)

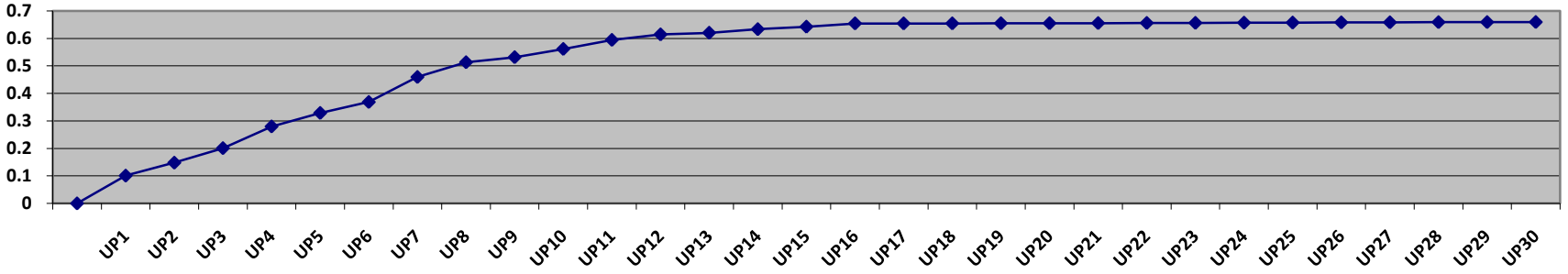

Figure 7.11: Accuracy of the 30 user profiles of the tenth day of testing (18-06-2011)

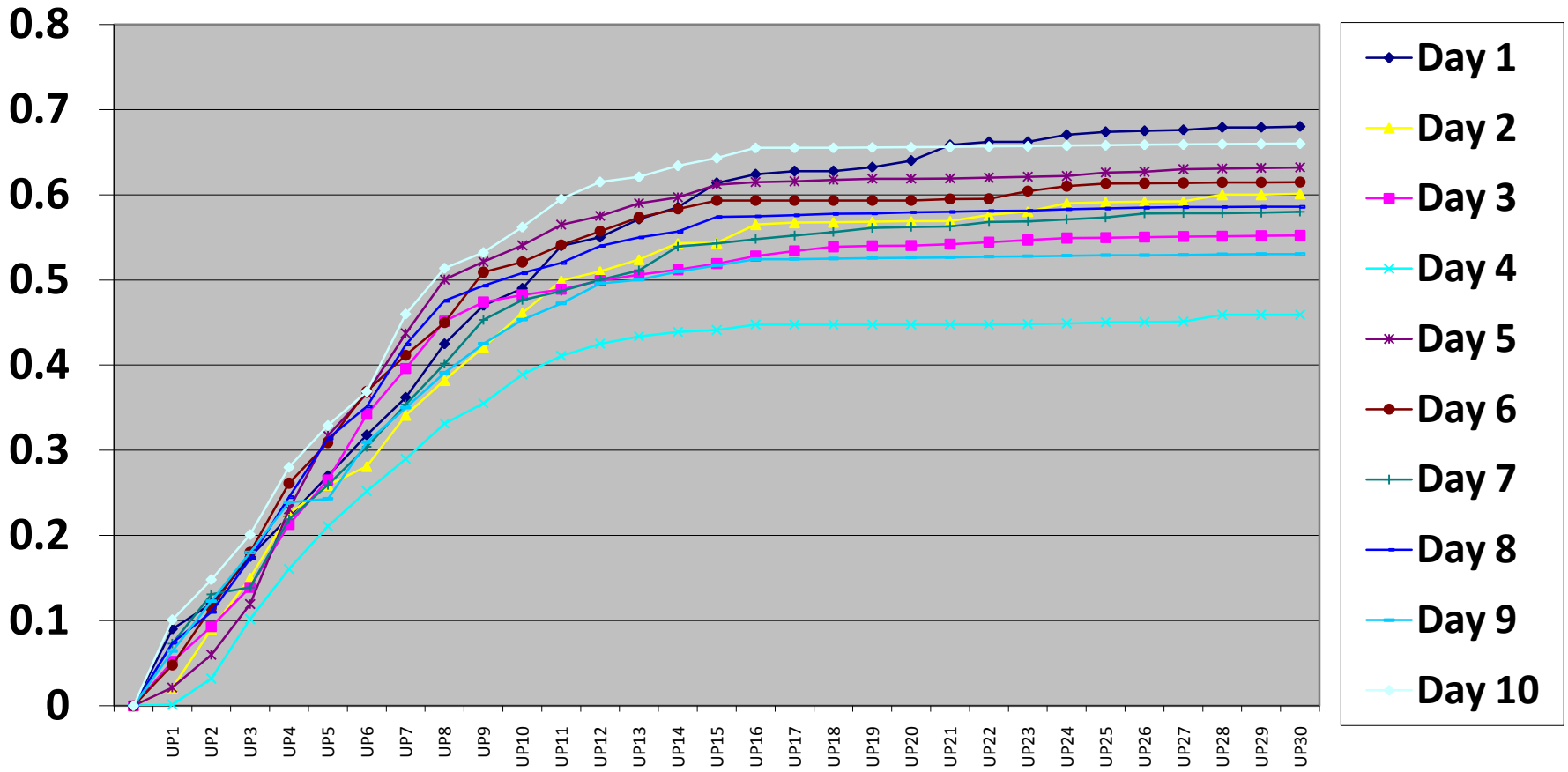

Figure 7.12: Accuracy of the 30 user profiles of all test days

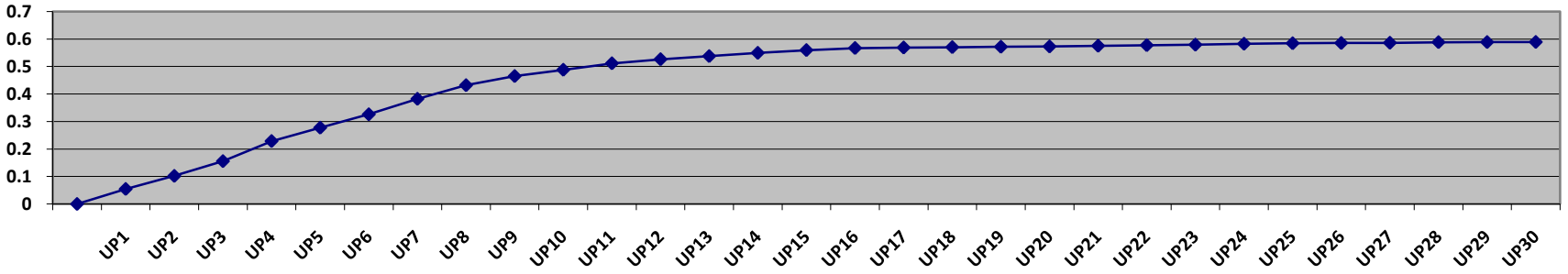

Figure 7.13: Average of the accuracy of the 30 user profiles of the entire 10 days of testing

Summary

The evaluation has been taken in several levels; overall system evaluation, text processing component evaluation, and stemming technique Evaluation.

The results show that the light10 stemmer outperforms the other stemmers in non-expanded experiments and Aljlayl-3 outperforms them in case of expansion.

Aljlayl-1, Aljlayl-2 and Aljlayl-3 experiments shows that different affixes lists could affect significantly the performance of one stemmer.

Aljlayl-2 and Al-Stem experiments shows that using different stemming algorithms for removing affixes even with the same affixes list produces different results.

The text processing component is going to be evaluated in terms of information retrieval effectiveness as the stemming techniques.

The use for concepts in retrieval led to significantly higher performance than ordinary normalization process. A good disambiguation technique is needed for concept disambiguation.

Although the stemming technique is still better, the continuous growth of Wikipedia improves the performance of concept-based information retrieval. Results show that the information retrieval performance is improving as Wikipedia develops and grows. Also, Wikipedia is a good source for concepts since the new concepts that appear on the scene are frequently added.

The curve of the average accuracy of the 30 user profiles of the 10 test days takes about 15 days to converge. We consider only an average of 25 relevant news articles (chosen randomly) of an average of 80 relevant articles. The convergence, definitely, takes less to converge if more than 25 relevant news articles are chosen and more time if less relevant articles are chosen.

CONCLUSION AND FUTURE WORK

The amount of information available on the World Wide Web is massive and continually increases. The simplicity of web attracted massive number of users. Different users with different backgrounds, motives, goals and languages have attracted to either produce information or consume it. Accordingly, they differ in their needs.

The amount of available information introduced an unprecedented state of information that overwhelmed the user. This state some time referred to as *Information overload.* As a solution of information overload search engines were invented.

There are more than 65.4 million users of the Internet in Arabic World. This is roughly 18.8% of the Arabic users is accessing the internet. The percentage of the Arabic users is about 3.3% of the internet users in the whole World. The Internet World Stats Web site, places the Arabic users among the top ten of other languages' users. The good news is that Arabic users have the biggest growth rate in internet in the last decade by about 2501.2% growth rate.

After examining and comparing some of the previous definitions for agency notion, researchers showed that there is no commonly agreement upon the definition of exactly what an agent is. Although there is no agreement between researchers on what agent is, agent-based systems possess some characteristics that distinguish them from other systems. The most characteristics are *Autonomy, learning, Discourse, Risk and trust, Domain, Graceful degradation, Reactive, Proactive, Communicative.*

Nwana assume that agents could have minimum two of any of the attributes *autonomy, learning* and *cooperation* and the agent that posses learning attribute called *intelligent agent*.

Our approach is to use *Intelligent Information Agent* as a general approach and framework to overcome the problem of overloading.

ArabAgent is used to assist its user and ("or acts on behalf of its user") to tailor web search results, modify user query, and filter news articles. ArabAgent accepts user's relevance feedback to adapt itself to the new interests of the user.

The ArabAgent system consists of seven components; user profile, user interface, query customizer, web wrapper, ranking and filtering component, user modeler and profiler, and text processing component.

Although the great advantages of multi-agent systems, our system, ArabAgent, is a stand-alone agent that act on behalf of his user to pursue his/her goal while accessing information on the Web. ArabAgent is an Autonomous intelligent agent capable of learning and adapting to represent his user. It represents his user interests and knowledge while surfing and searching the World Wide Web.

The ArabAgent system has been divided into part; the first part is in the client side and the other side exists in a server side. This architecture provides dynamic in design, since it is suitable for single content-based Agent as well as Multi-Agent Collaborative recommendation system.

The ArabAgent system implements the Internet-based Architecture; sometimes called multitier architecture. Each rectangle represents a tier. The first tier, that contains the presentation layer and presentation logic layer, is implemented at the client-side. Intuitively, this tier represents the user interface component.

The text processing step is fully or partially required for the following: user profile updating Task, filtering and customization Task, and query modifying task.

The main goal of the text processing component is identifying concepts from documents. Identifying concepts from documents considers two steps. The first step is to detect phrases that seems to be concepts and assign for each of them the corresponding prospective concept(s) id(s) (in case of polysemy, a term can have several concept

ids), and second step is to disambiguate between concepts for phrases that have multiple concepts.

The Arabic Wikipedia project has been chosen to be a source to provide data and statistics to build the user profile. Furthermore, Arabic Wikipedia project has facilitated building a word sense disambiguation technique based on Wikipedia link structure to detect and disambiguate between concepts to use in the text processing component.

ArabAgent system applies explicit techniques to focus mainly on the task of user modeling and expressing user context without being concerned by the efficiency and accuracy of the implicit techniques used to judge on the relevancy of an item or object the user inspect. ArabAgent provides two ways for explicit feedback; while searching and while browsing the Web.

The user profile is updated frequently to adapt to the frequently changing user interests and needs. A time window technique of size 30 days is used to adapt the user interest attributes automatically to mimic user interest behavior. The interest attribute value is get decay over time until it reaches value zero i.e. the user has no interest in a topic anymore; although he may has a great knowledge about the topic.

ArabAgent attempts to represent the user's interests, knowledge and attitudes to define the importance of a web page to the user as a way to individualize the access to the web. To make the user model reflect the above vision, two structures have been combined together to comprise the user model; a hierarchy of categories and semantic networks. The updating formula considers the following:

- Document weight in the calculation.
- Documents count in one day in the calculation.
- Differentiate between the days by considering day recency.
- The number of days

The ArabAgent Agent provides three forms of personalization; news filtering, tailoring web search results, and modifies search queries. ArabAgent combines two forms of personalization search results; results tagging and result re-ranking.

A graph comparison algorithm, adopted from [Sorensen H., 1997], is used to compare the semantic network of a web page against the semantic network of the user profile.

ArabAgent depends on the “meta” tag to identify the encoding of the web page. Then, the concepts are identified using the text processing component.
The evaluation has been taken in several levels; overall system evaluation, text processing component evaluation, and stemming technique Evaluation.
The results show that the light10 stemmer outperforms the other stemmers in non-expanded experiments and Aljlayl-3 outperforms them in case of expansion.
Aljlayl-1, Aljlayl-2 and Aljlayl-3 experiments shows that different affixes lists could affect significantly the performance of one stemmer.
Aljlayl-2 and Al-Stem experiments shows that using different stemming algorithm for removing affixes even with the same affixes list produce different results.
The text processing component is going to be evaluated in terms of information retrieval effectiveness as the stemming techniques.
The use for concepts in retrieval led to significantly higher performance than ordinary normalization process. A good disambiguation technique is needed for concept disambiguation.
Although the stemming technique is still better, the continuous growth of Wikipedia improves the performance of concept-based information retrieval. Results show that the information retrieval performance is improving as Wikipedia develops and grows. Also, Wikipedia is a good source for concepts since the new concepts that appear on the scene are frequently added.

The curve of the average accuracy of the 30 user profiles of the 10 test days takes about 15 days to convergence. We consider only an average of 25 relevant news articles (chosen randomly) of an average of 80 relevant articles. The convergence, definitely, takes less to converge if more than 25 relevant news articles are chosen and more time if less relevant articles are chosen.

Future Work

- The ArabAgent could serve as multi-agent collaborative recommendation system with a minor modification to the system to the framework.
- Evaluation of query modification techniques
- Investigate techniques to use the hierarchy of categories to solve the serendipity problem
- Leveraging the hierarchy of categories to apply the serendipity
- Removing the burden on user intervention by incorporating implicit user relevance feedback techniques.

APPENDICES

Appendix A: The Stemming Techniques Comparison

Due to the excel of the light stemmers when compared with other techniques for stemming for the purpose of information retrieval, the scope of this study is the light stemming techniques. We compare between the following stemmers: Al-Stem for Kareem Darwish [Darwish K., 2002b], [Aljlayl M., 2002], Light8 for Leah Larkey [Larkey L., 2002], Berkeley Light Stemmer (Chen A., 2002], Light10 [Larkey L., 2007], SP_WOAL Light Stemmer [Al Ameed, 2005], Restrict Stemmer [Nwesri A., 2005], [Nwesri A., 2007], linguistic-based stemmer [Kadri Y., 2006], and Domain-Specific Stemmer [El-Beltagy S., 2009]. The stemmers are compared against the following criteria:

- The main idea behind the stemmer built,
- The prefixes and suffixes they remove, and
- The basis of choosing the affixes
- The algorithm they use to remove the affixes.
- IR performance; precision and recall (this is in chapter 7)
- Limitation of the stemmer

A.1. Al-Stem Stemmer

A stemmer developed by Kareem Darwish [Darwish K., 2002b] and modified by Leah Larkey from University of Massachusetts and further modified later by David Graff from LDC. It is intended for research purposes only. The original stemmer of Kareem Darwish removes the following prefixes: (وال، فال، بال، بت، يت، لت، مت، وت، ست، نت، بم، لم، وم، كم، فم، ال، لل، وي، لي، في، وا، فا، لا، با) and removes the following suffixes: (ات، وا، ون، وه، ان، تي، ته، تم، كم، هم، هن، ها، ية، تك، نا، ين، يه، ة، ه،ي). In David Graff version of the stemmer the stemmer two additional prefixes are removed (تت، سي) and two additional suffixes are

removed; (تا، ا). Kareem Darwish claims that the Al-Stem is more Aggressive than Light10 stemmer. Graff version of the stemmer has two modes for normalization light and aggressive.

The stemmer works as follows:

- Remove the following prefixes if exist from beginning of the word in the next order from right to left: (وال، فال، بال، بت، يت، لت، مت، تت، وت، ست، نت، بم، لم، وم، كم، فم، ال، لل، وي، لي، سي، في، وا، فا، لا، با)
- Remove the following suffixes if exist from the end of the word in the next order from right to left: (ات، وا، تا، ون، وه، ان، تي، ته، تم، كم، هن، هم، ها، ية، تك، نا، ين، يه، ة، ه،ي، ا)

A.2. Aljlayl Stemmer

Mohammed Aljlayl [Aljlayl M., 2002] developed a light stemmer used for his own information retrieval researches in TREC cross-language track. Aljlayl didn't mention the prefix or suffix list going to be removed from word rather he mentioned only that "… to remove the most frequent suffixes and prefixes..," then he said "The most common suffixation includes duals and plurals for masculine and feminine, possessive forms, and pronoun forms," and "The definite articles and prefixes that can be attached to the head of the definite article are considered the most common prefixes. In addition, the letter (و) is a commonly used letter to start the sentences within the Arabic language,"

The algorithm of stripping the affixes is as follow:

- If word length is greater than or equal 3 characters, then remove the prefix و
- Remove the article from the beginning of the word if exist then normalize أ، إ، آ from the beginning of the word to ا
- If the length of the remaining stem is greater than or equal 3 characters, then remove the suffixes form the stem using longest first strategy (remove the longest suffix first) only if remaining part of the stem is greater than or equal 3 characters.

- **While** length of the remaining stem is greater than 3 characters **do**,
    - Remove the prefixes form the stem only if remaining part of the stem is greater than 3 characters.
- Return the stem

A.3. Light8 Stemmer

It is a light stemmer developed by Leah Larkey [Larkey L., 2002] for the purpose of researching. The construction of the stemmer based on heuristics; try to remove strings which would be found as affixes far more often than they would be found as the beginning or end of an Arabic word without affixes. The stemmer removes the following prefixes: (ال، وال، بال، كال، فال، و) and the following suffixes: (ها، ان، ات، ون، ين، يه، ية، ه، ة، ي)

The stemmer works as follows:

- Remove و if the remainder of the word is 3 or more characters long.
- Remove any of the definite articles if this leaves 2 or more characters.
- Remove any of the following suffixes in order form right to left (ها، ان، ات، ون، ين، يه، ية، ه، ة، ي) if this leaves 2 or more characters.

A.4. Light10 Stemmer

Light8, which has been developed by Leah Larkey, becomes Light10 [Larkey L., 2007] after some modifications. Light10 was designed to strip off strings that were frequently found as prefixes or suffixes, but infrequently found at the beginning or ending of stems *without intended to be exhaustive*, as light8 did before. Light10 tries to improve the Information Retrieval (IR) performance. Larkey used heuristic as a strategy for developing here stemmer. And it did it. It outperforms most of the morphological analyzers in that time; Amira 1.0, Khoja, Buckwalter morphological analyzer, etc. This is important because some researcher claims that light10 is not good because it doesn't return the right form of the word.

The stemmer removes the following prefixes: (ال، وال، بال، كال، فال، لل، و) and it removes the following suffixes: (ها، ان، ات، ون، ين، يه، ية، ه، ة، ي). The Light10 stemmer removes the same set of suffixes as Light8. However, Light10 add

(ال) to the prefix list to be removed. This addition made Light10 outperform Light8.

Something that is notable in Larkey stemmers, Light8 and Light10, that they only remove definite articles. The stemmers don't remove any Arabic prefixes from words.

Light10 stemmer works as follows:

- Remove و if the remainder of the word is 3 or more characters long.
- Remove any of the definite articles if this leaves 2 or more characters.
- Remove any of the following suffixes in order form right to left (ها، ان، ات، ون، ين، يه، ية، ه، ة، ي) if this leaves 2 or more characters.

A.5. SP_WOAL Stemmer

Al Ameed [Al-Ameed K., 2005] has reviewed multiple stemmers used in TREC 2001 and 2002 cross-language track. He reviewed the Al-Stem, Light8 stemmer and another stemmer he called it TREC-2001 stemmer; which is a modified version of Larkey's Light8 stemmer. Then he decided to enhance the performance of these stemmers in two ways. First enhancement is done by adding new affixes to the existing affixes of the mentioned stemmers. The second way is by modifying the sequence of algorithm components execution.

Although the author was intending to develop a stemmer to improve the performance of information retrieval tasks, he didn't conduct any IR evaluation. He also said that his stemmer is much better than the stemmers that have developed for the TREC cross-language track claiming that his stemmer produces much more correct words than other stemmers and he neglected that it doesn't depend only on the correctness of the words to make an efficient retrieval process.

The enhancement produced in SP_WOAL light stemmer. Although the user mentioned his prefixes list contain 17 two-characters, the list contains only 15. However, it contains 5 single-characters prefixes rather than 3. The stemmer removes the following prefixes (ال، وال، بال، فال، كال، لل، وللل، ب، ل، فا، ست، با، ي، سي، لت، ت، لي، في، وبال، ن، كا، وست، لن، فت، وسن، فن، وسا، ولا، ولي، سا، سن، ولت،

ين، ون، ات، ان، ي، ه، ها، هم، ة،) and removes the following suffixes (ولن، وسي، يه،كم، نا، وا، تم، ا، ت، هن، ك، ته، تك، تن، و، ن، كن، تا، ما، يا، ني،). The stemmer is considered very aggressive stemmer. The stemmer works as follows:

- Remove the prefix ال from the beginning of the word
- Recursively remove the suffix from the end of the word starting with longest suffixes first.
- Non-recursively removes the prefix form the beginning of the word starting with longest prefixes first.

A.6. Berkeley light stemmer

In 2002, University of California at Berkeley participated only in the cross-language track in TREC conference. They developed a light stemmer that made them among the best performers in the track [Chen A., 2002]. They used the standard Arabic data collection provided by Linguistic Data Consortium LDC to develop their light stemmer; they chose the affixes with the most frequently occurrence and that give highest performance when practically evaluated using the test collection. They also depended on the on the grammatical functions of the affixes and their English translations to choose their affixes.

The stemmer removes 26 prefixes and 22 suffixes during the stemming processing. The list of the prefixes that should be removed is: (ال، وال، بال، فال، كال، و، لل، وللل، ب، ل، فا، با، سي، وم، وت، وي، وا، لا، وب، ول، وس، كا، مال، اال، سال، لال) and the list of the suffixes that should be removed is: (ين، ون، ات، ان، ي، ه، ها، هم، ة، ية، كم، نا، وا، تم، ت، هن، تن، كن، ما، يا، ني). The Berkeley light stemmer works as follows:

1. If the word is at least five-character long, remove the first three characters if they are one of the following: (مال، اال، سال، لال، وال، بال، فال، كال، وللل).
2. If the word is at least four-character long, remove the first two characters if they are one of the following: (لل، فا، با، سي، وم، وت، ال، وي، وا، لا، وب، ول، وس، كا،)
3. If the word is at least four-character long and begins with و, remove it.

4. If the word is at least four-character long and begins with either ب or ل, remove ب or ل, only if, after removing the initial character, the resultant word is present in the Arabic document collection.
5. Recursively strips the following two-character suffixes in the order of presentation if the word is at least four-character long before removing a suffix:
   (ون، ات، ان، ين، تن، تم، كن، كم، هن، يا، ني، يا، وا، ما، نا، هم، ية، ها)
6. Recursively strips the following one-character suffixes in the order of presentation if the character is at least three-character long before removing a suffix: (ت، ي، ه، ة).

A.7. Kadri's linguistic-based stemmer

The developing of the Kadri's linguistic-based stemmer [Kadri Y., 2006] depended on idea that the Arabic word consists of five part their order is; antefixes, prefixes, stem, suffixes and postfixes. The first part which is the antefixes is the prepositions and conjunctions. However, the prefixes are the conjugations person of verbs. The suffixes are Termination of conjugation and numbers marks of the nouns. The postfixes are the pronouns that catch up with end of the word.

The list of Antefixes is: وبال، وال، بال، فال، كال، ولل، ال، وب، ول، لل، فس، فب، فل، وس، ك، ف، و، ب، ل and the list of prefixes is: ا، ن، ي، ت. The list of suffixes is: تما، يون، تين، تان، ات، ان، ون، ين، وا، تا، تم، تن، نا، ت، ن، ا، ي، و. And the list of postfixes is: كما، هما، كن، هن، تي، ها، نا، هم، كم، ك، ه، ي.

The linguistic-based stemmer has two phases to work:

1. Training Phase:
   - A list of stems with its frequency occurrence is build for each word using corpus to avoid ambiguity that my happen when removing affixes.
2. The Stemming Phase:
   - The stemmer truncates possible affixes according to the above table.

- If there an ambiguity raised for the stemmer (more than one combination was available), then stemmer selects the most appropriate candidate; according to corpus statistics computed in the training phase.

A.8. Restrict Stemmer

[Nwesri A., 2005] focused on removing conjunctions, و, and ف, and prepositions, ل, و, ك, and ب, that come as prefixes in the beginning of the words. He (only in 2005) didn't mention other affixes such as articles and suffixes in general. He just tried to find a way to recognize the two types of affixes; the conjunctions and the prepositions.

In [Nwesri A., 2007] he developed an Arabic stemmer called *Restrict*. Its main idea is to retaining valid Arabic core words. This because he claimed that removing wrong affixes sometimes results in incorrect stem and in most cases reduces retrieval precision by conflating different words to the same class.

Nwesri used in his proposed technique two things to improve his performance. The first was the Microsoft Office 2003 Arabic spellchecker to ensure that he extracted only correct words. The second was simple rules or heuristics exist in Arabic language to guarantee the correctness of the affixes removal. Although these rules don't guarantee correctness of hundred percent, they improve the information retrieval performance. The rules are removing a prefix keep the remaining word correct, adding the *waw* or *faa* conjunction keeps the modified word correct, altering the a prefix with *waw* and *faa* keeps the modified word correct, and duplicating a particle result in a wrong word except for the *lam* case.

He has a good justification for depending on correctness of words for improving the IR performance as follows, "Although correct words are not the main target of stemming, an incorrect stem can have a completely different meaning and correspond to a wrong index cluster."

The algorithm work as follows:

i. Dealing with لل, prefix:

a. If the word is correct after removing the prefix لل, then remove it.
b. Otherwise, we add the letter ا, before the word, if the new word is correct we drop one lam from the original word.

ii. Dealing with ل, particle when precedes definite article ال:
a. We replace the first lam with the letter ا, if the word exists in the lexicon remove the prefix without check lexicon (he depends on that the words that start with lam cannot preceded by Alef)
b. We remove the first letter and check to see whether we can drop the first lam.

iii. (Here could be one of the three algorithms suggested by Nwesri)

iv. If a word starting with either waw or faa and after stemming has three or more characters that has either waw, kaf, baa, or lam as its first character

He suggested three algorithms to handle the conjunctions and the prepositions. All the algorithms depend on checking the words in the lexicon after removing the first letter as follows:

- Remove and Check in Lexicon (RCL)
  - The prefix of a word is removed if the remaining word exists in the Arabic lexicon.
- Replace and Remove (RR)
  - Remove the prefix and check the remaining word in lexicon
  - If exist, produce to instances of the remaining word by appending waw and faa to the beginning of the word and check them if they correct words
  - If both of the new instances are correct then the prefix is removed
  - Otherwise the original word is returned
- Replicate and Remove (RPR)
  - Remove the prefix and check the remaining word in the lexicon
    - If the word not exist go to the duplicate step
    - If the word exist return the original word
  - Duplicate the initial letter except the lam and check the new word in the lexicon

        - If the word exist, return the original word
        - Else, remove the prefix
    - For the words start with the letter lam, we add both *baa* and *kaf* instead of replicating them
        - If both new instances are incorrect, we remove the first lam.
        - Else keep the original word.

Stemming algorithm:

- Dealing with للـ, prefix:
    - Replace the prefix للـ, with لـ, if the new word is in the lexicon remove the first لـ
    - Else return the original word
- Use the RPR method to remove the conjunctions and prepositions.
- Remove the definite article ال, from the beginning of the word
- If the word starts with سـ, يـ, تـ, نـ, and اـ, generate two instances of the word by adding كـ, and الـ, to the beginning of the word, if either of the new words exist in the lexicon
    - Return original word
    - Else, remove the starting letter
- Repeat the previous step until the condition is not longer exist
- Remove the suffixes ها, هم, هما, هن
- If the word ends with ان, replace it with ين, and remove it only if the word exists in lexicon.
- If the word ends with ات, replace the suffix with ة only if:
    - Removing the suffix produces a word that exists in the lexicon, or
    - Replacing the suffix with ة, produces a word that exists in the lexicon
- If the word ends with ين, remove the suffix only if:
    - Replacing the suffix with ان, produces a word that exist in the lexicon, or
    - Replacing the suffix with ون, produces a word that exist in the lexicon
- If the word ends with ون, remove the suffix only if replacing the suffix with ين, produces a word that exists in the lexicon,

- Else, remove it if the word start with ي, or سي,
- Remove the suffixes وا, ية, يه, ه, ة
- If the word ends with ي, remove the suffix only if:
  - Removing the suffix produces a word that exists in the lexicon, or
  - Replacing the suffix with ها, produces a word that exist in the lexicon, or
  - Replacing the suffix with ه, produces a word that exist in the lexicon

One limitation of his method is that it needs a lexicon contains all the forms of all the words in Arabic language which is very difficult to obtain. He used Microsoft office 2003 proof kit as resource of Arabic words. It contains about 15,500,000 Arabic words.

A.9. Beltagy Stemmer

Samhaa El-Beltagy and Ahmed Rafea [El-Beltagy S., 2009] have proposed a stemming technique that not only removes prefixes and suffixes from the beginning and the end of the word, but also converts the irregular plural form of the word to its singular form.

The stemmer also is a domain specific stemmer which conducts stemming according to the domain of the collection of text to be indexed. The domain specific idea is implemented using a stem list that contains the words and their stems. So, before accepting a stem that produce from a word using stem-based stemmer, the system check whether the produced stem exist in the list or not.

This idea is helpful since words could have different stems on different domain. For example, the word "دقيقة" could have the following different meanings: minute and very fine. In the first sense, stemming is going to harm the word. However, stemming will be good choice for the second meaning.

For simplicity, we refer to the stemmer as *Beltagy Stemmer*. The stemmer first has to be built or trained then it can be used for stemming. The construction of the stemmer is done using a subset of the documents to be stemmed. The construction done as follow:

- A subset of documents from the corpus to be stemmed is selected for the stem list building.
- Through a user interface, the user checks the stem list to verify its correctness.
- The user can provides stems he wanted for specific words that he wants to stem them in a particular way.

In the second phase (operational phase),

- Stemming can be done by checking whether the possible stem exists in the stem list or not.
- The stemmer has two modes; restrict mode (the original word is returned if the word does not exist in the stem list or the corpus) and light stemming mode (the stem rather than the original word is returned if the word I not exist in the stem list or the corpus).

Stemming algorithm is done in the two phases. In the training phase the stemmed word checked only in the corpus, but in the stemming phase the word checked in the stem list and the in the document:

- Remove the following prefixes if the length of the remaining word is greater than or equal 2: ال , وال, بال, كال, فال, لل, وبال, ولل, وكال, وفال
- Remove the prefixes و, ك, ف, ل, ب, only if the remaining word exists in the stem dictionary or in the input document collection.
- Remove the prefixes لا, if the length of the remaining word is greater than or equal 2
- Remove the suffixes يات, ات, اته, only if
    - The remaining word exists in the stem dictionary or in the input document collection, or,
    - Adding ة, to the remaining word and the modified word exist in the stem dictionary or in the input document collection,
- Remove the suffixes ها, ون, وا , ين , ان, ية , يه , هم, ي, ه , ة only if
    - The remaining word exists in the stem dictionary or in the input document collection, or,
    - If the remaining word ends with ت, replace the ت, with ة and check if it exists in the stem dictionary or in the input document collection,

There are two limitations for this stemming technique. First, the stemmer has to be used with a specific domain. It cannot be used as a general stemmer. This means that there is no sense to be compared with the Light10 stemmer. The second limitation is the stem list which is built during the construction phase has to have the user intervention to edit the mistakes done by the stemmer.

REFERENCES

Al-Assi R., (2010) Do you think that the Internet speaks Arabic?, find at: < http://webcache.googleusercontent.com/search?q=cache:24DNO3jBLoUJ:andfaraway.net/blog/2010/07/14/do-you-think-that-the-internet-speaks-arabic/+What+is+the+status+of+Arabic-language+content+on+the+Internet%3F&cd=4&hl=en&ct=clnk&gl=eg&source=www.google.com.eg>

Abdulla, Rasha A. (2007) The Internet in the Arab World: Egypt and Beyond. New York: Peter Lang,. Pp. xxi, 175. ISBN 978-0-8204-8673-4.

Abdulla, R. (2008). *Arabic language use and content on the Internet.* [In English]. Report presented to the Bibliotheca Alexandrina as part of the Access to Knowledge project.

Abu El-Khair I., (2006), Effects of Stop Words Elimination for Arabic Information Retrieval: A Comparative Study, *International Journal of Computing & Information Sciences*, pages 119-133.

Abu-Salem, H., Al-Omari, M., and Evens, M., (1999), Stemming methodologies over individual query words for Arabic information retrieval. *JASIS*, 50 (6), pp. 524-529.

Al-Ameed H., Al-Ketbi O. Shaikha, Al-Kaabi A. Amna, Al-Shebli S. Khadija, Al-Shamsi F. Naila, Al-Nuaimi H. Noura, Al-Muhairi S. Shaikha, (2005), Arabic Light Stemmer: A new Enhanced Approach, *The second international conference on innovations technology (IIT'05)*.

Al-Hajjar A., Hajjar M., Zreik K., (2009), Classification of Arabic Information Extraction methods, Proceedings of the Second International Conference on Arabic Language Resources and Tools.

Aljlayl, M., & Frieder, O., (2002), On Arabic search: Improving the retrieval effectiveness via light stemming approach. *In Proceedings of the 11th ACM International Conference on Information and Knowledge Management*, Illinois Institute of Technology (pp. 340–347). New York: ACM Press.

Al-Kharashi, I. and Evens, M. W., (1994), Comparing words, stems, and roots as index terms in an Arabic information retrieval system. *JASIS*, 45 (8), pp. 548-560.

Al-Sughaiyer I., and Ibrahim A. Al-Kharashi, (2003), Arabic morphological analysis techniques: a comprehensive survey, Journal of the American Society for Information Science and Technology, v.55 n.3, p.189-213.

Anand S. and B. Mobasher, (2007), Introduction to Intelligent Techniques for Web Personalization, ACM Transactions on Internet Technology, Vol. 7, No. 4, Article 18, Publication date.

Armstrong R., Dayne Freitag, Thorsten Joachims and Tom Mitchell. (1995), WebWatcher: A Learning Apprentice for the World Wide Web. In AAAI Press. Pages 6-12.

Asnicar, F. A. and Tasso, C. (1997) ifWeb: A prototype of user model-based intelligent agent for document filtering and navigation in the World Wide Web. In: P. Brusilovsky, J. Fink and J. Kay (eds.) Proceedings of Workshop "Adaptive Systems and User Modeling on the World Wide Web" at 6th International Conference on User Modeling, UM97, Chia Laguna, Sardinia, Italy, June 2, 1997, Carnegie Mellon Online, pp. 3-11.

Baeza-Yates, Ricardo, and Berthier Ribeiro-Neto. (1999). *Modern Information Retrieval*. AddisonWesley.

Bakry, S. H. and Bakry T.H. (2010) 'Assessment views of the Arabic content of the internet: directions for future development', Int. J. Arab Culture, Management and Sustainable Development, Vol. 1, No. 4, pp.359–374.

Marko Balabanovic. (1997). An adaptive Web page recommendation service. In Proceedings of the first international conference on Autonomous agents (AGENTS '97). ACM, New York, NY, USA, 378-385.

Barrilero, M.; Uribe, S.; Alduan, M.; Sanchez, F.; Alvarez, F., (2011), "In-network content based image recommendation system for Content-aware Networks," *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on* , vol., no., pp.115-120, 10-15.

Bartlett, J., and Beck, E. M., eds. (1980). Familiar Quotations. Boston, Mass.: Little, Brown.

Basu C., Haym Hirsh, and William Cohen. (1998). Recommendation as classification: using social and content-based information in recommendation. In Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence (AAAI '98/IAAI '98). American Association for Artificial Intelligence, Menlo Park, CA, USA, 714-720.

Beesley K., (1996). Arabic finite-state morphological analysis and generation. In *Proceedings of the 16th conference on Computational linguistics - Volume 1* (COLING '96), Vol. 1. Association for Computational Linguistics, Stroudsburg, PA, USA, 89-94.

Bergman, Michael K, (2001), "The Deep Web: Surfacing Hidden Value", The Journal of Electronic Publishing, <http://quod.lib.umich.edu/cgi/t/text/text-idx?c=jep;view=text;rgn=main;idno=3336451.0007.104>

Bhogal J., Macfarlane A., Smith P., (2007), A review of ontology based query expansion, *Information Processing and Management* 43, 866–886.

Billsus D., Michael J. Pazzani, (1999), "A Personal News Agent that Talks, Learns and Explains", Third International Conference on Autonomous Agents.

Boone G.. (1998). Concept features in Re:Agent, an intelligent Email agent. In *Proceedings of the second international conference on Autonomous agents* (AGENTS '98), Katia P. Sycara and Michael Wooldridge (Eds.). ACM, New York, NY, USA, 141-148.

Bradshaw J., (1997), An introduction to software agents. In Software agents, Jeffrey M. Bradshaw (Ed.). MIT Press, Cambridge, MA, USA 3-46.

Brin S. and Lawrence Page. (1998), The anatomy of a large-scale hypertextual Web search engine. *Comput. Netw. ISDN Syst.* 30, 1-7 (April 1998), 107-117.

Broder A., (2002), A taxonomy of web search. *SIGIR Forum* 36, 2, 3-10.

Budzik J. and Hammond K., (1999), Watson: Anticipating and contextualizing information needs. In Proceedings of the Sixty-second Annual Meeting of the American Society for Information Science, Information Today, Inc.

Büttcher S., Charles L. A. Clarke, Gordon V. Cormack, (2010), Information Retrieval: Implementing and Evaluating Search Engines, The *MIT Press*, ISBN: 0262026511, chapter 15.

Casasola E., (1998), ProFusion PersonalAssistant: An Agent for Personalize Information Filetering on the WWW. Master's thesis, The University of Kansas, Lawrence, KS.

Cesarano C., d'Acierno A. & Picariello A., (2003), An Intelligent Search Agent System for Semantic Information Retrieval on the Internet, 2003, WIDM'03, November 7–8, New Orleans, Louisiana, USA.

Chaffee J., and Susan Gauch. (2000), Personal ontologies for web navigation. In Proceedings of the ninth international conference on Information and knowledge management (CIKM '00). ACM, New York, NY, USA, 227-234.

Challam V. and Gauch S., (2004), Contextual Information Retrieval Using Ontology-Based User Profiles.

Chen L. and Katia Sycara, (1998), WebMate: a personal agent for browsing and searching. InProceedings of the second international conference on Autonomous agents (AGENTS '98), Katia P. Sycara and Michael Wooldridge (Eds.). ACM, New York, NY, USA, 132-139.

Chen, Z., Meng, X., Zhu, B. & Fowler, R. (2000). WebSail: From On-Line Learning to Web Search. In Proceedings of the 2000 International Conference on Web Information Systems Engineering.

Chen C., Chen M., Sun Y., (2002), PVA: A Self-Adaptive Personal View Agent. Journal of Intelligent Information Systems, p.173-194.

Chen, A., and Gey, F.,(2002), Building an Arabic stemmer for information retrieval. *In TREC 2002. Gaithersburg: NIST*, pp 631-639.

Chesnais P. R., Mucklo M.J., Sheena J.A., (1995), The Fishwrap Personalized News System with Mathew Mucklo and Jonathan Sheena. Proceedings of the 1995 2nd International Workshop on Community Networking. Princeton, NJ, June 1995, pp 275-282

Chirita P., Wolfgang Nejdl, Raluca Paiu, Christian Kohlschutter, (2005), Using ODP Metadata to Personalize Search.

Cilibrasi, R. L. and Vitanyi, P.M.B. (2007) The Google Similarity Distance. IEEE Transactions on Knowledge and Data Engineering 19(3), 370-383.

Cooley R., P-T. Tan., and J. Srivastava.(1999), WebSIFT: The Web site information filter system. In Workshop on Web Usage Analysis and User Profiling (WebKKD99), San Diego.

Cunningham, P., Bergmann, R., Schmitt, S., Traphoner, R., Breen, S. & Smyth, B. (2001). WebSell: Intelligent Sales Assistants for the World Wide Web. In E-2001.

Dai H., (2006), Detecting online commercial intention, WWW'06.

Darwish, K., Doermann, D., Jones, R., Oard, D., and Rautiainen, M., (2001), *TREC-10 experiments at Maryland: CLIR and video*. In TREC 2001. Gaithersburg: NIST.

Darwish, K., (2002a), Building a shallow morphological analyzer in one day. *ACL 2002 Workshop on Computational Approaches to Semitic languages*.

Darwish, K. and Oard, D.W., (2002b), CLIR Experiments at Maryland for TREC-2002: Evidence combination for Arabic-English retrieval. *In TREC 2002. Gaithersburg: NIST*, pp 703-710.

Darwish K., Hassan H., and Emam O., (2005), Examining the Effect of Improved Context Sensitive Morphology on Arabic Information Retrieval. *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 25–30, Ann Arbor.

Durfee E. and V. Lesser., (1989), Negotiating task decomposition and allocation using partial global planning. In L. Gasser and M. Huhns, editors, Distributed Artificial Intelligence Volume II, pages 229–244. Pitman Publishing: London and Morgan Kaufmann: San Mateo, CA.

Edakroury A., (2010), The Arabic Content in the Internet: Informational Perspective, Cybrarians Journal,

أيمن شعبان الدكروري، المحتوى العربي على العنكبوتية العالمية : منظور معلوماتي، Cybrarians Journal، العدد 23، سبتمبر 2010، تاريخ الاطلاع 2011/05/16

Eirinaki, M. and Vazirgiannis, M., (2003), Web Mining for Web Personalization, ACM Transactions on Internet Technology, Vol. 3, No. 1, Pages 1–27.

Eldesouki M., Arafa W. and Darwish K., (2009), Stemming techniques of Arabic Language: Comparative Study from the Information Retrieval Perspective. *The Egyptian Computer Journal , Vol. 36 No. 1*.

Eldesouki M., Arafa W., Darwish K. and Gheith M., (2011), Using Wikipedia for Representing Arabic Documents, *Proceedings of the Arabic Language Technology International Conference (ALTIC 2011)*, Alexandria, Egypt.

El-Beltagy S., Rafea A., (2009), A Framework For The Rapid Development Of List Based Domain Specific Arabic Stemmers, *Proceedings of the Second International Conference on Arabic Language Resources and Tools*.

Eliassi-Rad T., (2001), Building Intelligent Agents that Learn to Retrieve and Extract Information. *PhD Thesis*. AAI3033271. Computer Sciences Department. University of Wisconsin, Madison, WI.

Etzioni O. and Daniel Weld. (1994). A softbot-based interface to the Internet. Commun. ACM37, 7, 72-76.

Etzioni O. and Daniel S. Weld, (1995). Intelligent Agents on the Internet: Fact, Fiction, and Forecast. *IEEE Expert: Intelligent Systems and Their Applications* 10, 4, 44-49.

Foner, L. (1993), "What's an Agent, Anyway? A Sociological Case Study", Agents Memo 93-01, MIT Media Lab, Cambridge, MA.

Frakes W. and R. Baeze-Yates., (1992), *Information retrieval: Data Structures & Algorithms*. Prentice Hall, Englewood Cliffs, NJ, USA,.

Franklin, S., and Graesser, A. (1996). Is It an Agent or Just a Program? A Taxonomy for Autonomous Agents. In Proceedings of the Third International Workshop on Agent Theories,Architectures, and Languages. New York: Springer-Verlag.

Fröschl, C. (2005) User Modeling and User Profiling in Adaptive E-learning Systems, Master Thesis, University of Technology, Graz.

Fu G., (2005), Ontology-Based Spatial Query Expansion in Information Retrieval ODBASE: OTM Confederated International Conferences.

Gauch, S., Wang,G.,And Gomez, M. (1996). Profusion: intelligent fusion from multiple, distributed search engines. J. Univers. Comput. Sci. 2, 9, 637-649.

Gey, F. C. and Oard, D. W., (2001), The TREC-2001 cross-language information retrieval track: Searching Arabic using English, French, or Arabic queries. *In TREC 2001*. *Gaithersburg: NIST*.

Gilbert, D.; Aparicio, M.; Atkinson, B.; Brady, S.; Ciccarino, J.; Grosof, B.; O'Connor, P.; Osisek, D.; Pritko, S.; Spagna, R.; and Wilson, L., (1995), IBM Intelligent Agent Strategy, IBM Corporation.

Gilbert, D.,(1997), "Intelligent Agents: The Right Information at the RightTime." IBM white paper.

Glover E., Gary W. Flake, Steve Lawrence, William P. Birmingham, Andries Kruger, C. Lee Giles, and David M. Pennock.,(2001), Improving Category Specific Web Search by Learning Query Modifications. SAINT, p.23-34.

Godoy D. and Analia Amandi. (2005), User profiling in personal information agents: a survey.*Knowl. Eng. Rev.* 20, 4, 329-361.

Goecks J. and Jude Shavlik, (1999), Automatically Labeling Web Pages Based on Normal User Actions. In Proceedings of the IJCAI Workshop on Machine Learning for Information Filtering, Stockholm, Sweden.

Goldberg, D., Nichols, D., Oki, B.M. and Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. Communications of the ACM 35(12):61-70.

Good N., J. Ben Schafer, Joseph A. Konstan, Al Borchers, Badrul Sarwar, Jon Herlocker, and John Riedl. (1999). Combining collaborative filtering with personal agents for better recommendations. In *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence* (AAAI '99/IAAI '99). American Association for Artificial Intelligence, Menlo Park, CA, USA, 439-446.

Greengrass E., (2000), "Information Retrieval: A survey". DOD Technical Report TR-R52-008-001.

Gulli A. and A. Signorini. (2005). The indexable web is more than 11.5 billion pages. In *Special interest tracks and posters of the 14th int'l conf on World Wide Web* (WWW '05). ACM, New York, NY, USA, 902-903.

Han E., Boley D., Gini M., Gross R., Hastings K., Karypis G., Kumar V., Mobasher B., and Moore J.. (1998). WebACE: a Web agent for document categorization and exploration. In Proceedings of the second international conference on Autonomous agents (AGENTS '98), Katia P. Sycara and Michael Wooldridge (Eds.). ACM, New York, NY, USA, 408-415.

Hayes, C. & Cunningham, P. (1999). Smart Radio - a Proposal. In Trinity College Dublin, Computer Science, Technical Report, TCD-CS-1999-24.

Hayes, C. & Cunningham, P. (2000). Smart Radio: Building Music Radio on the Fly. In Proceedings of Expert Systems 2000 (ES2000). Cambridge, UK.

Heckmann, D., (2005), Ubiquitous User Modeling. PhD Thesis, Saarland University, Saarbrücken, Germany.

Hermans, B. (1996) "Intelligent Software Agents on the Internet."(updated 2000). hermans.org/agents/h22.htm (accessed July 2011).

Hill, W., Stead, L., Rosenstein, M. & Furnas, G. (1995). Recommending and Evaluating Choices in a Virtual Community of Use. In Proceedings of CHI'95, 194–201. Denver.

Hmeidi, I., Kanaan, G. and M. Evens (1997) Design and Implementation of Automatic Indexing for Information Retrieval with Arabic Documents. *Journal of the American Society for Information Science*, 48/10, pp. 867-881.

Hu J. and P. Chan, (2008), Personalized Web Search by Using Learned User Profiles in Re-ranking, Workshop on Knowledge Discovery on the Web, KDD conference pp. 84-97.

Huhns, M. N. & Singh, M. P. (1994), "Distributed Artificial Intelligence for Information Systems", CKBS-94 Tutorial, June 15, University of Keele, UK.

Hull, D., (1993), Using Statistical Testing in the Evaluation of Retrieval Performance. In Proc. of the 16th ACM/SIGIR Conference, pages 329-338.

Ivory, M. Y, and Hearst, M. A. (2002). Statistical profiles of highly-rated Web sites. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pages 367- 374. Minneapolis, Minnes ota.

Janca P., (1995), "Pragmatic Application of Information Agents: BIS Strategic Decisions.

Jansen B., Tracy Mullen, Amanda Spink, and Jan Pedersen. (2006). Automated gathering of Web information: An in-depth examination of agents interacting with search engines. ACM Trans. Internet Technol. 6, 4, 442-464.

Joachims T., D. Freitag, T. Mitchell, (1997), Webwatcher :A tour guide for the World Wide Web. In :Proc. 15 th In t erna t ional Conferenceon Art i ficial In t ell igence, Nagoya, Japan, pp. 770 - 775

Kadri, Y., and Nie, J. Y. (2006), Effective stemming for Arabic information retrieval". The challenge of Arabic for NLP/MT Conference, The British Computer Society. London, UK.

Kamba T., K. Bharat and M. Albers. (1995), The Krakatoa Chronicle – An Interactive, Personalized Newspaper on the Web. In Proceedings of the 4th International WWW Conference, pp. 159-170.

Khoja, S. and Garside, R., (1999), Stemming Arabic text. *Computing* Department, Lancaster University, Lancaster.

Shereen Khoja, (2001), APT: Arabic Part-of-speech Tagger. Proc. of the Student Workshop at NAACL'01.

Kim H. and P.K. Chan,(2003), Learning implicit user interest hierarchy for context in personalization, Proc. of International conference on Intelligent User Interfaces (IUI), Miami, Florida.

Kim H. & P.,(2006), Chan, Personalized Ranking of Search Results with Learned User Interest Hierarchies from Bookmarks, In Advances in Web Mining and Web Usage Analysis (LNCS 4198), O. Nasraoui, O. Zaine, M. Spiliopolou, B. Mobasher, B. Masand & P. Yu (editors) pp 158-176, Springer.

Kim H. and Philip K. Chan. (2008). Learning implicit user interest hierarchy for context in personalization. Applied Intelligence 28, 2 (April 2008), 153-166.

Kleinberg, J. M. (1998). Authoritative sources in a hyperlinked environment. In Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms, p ages 668- 677. San Francisco , California.

Kleinberg, J. M. (1999). Authoritative sources in a hyperlinked environment. Journal of the ACM, 46(5):604- 632.

Kobsa, A., Koenemann, J. & Pohl, W. (2001). Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships. The Knowledge Engineering Review 16: 111–155.

Koch N., (2000), Software Engineering for Adaptive Hypermedia Systems. PhD thesis, Ludwig-Maximilians-University Munich/Germany.

Konstan J., Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John Riedl. (1997). GroupLens: applying collaborative filtering to Usenet news. Commun. ACM 40, 3, 77-87.

Krovetz R., (1993) "Viewing morphology as an inference process," in R. Korfhage et al., *Proc. 16th ACM SIGIR Conference*, Pittsburgh, pp. 191-202.

Krulwich, B. (1997). LifeStyle Finder: Intelligent User Profiling Using Large-Scale Demographic Data. AI Magazine 18(2): 37–45.

Kurki T., S. Jokela, R. Sulonen and M. Turpeinen.(1999), Agents in Delivering Personalized Content Based on Semantic Metadata. In Proceedings of the 1999 AAAI Spring Symposium Workshop on Intelligent Agents in Cyberspace, Stanford, USA, pp. 84-93.

Lang, K. (1995). NewsWeeder: Learning to Filter News. In Proc. of the 12th International Conference on Machine Learning, 331–339. Lake Tahoe, CA.

Larkey, Leah S., Ballesteros, Lisa, and Connell, Margaret. (2002) Improving Stemming for Arabic Information Retrieval: Light Stemming and Co-occurrence Analysis. *In Proceedings of the 25th Annual International*

Conference on Research and Development in Information Retrieval (SIGIR 2002), Tampere, Finland, August 11-15, 2002, pp. 275-282.

Larkey, S. L., Ballesteros, L., and Connell, E. M. (2005), Light stemming for Arabic information retrieval. *Arabic Computational Morphology: Knowledge-based and Empirical Methods*.

Larkey L., Lisa Ballesteros and Margaret E. Connell. (2007). Light stemming for Arabic information retrieval. In Abdelhadi Soudi, Antal van den Bosch, and Günter Neumann, editors, Arabic Computational Morphology: Knowledge-based and empirical method, volume 38 of Text, Speech and Language Technology, Springer Verlag.

Lempel, R., and Moran, S. (2000). The stochastic approach for link- structure analysis (SALSA) and the TKC effect. Computer Networks, 33(1-6):387- 40l.

Li L. Zhenglu Yang, Botao Wang, and Masaru Kitsuregawa. (2007). Dynamic adaptation strategies for long-term and short-term user profile to personalize search. In Proc. of the joint 9th Asia-Pacific web and 8th int'l conf on web-age information management conf on Advances in data and web management (APWeb/WAIM'07), Dong G., Lin X., Wang W., Yun Yang, and Jeffrey Xu Yu (Eds.). Springer-Verlag, Berlin, Heidelberg, 228-240.

Lieberman H. (1995). Letizia: an agent that assists web browsing. In Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1 (IJCAI'95), Chris S. Mellish (Ed.), Vol. 1. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 924-929.

Lieberman H., (1997). Autonomous interface agents. In Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '97). ACM, New York, NY, USA, 67-74.

Lieberman, H., Van Dyke, N. W. & Vivacqua, A. S. (1999). Let's Browse: A Collaborative Web Browsing Agent. In Proceedings of International Conference on Intelligent User Interfaces, 924–929.

Liu F., Clement Yu, and Weiyi Meng. (2004). Personalized Web Search For Improving Retrieval Effectiveness. IEEE Trans. on Knowl. and Data Eng. 16, 1 (January 2004), 28-40.

Lovins, J. (1968) *Development of a Stemming Algorithm*, Mechanical Translation and Computational Linguistics, 11: 22-31

Maes, P. (1994), "Agents that Reduce Work and Information Overload", *Communications of the ACM* **37** (7), 31-40.

Maes, P. (1995) "Artificial Intelligence Meets Entertainment: Life-Like Autonomous Agents." Communications of the ACM.

Manning C., Prabhakar Raghavan and Hinrich Schütze. (2008). Introduction to Information Retrieval. Cambridge University Press, ISBN: 0521865719

McGowan J., Nicholas Kushmerick, and Barry Smyth. (2002). Who Do You Want to Be Today? Web Personae for Personalised Information Access. In Proc. of the Second International Conference on Adaptive Hypermedia and

Adaptive Web-Based Systems (AH '02), Paul De Bra, Peter Brusilovsky, and Ricardo Conejo (Eds.). Springer-Verlag, London, UK, UK, 514-517.
Meng X., Z. Chen. (1999), Improve Web Search accuracy using personalized profiles.
Milne, D. and Witten, I.H. (2008a) An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In Proc. of the first AAAI Workshop on Wikipedia and Artificial Intelligence (WIKIAI'08), Chicago, I.L.
Milne, D. and Witten, I.H. (2008b) Learning to link with Wikipedia. In Proc. of the ACM Conference on Information and Knowledge Management (CIKM'08), Napa Valley, California.
Minio, M. & Tasso, C. (1996). User Modeling for Information Filtering on Internet Services: Exploiting an Extended Version of the UMT Shell. In UM96 Workshop on User Modeling for Information Filtering on the WWW.
MCIT, (2006). Ministry of Communications and Information Technology Yearbook. Cairo.
Mladenic, D., (1996) Personal WebWatcher: Implementation and Design (uncompressed) Technical Report IJS-DP-7472, Department of Intelligent Systems, J.Stefan Institute, Slovenia.
Mobasher B. and S. S. Anand, (2005b). Intelligent Techniques for Web Personalization. In Intelligent Techniques for Web Personalization, Eds. Lecture Notes in Computer Science, vol. 3169. Springer-Verlag, 1–36.
Mobasher B., (2007), Data mining for Web personalization. In The Adaptive Web: Methods and Strategies of Web Personalization, P. Brusilovsky, A. Kobsa, and W. Nejdl, Eds. Lecture Notes in Computer Science, vol. 4321. Springer-Verlag, Berlin, Heidelberg, Germany/New York, NY.
Montebello M., W. Gray and S. Hurley., (1998), A Personable Evolvable Advisor for WWW Knowledge-Based Systems. In Proc. of the 1998 International Database Engineering and Application Symposium (IDEAS'98), pp. 224-233.
Morita M., Yoichi Shinoda, (1994), Information filtering based on user behavior analysis and best match text retrieval, Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, p.272-281, Dublin, Ireland
Moukas A., (1996), Amalthaea: Information discovery and filtering using a multiagent evolving ecosystem. In Proceedings of the Cottference on Practical Applications of Agents and Mul. tlagent Technology.
Moukas A., (1997), Amalthaea: Information discovery and filtering using a multiagent evolving ecosystem," J. Appl. Intell. vol.11(5) pp. 437-457.
Moulin, B., and Chaib-draa, B. (1996). An Overview of Distributed Artificial Intelligence. In Foundations of Distributed Artificial Intelligence, eds. G. M. P. O'Hare and N. R. Jennings, 3–55. New York: Wiley, pp. 8-9.
Newell A., (1982). The Knowledge Level. Artificial Intelligence, 18(1):87-127.
Nichols D., (1997), Implicit Rating and Filtering. In Proceedings of the Fifth DELOS Workshop on Filtering and Collaborative Filtering.

Nwana H., (1996), Software Agents: An Overview, Knowledge Engineering Review, Vol. 11, No 3, pp.1-40, Cambridge University Press.

Nwesri A., S.M.M Tahaghoghi, Falk Scholer,(2005) Stemming Arabic Conjunctions and Prepositions, In Mariano Consens and Gonzalo Navarro (eds.), *Lecture Notes in Computer Science - Proceedings of the Twelfth International Symposium on String Processing and Information Retrieval (SPIRE'2005)*, Buenos Aires, Argentina, 3772:206-217.

Nwesri A., S.M.M. Tahaghoghi and Falk Scholer, (2007) Arabic Text Processing for Indexing and Retrieval, *Proceedings of the International Colloquium on Arabic Language Processing*, Rabat, Moroc, 18-19.

O'Riordan A. and Sorensen H., (1995), An Intelligent Agent for High-Precision Text Filtering.

Oard, D.W. and Marchionini, G. (1996), A Conceptual Framework for Text Filtering, Technical Report CAR-TR-830, Human Computer Interaction Laboratory, University of Maryland at College Park.

Paice, C.D. (1990) *Another Stemmer*, SIGIR Forum, 24: 56-61.

Page, L., Brin, S., Motwani, R., and Winograd, T., (1998), The Pagerank citation ranking: Bringing order to the web, Technical report, Stanford University.

Parent S., B. Mobasher, and S. Lytinen.(2001), An adaptive agent for web exploration based on concept hierarchies. In Proceedings of the Ninth International Conference on Human Computer Interaction, New Orleans, LA.

Petrie, C. J. (1996). Agent-Based Engineering, the Web, and Intelligence, *IEEE Expert*, 11(6): 24-29.

Pazzani, M, Muramatsu, J, and Billsus, D., (1996), "Syskill & Webert; Identifying interesting Web sites", In *Proc of the National Conf. on AI (AAA I96)*.

Pazzani, M. & Billsus, D. (1997). Learning and Revising User Profiles: The Identification of Interesting Web Sites. Machine Learning 27: 313–331.

Porter, M.F. (1980) *An Algorithm for Suffix Stripping*, Program, 14(3): 130-137.

Pretschner A. and Gauch S.,(1999), Personalization on the Web, University of Kansas, Tech. Report.

Pretschner A., (1999a). Ontology based personalized search. Master's thesis, University of Kansas, Electrical Engineering and Computer Science.

Pretschner A., and Gauch S., (1999b), Ontology Based Personalized Search. In Proc. of the 11th IEEE Int'l Conf on Tools with AI (ICTAI), pp. 391-398.

Ramanathan K, Giraudi J., Gupta A., (2008), Creating hierarchical user profiles using Wikipedia, HP Laboratories

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. & Riedl, J. (1994). Grouplens: An Open Architecture for Collaborative Filtering of Netnews. In Proceedings of ACM CSCW'94, 175–186.

Richardson, M., Prakash, A., and Brill, E. (2006). Beyond PageRank: Machine learning for static ranking. In Proceedings of the 15th International World Wide Web Conference, pages 707-715. Edinburgh, Scotland.

Rose D. and Danny Levinson. (2004). Understanding user goals in web search. In *Proceedings of the 13th international conference on World Wide Web* (WWW '04). ACM, New York, NY, USA, 13-19.

Rucker J. and Marcos J. Polanco. (1997). Siteseer: personalized navigation for the Web.Commun. ACM 40, 3, 73-76.

Russell, J., and P. Norvig, (1995), Artificial Intelligence: A Modern Approach. Upper Saddle River, NJ: Prentice Hall.

Sakagami H. and Tomonari Kamba, (1997) Learning personal preferences on online newspaper articles from user behaviors, Comput. Netw. ISDN Syst., 29(8-13):1447–1455.

Salton, G. & McGill, M.J. (1983). Introduction to Modern Information Retrieval. New York: McGraw-Hill.

Selker T., (1994) "A Teaching Agent that learns" Communications of the ACM 37 (7) pp 92-99.

Shardanand U., (1994), "Social Information Filtering for Music Recommendation", MIT EECS M. Eng. thesis, also TR-94-04, Learning and Common Sense Group, MIT Media Laboratory.

Shardanand U. and Pattie Maes. (1995). Social information filtering: algorithms for automating "word of mouth". In Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '95), Irvin R. Katz, Robert Mack, Linn Marks, Mary Beth Rosson, and Jakob Nielsen (Eds.). ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 210-217.

Sheth, B., and Maes, P., (1993), Evolving agents for personalized information filtering. In Proceedings of the 9th IEEE Conference on Artificial Intelligence for Applications (Orlando, Fla).

Sheth B., (1994) A Learning Approach to Personalized Information Filtering. Master's thesis, Massachusetts Institute of Technology.

Shoham, Y. (1997). An Overview of Agent-oriented Programming. In Software Agents, ed J. M. Bradshaw. Menlo Park, Calif.: AAAI Press.

Sieg, A., Mobasher, B., Burke, R., (2004), Inferring user's information context: Integrating user profiles and concept hierarchies. In: Proceedings of the 2004 Meeting of the International Federation of Classification Societies.

Sieg, A.,Mobasher, B., Burke, R., Prabhu, R.G., Lytinen, S., (2005), Representing user information context with ontologies. In: Proceedings of HCI International Conference, PP 210–217

Sieg A., B. Mobasher, and R. Burke, (2007) "Representing context in web search with ontological user profiles," in *Proc of the Sixth Int'l and Interdisciplinary Conference on Modeling and Using Context*, Roskilde, Denmark.

Sieg A., B. Mobasher, and R. Burke. (2007) Ontological user profiles for personalized web search. In Proceedings of the 5th Workshop on Intelligent Techniques for Web Personalization, Vancouver, Canada.

Sieg A., B. Mobasher, and R. Burke,(2007) “Web search personalization with ontological user profiles,” in ACM Sixteenth Conference on Information and Knowledge Management, CIKM 2007, Lisbon, Portugal.

Smith D., A. Cypher and J. Spohrer (1994) “Programming Agents without a programming language” Communications of the ACM 37 (7) pp 55-67.

Sorensen H. and M. McElligot, (1997) PSUN: A Profiling System for Usenet News. CKIM '95 Workshop on Intelligent Information Agents, Baltimore, Maryland, December 1995. (http://odyssey.ucc.ie/filtering/psun.ps)

Sorensen H., O’Riordan A. and O’Riordan C., (1997) Profiling with INFOrmer Text Filtering,.

Speretta M. and Susan Gauch. (2005). Personalized Search Based on User Search Histories. In Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI '05). IEEE Computer Society, Washington, DC, USA, 622-628.

Spink, A., & Jansen, B.J. (2004). "A study of Web search trends". Webology, 1(2), Article 4.

Stefani A. and C. Strappavara, (1998) Personalizing Access to Web Sites: The SiteIF Project. In Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia HYPERTEXT’98, June.

Sugiyama K., Kenji Hatano, and Masatoshi Yoshikawa. 2004. Adaptive web search based on user profile constructed without any effort from users. In Proceedings of the 13th international conference on World Wide Web (WWW '04). ACM, New York, NY, USA, 675-684.

Taghva K., Elkhoury R., and Jeffrey Coombs. (2005). Arabic Stemming Without A Root Dictionary. In *Proc of the International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume I - Volume 01* (ITCC '05), Vol. 1. IEEE Computer Society, Washington, DC, USA, 152-157.

Thomas C., and Fischer G. (1997) Using Agents to Personalize the Web. In Proceedings of the 2nd International Conference on Intelligent User Interfaces, Orlando, Florida, USA, , pp.53-60

Trajkova J., and S. Gauch, (2004) Improving Ontology-Based User Profiles. RIAO, Vaucluse, France, April 26-28, pp. 380-389.

TREC, 2001, The TREC-2001 Cross-Language Information Retrieval Track: Searching Arabic Using English, French or Arabic Queries, page 16 F.C. Gey, University of California, Berkeley D.W. Oard, University of Maryland, College Park

TREC, 2002, The TREC 2002 Arabic/English CLIR Track, page D.W. Oard, University of Maryland, College Park F.C. Gey, University of California, Berkeley

Upstill, T., Craswell, N., and Hawking, D. (2003). Query-independent evidence in home page finding. *ACM Transactions on Information Systems*, 21(3):286- 313

Wahlster, W. and Kobsa, A. (1989). User models in dialog systems. In Kobsa, A. andWahlster,W., editors, *User Models in Dialog Systems*, pages 4–34. Springer, Berlin. (Symbolic Computation Series).

Wardrip-Fruin, Noah and Nick Montfort, ed, (2003), The New Media Reader, Section 54, The MIT Press, ISBN 0-262-23227-8.

Whitaker B., (2010), What's Really Wrong With the Middle East, Saqi Books, ISBN 978-0-86356-624-0

Whitten J., Kevin C. Dittman, and Lonnie D. Bentley., (2004), *Systems Analysis and Design Methods*. Chapter 13

Widyantoro D.H., (1999). *Learning User Profile in Personalized News Agent*. Master's thesis, Department of Computer Science, Texas A&M University, College Station, TX.

Widyantoro D., Ioerger, T., and Yen, J. (1999). An Adaptive Algorithm for Learning Changes in User Interests. In S. Gauch (Ed.), *Proc of the Eighth Int'l Conf on Info. and Knowledge Management* (pp. 405-412). New York, NY: ACM Press.

Widyantoro D., T. Ioerger, and J. Yen., (2001), Learning User Interest Dynamics with a Three-Descriptor Representation. Journal of the American Society of Information Science and Technology (JASIST) , Vol 52, No. 3, pp. 212-225.

Wonnacott, R. and Wonnacott, T.,(1990) Introductory Statistics, John Wiley & Sons, Fourth Edition.

Wooldridge, M. J., and Jennings, N. R. (1995a). Agent Theories, Architectures, and Languages: A Survey. In Intelligent Agents: ECAI-94 Workshop on Agent Theories, Architectures, and Languages, eds. M. J. Wooldridge and N. R. Jennings, 1–39. Berlin: Springer- Verlag.

Wooldridge, M. & Jennings, N. (1995b), "Intelligent Agents: Theory and Practice", The Knowledge Engineering Review 10 (2), 115-152.

Wooldridge, M. J., and Jennings, N. R. (1996). Software Agents. IEE Review, January 1996, pp 17-20.

Xu Y., B. Zhang, Z. Chen and K. Wang, (2007) Privacy enhancing personalized web search, WWW'07.

Yan T. and Hector Garcia-Molina. (1995). SIFT: a tool for wide-area information dissemination. In Proceedings of the USENIX 1995 Technical Conference Proceedings(TCON'95). USENIX Association, Berkeley, CA, USA, 15-15.

Zhu D. and Dreher H.. (2008), Improving Web Search by Categorization, Clustering, and Personalization. In Proc of the 4th int'l conf on Advanced Data Mining and Applications (ADMA '08), Changjie Tang, Charles X. Ling, Xiaofang Zhou, Nick J. Cercone, and Xue Li (Eds.). Springer-Verlag, Berlin, Heidelberg, 659-666.