SEARCHING A WEBSITE VIA SPOKEN QUESTIONS

by

Mesut Durukal

B.S, Electrical and Electronics Engineering, Boğaziçi University, 2008

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University

2011

SEARCHING A WEBSITE VIA SPOKEN QUESTIONS

APPROVED BY:

Assoc. Prof. Murat Saraçlar (Thesis Supervisor)	.....
Prof. Levent Arslan	.....
Prof. Lale Akarun	.....

DATE OF APPROVAL: 05.09.2011

ACKNOWLEDGEMENTS

First of all, I would like to thank Assoc. Prof. Murat Saraçlar for his help, encouragement and support throughout my study. Since the beginning of the work, he has enabled me to understand and develop a stronger perception of the relevant subjects. He has been a guide for me whenever I was in trouble with something and tried to find alternative solutions by which I would overcome the drawbacks in a better way. He is definitely a major factor in the accomplishment of this thesis with his suggestions.

I also wish to thank the project members and friends Erinç Dikici, Haşim Sak and Barış Evrim Demiröz for their help in different stages of my study. They were always with me when I needed help. Data preparation environment setup, data collection and integration tasks were done thanks to their support.

İpek Şen, Oya Çeliktutan, Burak Varan, Buğra Tekin, Mehtap Karamanlı are the other friends who contributed to my work. My friends and colleagues were always with me whenever I needed to talk to and with their friendship I was always able to feel comfortable.

Finally I am of course deeply grateful to my family for their sincere love and support.

ABSTRACT

SEARCHING A WEBSITE VIA SPOKEN QUESTIONS

To reach the desired information on a web site can be a very time consuming task when the fact that the site may not even have that information is considered. In addition to the time consuming feature of checking all the pages, the user can also overlook what she looks for which can be regarded as a human factor. For all these reasons, some automatic search engines are developed, at different success levels. In this thesis, a system is proposed which takes the spoken queries from users and forwards them to the page that has the best answer in order to save them to check all the pages manually. In addition, this kind of solution gives users the chance to communicate with the system by asking spoken questions. This is a very important advantage especially for the visually handicapped people. Another service of the system is to prepare the page to be read by the Text-To-Speech unit. In this way, people can talk to the system and find answers to their questions in various platforms like Kiosk applications without any written command. After automatic recognition; the query is classified and keyword assignment is done. Then this keyword is searched in the target pages and finally the most suitable page's address is returned to the user. For the classification process, sufficient amount of data is collected and the system is trained. After the classifier is constructed, the input is classified into one of the previously specified 110 classes. To sum up the entire system, the user interacts verbally with the system and is forwarded to the page which contains information evaluated as a candidate answer to the question. In short, users are supposed to ask questions requesting information, an answer, or a transaction the system is supposed to bring a relevant page accordingly.

ÖZET

SESLİ SORULAR ARACILIĞIYLA WEB SİTESİNDE ARAMA

Bir web sitesinde istenen bilgiye ulaşmak, sitenin merak edilen sorunun cevabını içerdiğinin de garanti olmaması göz önünde bulundurulduğunda, kimi zaman oldukça vakit isteyen bir işlem haline gelmektedir. Tüm sayfalara göz atmak kullanıcıya çok zaman kaybettirebileceği gibi aranan bilginin gözden kaçırılma riski de olduğu için internet üzerinde çeşitli otomatik arama işlemleri geliştirilmiştir. Bu tezde, kullanıcıyı manüel olarak sayfalar arası dolaşmaktan kurtararak sorusunu sesli olarak aldıktan sonra doğrudan ilgili cevapları bulabileceği sayfaya yönlendirmek amaçlanmıştır. Diğer yandan, bu çözümün kullanıcılara sistemle sesli iletişim şansı sunması, özellikle görme engelli kullanıcıları göz önüne alınca, çok önemli bir avantajdır. Bu kapsamda bir diğer amaç da sesli yanıt sistemine okunacak sayfayı hazırlamaktır. Sonuç olarak insanlar Kiosk uygulamaları gibi farklı platformlarda sistemle konuşarak hiçbir yazılı komut olmaksızın sorularına cevap bulabilirler. Yöntem olarak, akustik tanımanın gerçekleştirilmesinin ardından sorunun daha önceden belirlenmiş olan sınıflardan hangisine ait olduğuna karar vermek ve bir anahtar sözcükle ilişkilendirip sayfalarda bu sözcüğü arayarak kullanıcıyı sözcüğün en fazla rastlanıldığı sayfaya yönlendirmek hedeflenmiştir. Sınıflandırma işlemi için veri toplanıp sistem eğitilmiş olup sorunun, önceden belirlenmiş olan 110 sınıftan birine düşürülmesi ve sitede bulunan sayfaları kontrol ederek soru ve cevapla en ilişkin sayfanın kullanıcıya sunulması yoluyla hedefe ulaştırılması odak noktasıdır. Bu yolla kullanıcı sesli şekilde sistemle iletişim kuracak ve çok kısa bir sürede sorusunun cevabına ulaşacak veya liste, kayıt, durum gibi sorgularına ilişkin bilgileri bulacaktır.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . iii
ABSTRACT . . . iv
ÖZET . . . v
LIST OF FIGURES . . . viii
LIST OF TABLES . . . x
LIST OF ACRONYMS/ABBREVIATIONS . . . xii
1. INTRODUCTION . . . 1
1.1. Problem Statement . . . 1
1.2. Previous Work . . . 2
1.3. Contribution of the Thesis . . . 6
1.4. Thesis Outline . . . 8
2. BACKGROUND . . . 9
2.1. Automatic Speech Recognition . . . 9
2.1.1. General ASR Applications . . . 9
2.1.2. SR over Web . . . 10
2.1.3. Recognition Network . . . 12
2.2. Search . . . 13
2.2.1. Site Search . . . 13
2.2.2. Voice Search . . . 16
2.3. Classification . . . 19
2.3.1. Description . . . 19
2.3.1.1. Feature Extraction . . . 20
2.3.1.2. Feature Selection . . . 20
2.3.1.3. Document Representation . . . 20
2.3.2. Approaches . . . 21
2.3.2.1. Decision Trees . . . 21
2.3.2.2. Decision Rules . . . 23
2.3.2.3. Information Gain . . . 24
2.3.2.4. k-Nearest Neighbor . . . 25

2.3.2.5. Bayesian Approaches 26
2.3.2.6. Neural Networks 26
2.3.2.7. Regression-based Methods 27
2.3.2.8. Vector-based Methods 27
2.4. Information Retrieval 28
2.4.1. Description 28
2.4.2. History 30
2.4.3. Techniques 30
2.4.4. Tf-Idf 33
2.4.5. Web Crawling 36
2.4.6. Web Mining 40
3. METHODOLOGY 43
3.1. System Overview 43
3.2. Recognizer 44
3.3. Classifier 46
3.4. Information Retriever 48
3.4.1. Crawler 48
3.4.2. Hunter 49
3.4.3. Ranker 51
3.4.4. Indexer 52
4. EXPERIMENTS & RESULTS 54
4.1. Classification Results 55
4.2. Process Times 55
4.3. Recognition Results 56
4.4. The Effect of Weighting 58
4.5. Overall Results 58
4.6. Discussion 60
5. CONCLUSION 63
APPENDIX A: GRAMMAR 65
APPENDIX B: CLASSES 67
REFERENCES 69

LIST OF FIGURES

Figure 1.1. The overview of the website query system components. 3

Figure 2.1. Data mining from web with speech using a variety of access devices. 11

Figure 2.2. Sample grammar. 13

Figure 2.3. General architecture of a search application. 16

Figure 2.4. An application of voice search for IPTV movie-on-demand. 17

Figure 2.5. An example to decision tree algorithm. 22

Figure 2.6. Question answering components. 33

Figure 2.7. Web crawler. 37

Figure 2.8. Periodic indexing of a web search engine. 38

Figure 2.9. The distributed data mining crawler design. 40

Figure 2.10. The diagram of website analyzer. 42

Figure 3.1. Block diagram of the overall system. 44

Figure 3.2. Block diagram of the response task. 46

Figure 3.3. Overview of the classification process. 47

Figure 3.4. Crawler workbench. 50

Figure 4.1. Main parts of the system. 55

Figure 4.2. Sample sentences vs WER. 56

Figure 4.3. The effect of weighting. 58

Figure 4.4. Speaker based accuracy results of the experiments. 60

Figure 4.5. Results of experiments. 61

LIST OF TABLES

Table 1.1. Possible questions that can be answered inside and outside of the domain. 6

Table 1.2. Survey questions for the evaluation of the system. 6

Table 1.3. Evaluation results. 7

Table 2.1. Basic components of a web search engine. 14

Table 3.1. Form of the grammar. 45

Table 3.2. Mallet training script. 47

Table 3.3. Some of the links found after crawling. 51

Table 3.4. A preview of the index file. 53

Table 4.1. Accuracy results of different classifier trainers in percentages. . . . 55

Table 4.2. Process times needed to check a keyword in all the pages. 57

Table 4.3. Accuracy results of the experiments. 59

Table 4.4. Accuracy results of the experiments (Speaker based). 59

Table 4.5. Accuracy results of the experiments (Gender based). 59

Table 4.6. The effect of classification. 62

Table B.1. Classes. 68

LIST OF ACRONYMS/ABBREVIATIONS

ASLR	Automatic Sign Language Recognition
ASR	Automatic Speech Recognition
CMS	Cepstral Mean Substraction
DM	Dialog Manager
DOM	Document Object Model
HMI	Human Machine Interface
HMM	Hidden Markov Model
HTML	Hyper Text Markup Language
IDF	Inverse Document Frequency
IR	Information Retrieval
LG	Language Generation
MALLET	MAchine Learning for LanguagE Toolkit
QA	Question Answering
SVM	Support Vector Machines
TF	Term Frequency
TREC	Text Retrieval Conference
TTS	Text-to-Speech Synthesis
WER	Word Error Rate

1. INTRODUCTION

Firms and companies spend most of their effort and time to reduce the costs while economic aspects have a great importance in order to have the ability to be competitive in the market. Those ones, who succeeded to minimize the expenses, can behave more freely and can tolerate some other investments. With this point of view, when the difficulties of employing people are also considered, it can be easily seen that solutions against the cost of call centers and employees are very attractive research topics for all organizations.

Recently, people can encounter spoken dialog systems in different applications and forms in various sectors, substituting call centers. However, most commercially available dialog systems are finite state based. That is to say, these systems allow only a sequence of predetermined steps or states. Although it is suitable for simple and well-structured tasks, the resulting system is less flexible and requires a great deal of human effort and time. Because of these disadvantages, spoken dialog is not preferred by most companies.

Alternatively, every kind of information is accessible through the internet and millions of companies invest in developing and maintaining customer-oriented websites. A company website is supposed to attract visitors, serve customers, and meet the company's business needs. Organizations share general information about the company, the products and services the company provides and e-commerce or customer care applications in their web sites. People first look to the relevant web site in order to learn something [1].

1.1. Problem Statement

The chance to learn or check something using internet is a great opportunity in terms of time and effort saved. However sometimes reaching the target information is not that easy. Searching the site and looking at pages may take minutes and does not

guarantee to find what you look for. In addition, the user may overlook what he looks for because a website may consist of millions of web pages, various contents such as menus, tables, images, forms and different formats of text. Therefore, it is difficult to locate the wanted information from this large collection.

To simplify and automate the searching process, search engines are embedded to web sites; nevertheless, the success of this attempt is often doubtful. Additionally, identifying the relevant piece of information from a list of web pages is still time consuming.

In summary, call centers and eventually web sites substituted support teams in companies over time. Moreover, in most of the sites, search engines are integrated to cope with the difficulty of finding the desired information. However, people feel most comfortable when they communicate as they always do. That is to say, speaking with the user is the easiest way to learn what he or she needs and respond accordingly.

In addition to the time and effort consumption, especially visually handicapped users face another problem of the manual search which is the dependence to written commands and reading passages. Since people, even who can read and write, prefer speaking and listening rather than writing and reading because of the simplicity; when people who can not read and write are considered, the necessity to such a system with spoken communication can easily be understood. In short, a lot of manual work is eliminated by asking spoken questions and getting the relevant page on a website.

1.2. Previous Work

There are some smart Kiosk applications that utilize Human Machine Interface such as Touch'n'Speak system developed by the Tampere University (Finland); Memphis Intelligent Kiosk Initiative (MIKI) from the Memphis University (USA); French system Multimodal-Multimedia Automated Service Kiosk (MASK); Multimodal Access to City Help Kiosk (MATCHKiosk) manufactured by AT&T company [2]. These are information enquiry automatons, which can automatically detect presence of a user

in a working area and verbally communicate with him. An information input can be organized simultaneously by touchscreen, keyboard as well as by voice and even by manual gestures.

Fujie, et al. [3] worked on a humanoid which utilizes speech recognition, speech synthesis and image processing. It is an integrated robot responsible from communicating people with spoken language. They tried to cope with two fundamental problems: Real time operation and noise, which can be sourced from air conditioners, internal noise of robot, speech from another person or any other media.

Junlan Feng, et al. [4] designed 'WebTalk', an interactive dialog system model, which uses six major technology components including Website Understanding, Automatic Speech Recognition (ASR), Question Answering (QA), Dialog Manager (DM), Language Generation (LG) and Text-to-Speech synthesis (TTS).

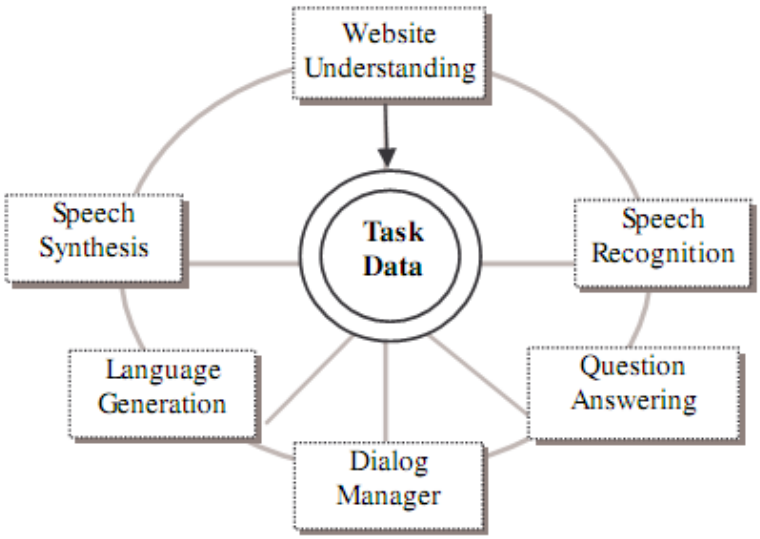

Figure 1.1. The overview of the website query system components [1].

Where ASR, LG and TTS modules are speech related, the website understanding component translates a website in a form that can be used to configure the dialog modules. The task of ASR is to convert the user's input utterance into a sequence of

words. The LG component constructs the message that is to be spoken to the user. The role of the Text-to-Speech component is to translate the text response into spoken form [4].

In WebTalk, there are a number of challenges including constructing the language models that reflect human-machine interaction by using only the web data, designing a language generation component in a manner that can intelligently present the system's knowledge state, and building a good quality web-based task-specific speech synthesis component. In summary, the question is taken and recognized, and then the data is extracted from the pages which is parsed and reformed to be checked easily. Thereafter, the most suitable answer is picked, formed to the language model and synthesized to speech. According to this model, the question answering process consists of five stages:

- Question parsing: labels the question with tags
- Question classification: categorizes the question into five classes:
  (i) Generic Information Request
  (ii) Problem Reporting
  (iii) Factoid Questions
  (iv) Transaction Request
  (v) Information Search
- Information retrieval: takes a query as input and returns a list of answer candidates
- Answer extraction: extracts an answer by checking the ranked list of answer candidates. It selects those candidates with scores higher than a threshold. If none of them meets these conditions, the system returns an empty answer.

They suggest that the system can give better results by improving the query formulation module and the answer extraction module by using statistical approaches and observations. For instance, some words are more likely than others to co-occur in a pair of question and answer, like the word "discount" rather than the word "give". They also focus on the fact that there exists a lexical and stylistic gap between questions

and answers. For instance, a question containing the phrase “travel” is likely to be paired with an answer containing words like “airline”, “flight” and “reservation”.

They used the perceptron algorithm in the learning procedure to estimate the lexical relevance between questions and answers. The dataset consists of 52449 FAQ-answers pairs which were crawled from a variety of business websites. WebTalk achieved 54 % answer accuracy with the learned weights in an experiment. In another one, the answer accuracy increased to 68 %, a 27 % rise from the baseline [1]. That is to say, the query expansion feature makes a big contribution.

In another work of Junlan Feng, et al. [5], WebTalk is instantiated on a telecom company website. They crafted 30 scenarios, of which 24 scenarios are in-domain requests and 6 scenarios are out-of-domain. The scenarios are phrased as broad as possible so that the evaluators can express the requests in their own words. Scenarios are randomly assigned to the evaluators. 100 calls were received from 16 volunteered callers who were presented a web interface for the scenarios, call instructions and survey questions. After each call, each user is asked to fill in a survey of 6 questions related to the success of the dialog.

The results shown in the below tables denote that users were able to successfully obtain the information they requested in 49 % of the dialogs for in-domain requests. This number was zero for out-of-domain requests indicating that all users were unable to find information that the system never had in the first place. The system’s ability to converse with users in a sensible manner when it failed to respond with the exact answer is scored an average of 2.8. Subjects were generally not satisfied that the system “understood” them, giving an average score of 2.4 but that number is 2.7 for in-domain scenarios. This may be attributed to lower recognition and understanding accuracy.

Q4 receives the highest average score of 3.9 which indicates that users considered the quality of the language generation and synthesized speech to be good independent of the domain of the scenarios. Q5 is related to the ease-of-use of the system and receives the lowest average of 2.1 which is certainly related to the low performance

of Q3. Finally, in terms of overall rating (Q6), this system scored an average of 2.7 for in-domain scenarios which incidentally compares favorably with the 3.2 that was obtained for the W99 spoken dialog system [5].

Table 1.1. Possible questions that can be answered inside and outside of the domain.

In-Domain:	You would like to know what type of hardware or equipment you would need in order to access the phone service.
Out-Domain:	You are taking a trip to Florida this Thursday, and you want to check out the weather there.

Table 1.2. Survey questions for the evaluation of the system.

Q1: Did you get the information you requested successfully?
Q2: When the system was unable to give you the information you wanted, were its responses sensible?
Q3: In this conversation, did the system understand what you said?
Q4: In this conversation, did you understand what the system
Q5: In this conversation, was it easy to find the information you wanted?
Q6: In this conversation, how would you rate your overall impression and interaction with the system?

1.3. Contribution of the Thesis

In this thesis, the main focal point is to find a reasonable page or pick the most sensible one in response to the user's question from a web site. That is to say, to save the user from searching the web site for specific information, she will ask the question and is supposed to go to the page containing the answer.

In order to accomplish this object, a number of tasks are performed. First, the question is acoustically recognized to understand the user's concerns. So, mainly in the first step, the question that would be tried to be answered is determined.

Table 1.3. Evaluation results.

	In- domain	Out-of-domain	Total
No of dialogs	79	21	100
Q1 (% of yes)	49 %	0 %	37 %
Q2	2.9	2.3	2.8
Q3	2.7	1.5	2.4
Q4	3.9	3.8	3.9
Q5	2.4	1.2	2.1
Q6	2.7	1.7	2.5

Then, the first thing to do after the acoustic recognition of the question, is to discover what it is about. In this way, after the determination of the question; the subject of the question is also determined and the subject is labelled to the question with a keyword.

Third, the corresponding page has to be found. The answer is thought to exist in the page which is the most relevant with the question. For instance, if a user asks something about an instructor, then the answer is possibly on the instructor's page. The page of something is within reason the one which contains the relevant word most. If a page includes the word "Ahmet" five times, where another does only one time; one assumes naturally that Ahmet's page is the former one. In this extend, the website is searched and the page containing the labeled word to the class of the question most is returned assuming it includes the answer of the question.

In summary, mainly, the application provides an interface to the underlying web-site content taking voiced questions and returning candidate answer pages by means of mining the wealth of information on the website. This is fulfilled thanks to acoustic recognition, classification and search tasks. In order to perform searching, the web site should be crawled and all the target links should be listed previously. After the search results, they are indexed and by this way, the results are returned faster.

1.4. Thesis Outline

In Chapter 1, why an automatic question handling algorithm is important is explained and briefly the problem is stated. After the explanation of the importance of the issue and the problem description, the previous work about the same subject where various researches and their advantages/disadvantages are told and the contribution of the thesis are mentioned.

In Chapter 2, the background of the system architecture is described. The underlying technologies, challenges and efforts, differences from similar systems of each part are discussed. The main principles are told about how the general mentality is built and which operations are realized accordingly. Automatic speech recognition technologies and sample applications; web site search with text and voice; classification methods; and information retrieval techniques are explained respectively.

In Chapter 3, the components of the overall system and the modeling of the process are clarified. So, in this chapter, it is possible to see which processes are applied in order to perform the aim of the work. The recognizer, classifier and information retrieval units are discussed.

The experimental results are given in Chapter 4. And finally, the observations are shared and the possible additions or points to be strengthened are discussed in Chapter 5.

2. BACKGROUND

To go deeper in the underlying technologies, every module has to be dealt with in detail. The fundamental divisions of the system are Automatic Speech Recognition, Search over Web, Classification and Information Retrieval. Each division utilizes distinct algorithms or the general philosophy of the subject.

2.1. Automatic Speech Recognition

2.1.1. General ASR Applications

Spoken language communication is of great prominence because we use this kind of communication every day. In computer world, things get easier with automation rather than commanding everything by hand. Therefore, speech recognition is commonly used and being tried to embed to various applications [6].

However, recognition rates of distant talking speech recognition applications substantially decrease if the acoustic environment contains reverberation. Although standard approaches for compensating such distortions, e.g. cepstral mean subtraction (CMS), are quite effective, they are not appropriate for dynamic human machine interaction [7].

Automatic speech recognition (ASR) for HMI applications often needs to deal with speech recorded by distant microphones. In contrast to the (rather intrusive, hence inappropriate) use of head-sets, in this setting the quality of the recorded utterances is limited. Reasons for this are additionally recorded noise and distortions of the speech signal caused by the recording environment (reverberations).

Additionally, there is another issue, which is about getting rid of most prominent distortions introduced by realistic acoustic environments, namely, noise and reverberation. All known techniques, such as spectral subtraction or multi-stage linear predic-

tion, are designed to suppress only one of these two distortions. Other solutions must be investigated for the combination of a particle filter and multistage linear prediction that aims to simultaneously suppress both distortions. In short, noise and reverberation are traditionally seen as separate problems. But both are invariably present in any distant speech recognition scenario involving a realistic acoustic environment. An effective solution must consider both, not one in isolation [8].

2.1.2. SR over Web

After the general milestones of the speech recognition process and applications and some complications that should be overcome in this field, we focus on the speech processing applications over the web.

In an environment in which World Wide Web has evolved into a central data center where people find what they look for in various forms such as text, speech, video and graphics; communication is also changing from voice over traditional landline phones to multimedia data across multiple mobile devices, services, and networks. Many applications arose that get data over the World Wide Web utilizing data mining techniques by speech recognition. The simplicity of the interface is an important factor for its popularity in addition to the functionality.

As the usage of web continues to grow, the need for accurate systems to extract information will continue to rise. We can talk about three types of classes of systems extracting information [9].

The first class includes systems capable of searching through documents using keywords. These systems, more commonly known as search engines, such as Google Search and Yahoo Search, indexes words and phrases to enable rapid retrieval of documents. Search engines have also been applied for voice search at a smaller scale. These systems support voice or multimodal commands (e.g. voice and mouse clicks) and interface with web contents to retrieve text and images.

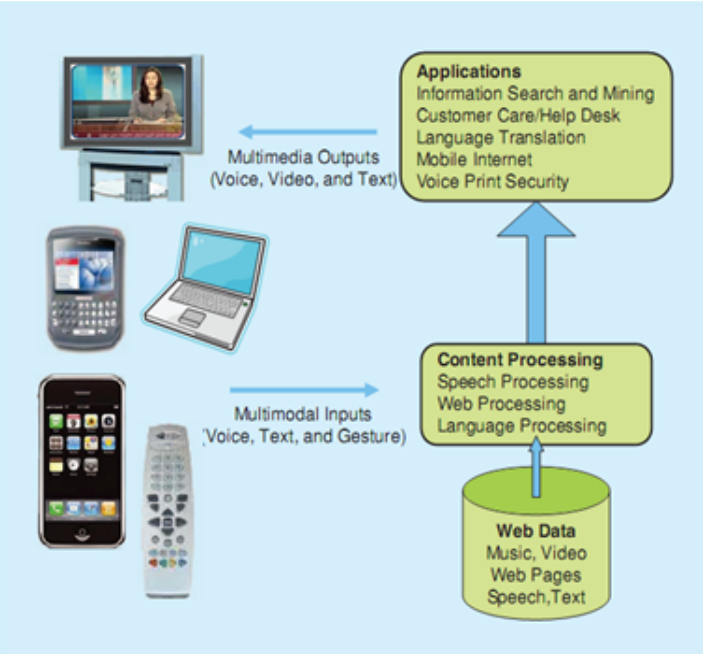

Figure 2.1. Data mining from web with speech using a variety of access devices [9].

The second class of search systems supports interactive question answering. Users can ask a question as opposed to keywords. This is especially important in mobile devices since it is not very comfortable to work with vocal commanding rather than text. The system parses the request and retrieves direct answers rather than document links, hence rescuing the user from manual search. In this way, a Help Desk capability can be provided on websites by companies using "virtual agents" that interact with users about company products and services. Interactive question/answering systems that mine the web and use voice interfaces are in their infancy.

The third class of systems is applied to information mining. This class of systems involve extracting key elements from large unstructured data, discovering information, and summarizing relationships and trends. Information mining systems can infer knowledge and trends automatically from data, analyze and report application performance, and adapt and improve over time with minimal or zero human involvement. For example, monitoring voice recordings, videos, or e-mails to discover patterns, capture useful trends, and generate alarms is essential for enhancing a variety of applications including call centers operation and Internet security. These systems are also useful

for analyzing, monitoring, and tracking customer preferences and interactions to better establish customized sales and technical support strategies.

2.1.3. Recognition Network

There are various tools that can be used to perform the recognition like HTK [10] and Sphinx [11]. These packages perform the recognition according to the path processing algorithms like Viterbi. However, at first, the base model must be constructed; that is to say, the recognition forms and patterns have to be stated.

A network describes the sequence of words that can be recognised and, for the case of sub-word systems, a dictionary describes the sequence of HMMs that constitute each word. A word level network will typically represent either a Task Grammar which defines all of the legal word sequences explicitly or a Word Loop which simply puts all words of the vocabulary in a loop and therefore allows any word to follow any other word. Word-loop networks are often augmented by a stochastic language model. Networks can also be used to define phone recognisers and various types of word-spotting systems [12].

The formation rules and statements about the architecture constitute the "grammar" of the model. Models without grammar are also usable, however, if the sentences are certain and it is not a drawback to limit the formation; using a grammar is advantageous in terms of recognition performance and sentence accuracy.

Continuous speech recognition requires some language information to restrict the recognition. Simple dictionary based recognition is rarely enough for achieving acceptable recognition accuracy. One of the simplest forms of additional language restriction is a task grammar. It defines the allowed word sequences or sentences as word networks or finite state machines.

A speech recognition grammar is the collection of word patterns and the recognition system expects users to construct sentences according to this set [13]. For instance,

if you call a voice directory application, it will prompt you for the name of the person you would like to talk with. It will then start up a speech recognizer, giving it a speech recognition grammar. This grammar contains the names of the people in the directory, and the various sentence patterns callers typically respond with.

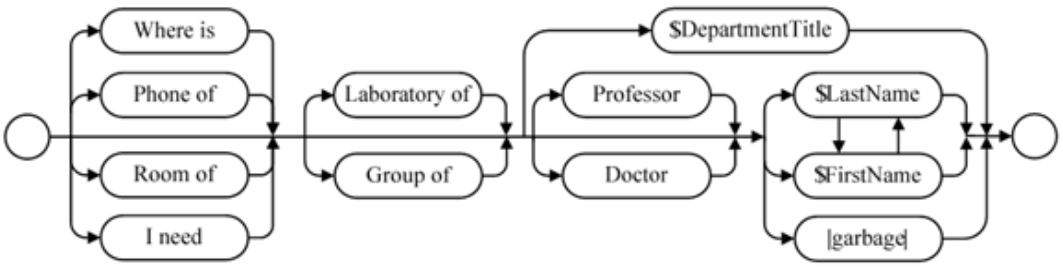

Figure 2.2. Sample grammar.

2.2. Search

2.2.1. Site Search

A web search engine is a text-based information retrieval system that mainly shows or brings you which information you want. For a work in which the search engine is limited to a specific site; a site search engine is a better description rather than a web search engine. However, when the general concept is taken into consideration, a web search engine indexes the web and responds to keyword queries with a ranked list of web pages in a basic manner.

A typical web search engine has four components: crawling, indexing, querying, and ranking. A web crawler collects targeted content. An indexer indexes the content of each collected item and keeps the generated indices. Ranking compares the query against the stored indices, retrieves a list of documents considered relevant to the query, and ranks the documents based on query relevancy scores.

Table 2.1. Basic components of a web search engine.

Crawler

(i) Start with a list of URLs, called seeds.

(ii) Retrieve web pages from the URL list, store pages on local disks, and extract hyperlinks from these pages.

(iii) Update the URL list with the hyperlinks that have not been visited.

(iv) Go back to Step (ii).

Indexer

(i) Tokenize documents to sequences of terms and remove stop words such as "a" and "the."

(ii) Build inverted index, a data structure storing a mapping from terms (t) to document identifications (d_j), term frequency (tf), inverse document frequency (idf), and positions of ti in d_j. tf(t_i,d_j) is the number of times t_i appears in d_j. idf is computed as idf(t_i) = log (n/d_{fi}), where d_{fi} is the number of documents that contain t_i.

Ranker

(i) Each document (d_j) and query (q) is represented as a vector in a n-dimensional euclidian space: dj = (w_{1j}, —w_{ii}, —w_{nj}), where n is the number of terms in the document collections, w_{ij} = tf(t_i,d_j)idf(t_i) and w_{iq} = tf(t_i,q)idf(t_i)

(ii) Relevancy between q and d_j is measured as the cosine of the vectors tf-idf is used to populate the vector :

$$cos(q, d_j) = \frac{\sum_i w_{ij} w_{iq}}{\sqrt{\sum_i {w_{ij}}^2} \sqrt{\sum_i {w_{iq}}^2}} \tag{2.1}$$

There are a number of features and methods that are used in searching and naturally affect the success and performance. Besides the fact that term frequency–inverse document frequency (TF-IDF) is the basic mechanism used for ranking; a much richer set of features such as linking properties of the web, user preference, and date and time of documents can also be utilized in a reasonable scale [9].

A number of natural language processing techniques have been explored in document search including stemming, synonyms, and machine translation. Stemming refers to the process of mapping a word to its root form when tokenizing documents and queries. The motivation is that a user searching for a keyword, such as "meetings," is also interested in documents containing the word "Meeting." Another advantage of stemming is reducing the language complexity. The number of distinct terms is dramatically reduced after stemming. Synonym search brings the search one level closer to semantic search. For instance, a user can type "∼meeting" to find pages containing "meeting", "conference" and "netmeeting."

The more information a web site has, the more useful it is and the harder to navigate. No matter how well the designers form the site navigation elements, visitors will need other ways to find what they are looking for. Site search tools provide a powerful and familiar means to provide that access. Visitors can just type the words and press the Search button in a form, and get a list of all the documents that match those words on the site.

The search engine is the application which searches the data and returns the results to the client. This usually means creating an HTML page in the specified format. Most search engines search within an index, created by an Indexer application. A few just search the files in real-time, but that can get very slow. To send a search to the search engine, most systems include forms. The site visitor enters their search terms in a text field, and may select appropriate settings in the form. When they click the Submit button, the server passes that data to the search engine application.

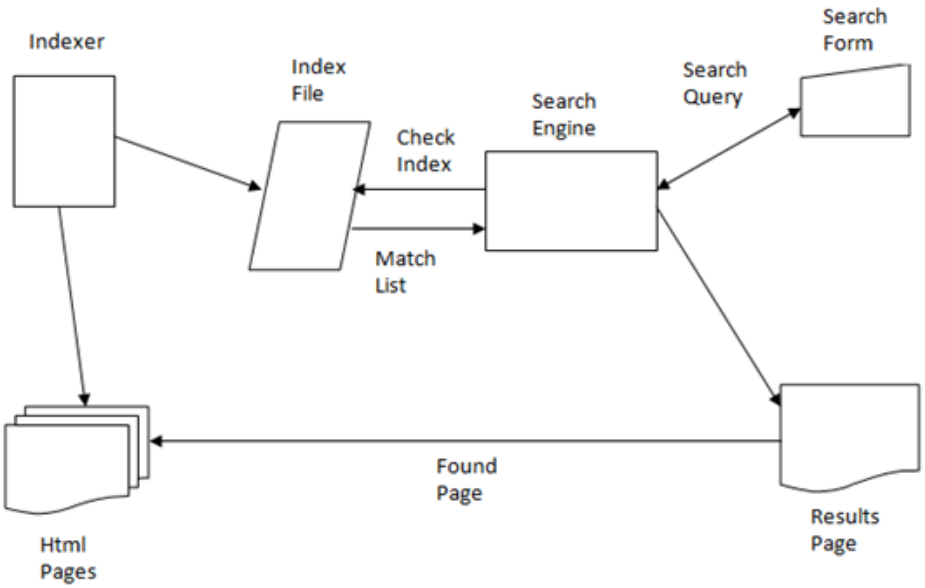

Figure 2.3. General architecture of a search application [14].

2.2.2. Voice Search

Web search is widely used by people for various aims; however, the increasing availability of multimedia data, broadband access, voice over IP, and powerful mobile devices is fostering a new wave of human–computer interfaces that support multiple modalities. Those so-called multimodal user interfaces combine interactive multimedia output where audio, video, graphics, and text are accurately synchronized under time constraints, with several modes of input such as speech, keyboard, pointing devices, handwriting, and gestures.

Multimodal search uses the Web as a source for information retrieval of maps, business listings, investment information, travel reports, etc. The integration of speech, text, and visual interfaces with Web contents provide richer, more versatile, and robust user interfaces than traditional speech input or text input alone. Voice search of directory assistance applications, for example, which has been traditionally limited to voice input, are ideal candidates for multimodal user interfaces.

With the use of mobile devices, a person can invoke a business listing using a voice command, select the appropriate output using touch (or click), and either make the call or retrieve additional information including a business resume, directions, and reviews acquired from the Web. Multimodal search plays an important role in entertainment. In IPTV, for example, the traditional remote control will soon become obsolete. Using up/down and left/right keys is insufficient when navigating through a large number of channels, programs, movies, and music, all changing continuously every day. With multimodal search, users will have the ability to apply speech to identify the desired program or narrow down the search into a short menu and then use up/down and left/right keys to select the program of choice. One sample application for the mobile devices is the "Location Grounding in Multimodal Local Search" application [15] developed by AT&T.

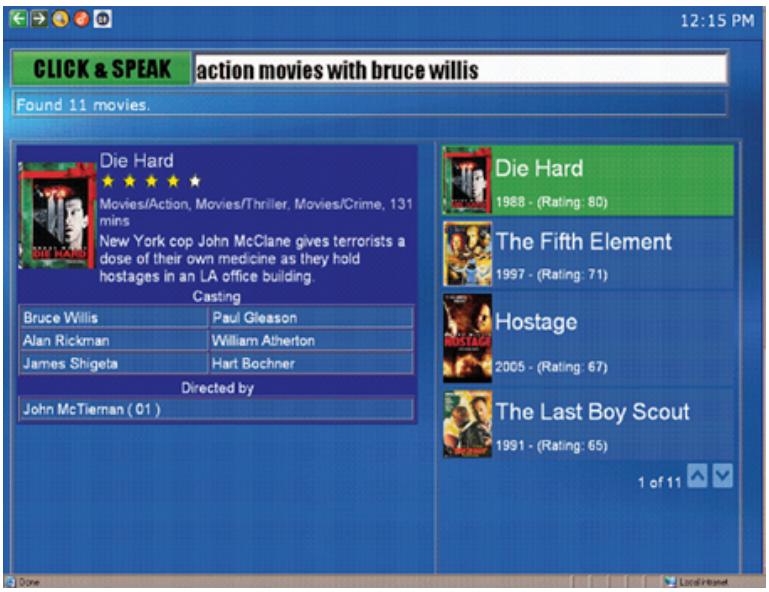

Figure 2.4. An application of voice search for IPTV movie-on-demand [9].

The ease-of-use of multimodal search over standard clicking using a remote control will totally change the way people watch television. It should be noted that much of the progress made in voice and multimodal search is for vertical industries with application-specific language models. There are numerous technical problems that need to be solved before users are able to break down the application barrier and be

able to do mobile search freely for unlimited information access. New research in the areas of personalization and customization will be needed as people become accustomed to their devices and become power users of the technology.

Voice search is essentially the combination of automatic speech recognition (ASR) and document search. User says something in order to make a search, rather than writing it. In other words, a system takes a spoken query as input, formulates a text query, queries the index, and provides users the search results. ASR is a task of finding the most likely set of words for a given acoustic signal by means of searching through all possible recognition choices using a Viterbi decoding algorithm. It takes place commonly especially in mobile services since the text processing is not comfortable enough due to screen restrictions and similar problems. However, it faces several technological and user interface challenges [9].

First, as in other general speech recognition applications, noise is the biggest factor resulting in poor ASR performance in mobile environments. The varying acoustic environment, background noise, and transmission channel noise caused by the speech codec and transmission errors significantly degrade speech recognition performance.

Second, creating word pronunciation, text normalization, and language models to represent how people speak search queries is important to achieve high ASR accuracy. This is particularly problematic for mobile search where the huge vocabulary size and Web search terms result in language models that have very high perplexity. Although there are plenty of Web search queries that are available for modeling, the data are not representative of the way people use voice search.

In addition, the integration between speech recognition and web search is another key factor in the accuracy. While speech recognition aims to minimize the expected word error rate, Web search identifies, retrieves, and ranks Web documents that are most relevant to the search query. Frequent words that are unimportant for search, such as “the,” often have low word error rates, while important yet infrequent search terms such as “camera” end up having high recognition error rates.

Finally, for the sake of usability, an appropriate visual interface for taking the input query and presenting search results can be regarded as the visual success of the application. The challenge is not only in presenting the information to the user but also in optimizing the screen for maximum output display. This optimization may be required for every mobile device due to the limited screen availability.

2.3. Classification

2.3.1. Description

When the fact that knowledge and information are kept in different types of documents is considered, the importance of the idea about where to locate them can easily be realized. In this point of view, the objective of document classification is to reduce the detail and diversity of data and the resulting information overload by grouping similar documents together. The main classification objective, particularly with respect to knowledge management, is to simplify access to and processing of explicit knowledge. Classification supports analyzing the knowledge and, thus, can ease the processes in the areas of:

(i) Retrieval
(ii) Organization
(iii) Visualization
(iv) Development
(v) Exchange of knowledge

The use of identified class representatives reduces the amount of knowledge to cope with. Document categorization may be viewed as assigning documents or parts of documents in a predefined set of categories. Usually this set is created once and for all with so called training documents and, thus, remains unchanged over time.

To apply the processing algorithms for document classification the words of the documents have to be converted into numerical representations. This step is called doc-

ument preprocessing and subsumes feature extraction, feature selection, and document representation as activities [16].

2.3.1.1. Feature Extraction. Feature extraction is the first step in document preprocessing. The general problem in this phase is to generate a list of terms that describes the documents sufficiently. Therefore the training documents are parsed to determine a list of all words (i.e. features) contained in the documents. Afterwards feature reducing techniques are applied to reduce the dimension of the list created by the parsing process; this list is often denoted as dictionary. The most popular methods for this purpose are stop word removal and word stemming. The main goal of stop word removal is to purge the dictionary from "noise" (e.g. articles, prepositions and numbers). It is usually realized by comparing the dictionary entries with a predefined stop word list and then eliminating the accordant ones. Word stemming tries to treat terms that differ only in the affix (suffix or prefix), i.e. words with the same stem, as one single feature. Commonly applied word stemming techniques are affix removal, successor variety, and n-grams.

2.3.1.2. Feature Selection. After the features are extracted, the most useful of them must be selected. That is to say, the main objective of this phase is to eliminate those features that provide only few or less important information. This time statistical values are used to determine the most meaningful features. The most common indicators are term frequency (TF), inverse document frequency (IDF), and their multiplicative combination (TFIDF). By using TF it is assumed that important words occur more often in a document than unimportant ones. When applying IDF, the rarest words in the document collection are supposed to have the biggest explanatory power. With the combined procedure TFIDF the two measures are aggregated into one variable. Whatever metric is used, at the end of the selection, only the top n words with the highest score are selected as features.

2.3.1.3. Document Representation. Document representation is the final task in document preprocessing. Here the documents are represented in terms of those features to

which the dictionary was reduced in the precedent steps. Thus, the representation of a document is a feature vector of n elements where n is the number of features remaining after finishing the selection process. The whole document collection can therefore be seen as an MxN–feature matrix A (with M as the number of documents) where the element a (i, j) represents the frequency of occurrence of feature j in document i. Typical frequency measures are the above mentioned values TF, IDF, and TF-idf; all positive values may be replaced by 1, leading to a binary representation which indicates whether or not a certain feature appears in the document.

2.3.2. Approaches

Major categorization approaches are decision trees, decision rules, k-nearest neighbors, Bayesian approaches, neural networks, regression-based methods, and vector-based methods [16].

2.3.2.1. Decision Trees. Decision trees are a powerful classification technique. Two widely known algorithms for building decision trees are Classification and Regression Trees and C4.5 [16]. The tree tries to infer a split of the training data based on the values of the available features to produce a good generalization. The split at each node is based on the feature that gives the maximum information gain. Each leaf node corresponds to a class label. A new example is classified by following a path from the root node to a leaf node, where at each node a test is performed on some feature of that example. The leaf node reached is considered the class label for that example. Classification using a decision tree is performed by routing from the root node until arriving at a leaf node [17]. The algorithm can naturally handle binary or multiclass classfication problems. The leaf nodes can refer to either of the K classes concerned [18].

Decision tree methods rebuild the manual categorization of the training documents by constructing well-defined true/false-queries in the form of a tree structure where the nodes represent questions and leafs represent the corresponding category of

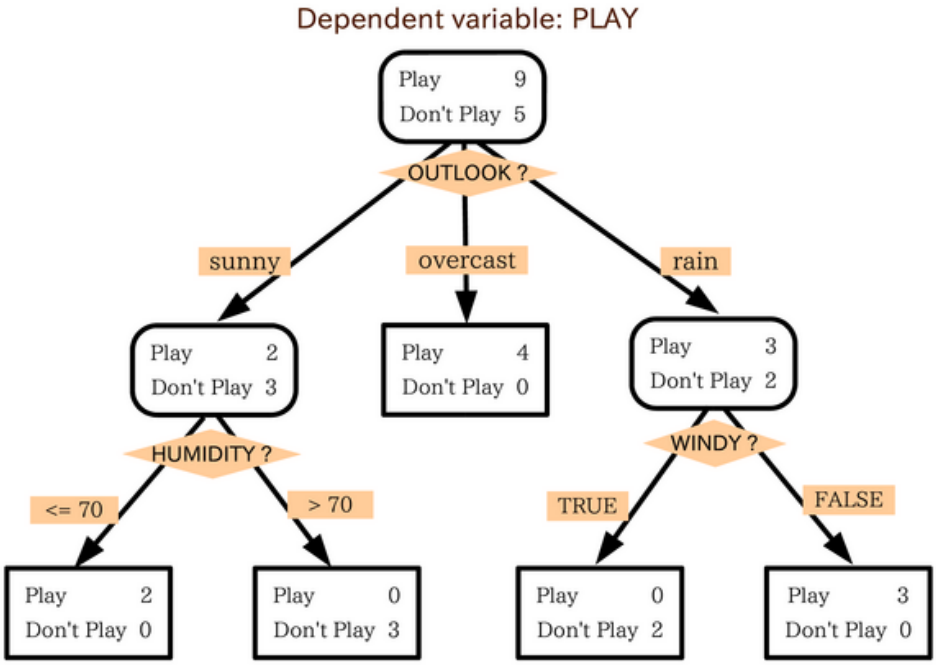

Figure 2.5. An example to Decision Tree Algorithm [17].

documents. After having created the tree, a new document can easily be categorized by putting it in the root node of the tree and let it run through the query structure until it reaches a certain leaf.

The main advantage of decision trees is the fact that the output tree is easy to interpret even for persons who are not familiar with the details of the model. The tree structure generated by the model provides the user with a consolidated view of the categorization logic and is therefore useful information.

A risk of the application of tree methods is known as “overfitting”: A tree overfits the training data if there exists an alternative tree that categorizes the training data worse but would categorize the documents to be categorized later better. This circumstance is the result of the algorithm’s intention to construct a tree that categorizes every training document correctly; however, this tree may not be necessarily well suited for other documents. This problem is typically moderated by using a validation data set for which the tree has to perform in a similar way as on the set of training data.

Other techniques to prevent the algorithm from building huge trees (that anyway only map the training data correctly) are to set parameters like the maximum depth of the tree or the minimum number of observations in a leaf. If this is done, Decision Trees show very good performance even for categorization problems with a very large number of entries in the dictionary.

2.3.2.2. Decision Rules. Decision rule algorithms construct for every category a rule set that describes the profile of this category. In general, a single rule consists of a category name and a feature of the dictionary which is typical for the training documents belonging to the considered category. Then the rule set is created by combining the separate rules with the logical operator "or". Usually not all of the rules are required to categorize the documents adequately. Therefore, heuristics are applied to reduce the size of the rule sets. The goal is to achieve a reduced rule set per category which, however, does not affect the categorization of the training documents.

The main advantage of decision rules is the possibility to create local dictionaries (i.e. per category) during the feature extraction phase. Consider the homonym "bark", describing as well the outer part of a tree and the sounds of a dog. In a universal dictionary "bark" will be listed only once, thus documents with "the bark" and documents with "to bark" get a similar feature vector even they have nothing in common. Local dictionaries are -to a certain extent- able to distinguish different meanings of homonyms. If in the precedent example dog-related documents belong to a different category than tree-related do, the local dictionaries of these categories both list the term "bark" but with different meaning, because once it appears in a rule set with "dog" and another time with "tree".

A disadvantage is that it is impossible to assign a document exclusively to one category because rules from different categories are applicable. In this case the document is usually assigned to several categories.

2.3.2.3. Information Gain. Maximum entropy is the most common concept for information gain applications which can be defined as a probability distribution estimation technique widely used for a variety of natural language tasks, such as language modeling, part-of-speech tagging, and text segmentation [19]. The underlying principle of maximum entropy is that without external knowledge, one should prefer distributions that are uniform. Constraints on the distribution, derived from labeled training data, inform the technique where to be minimally non-uniform. The maximum entropy formulation has a unique solution which can be found by the improved iterative scaling algorithm.

Maximum entropy is a general technique for estimating probability distributions from data. The overriding principle in maximum entropy is that when nothing is known, the distribution should be as uniform as possible, that is, have maximal entropy. Labeled training data is used to derive a set of constraints for the model that characterize the class-specific expectations for the distribution. Constraints are represented as expected values of features, any real-valued function of an example. The improved iterative scaling algorithm finds the maximum entropy distribution that is consistent with the given constraints.

A document is represented by a set of word count features. The labeled training data is used to estimate the expected value of these word counts on a class-by-class basis. Improved iterative scaling finds a text classifier of an exponential form that is consistent with the constraints from the labeled data.

To illustrate the methods, let's look for an example: Consider we have four classes have unique information about the categorization that that on average 40 % of documents with the word "professor" in them are in the faculty class. Then, we can say that a document with "professor" in it has a 40 % chance of being a faculty document, and a 20 % chance for each of the other three classes. On the other hand, if a document does not have "professor" we would guess the uniform class distribution, 25 % each. In short, always it has a tendency to distribute the probabilities uniformly, except the constraints learnt from the training data.

2.3.2.4. k-Nearest Neighbor. In k-Nearest Neighbor method, to classify an unknown example, the distance (using some distance measure e.g. Euclidean) from that example to every other training example is measured. The k smallest distances are identified, and the most represented class in these k classes is considered the output class label. The value of k is normally determined using a validation set or using cross-validation [18].

Whereas the other categorization methods are based on a learning phase in which the guiding principles to be used are determined and consist therefore of at least two phases, k-nearest neighbor completely skips the learning phase and categorizes on-the-fly.

The categorization itself is usually performed by comparing the category frequencies of the k nearest documents (neighbors). How close documents are to each other can be evaluated by measuring for instance the angle between the two feature vectors or calculating the Euclidean distance between the vectors. In the latter case the feature vectors have to be normalized to length 1 to take into account that the size of the documents (and, thus, the length of the feature vectors) may differ.

A doubtless advantage of the k -nearest neighbor method is its simplicity. It has reasonable similarity measures and does not need any resources for training. K-nearest neighbor performs well even if the category-specific documents form more than one cluster because the category contains, e.g. more than one topic. This situation is badly suited for most categorization algorithms.

A disadvantage is the above-average categorization time because no preliminary investment (in the sense of a learning phase) has been done. Furthermore, with different numbers of training documents per category the risk increases that too many documents from a comparatively large category appear under the k nearest neighbors and thus lead to an inadequate categorization.

2.3.2.5. Bayesian Approaches. A Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem (from Bayesian statistics) with strong (naive) independence assumptions and is particularly suited when the dimensionality of the inputs is high [20].

There are two groups of Bayesian approaches in document categorization: Naïve and non-naïve Bayesian approaches. The naïve part of the former is the assumption of word (i.e. feature) independence, meaning that the word order is irrelevant and consequently that the presence of one word does not affect the presence or absence of another one. This assumption makes the computation of Bayesian approaches more efficient. But although the assumption is obviously severely violated in every language, it has been shown that the classification accuracy is not seriously affected by this kind of violations.

Nevertheless, several non-naïve Bayesian approaches eliminate this assumption. Naïve Bayesian approaches have been developed comparatively early and have been studied frequently in data mining before the topic of document categorization gained importance. They perform as well as newer, more sophisticated methods and also show a very good runtime -behavior during the categorization of new documents.

A disadvantage of Bayesian approaches in general is that they can only process binary feature vectors and, thus, have to abandon possibly relevant information.

2.3.2.6. Neural Networks. Different neural network approaches have been applied to document categorization problems. While some of them use the simplest form of neural networks, known as perceptrons, which consist only of an input and an output layer, others build more sophisticated neural networks with a hidden layer between the two others. In general, these feed-forward -nets consist of at least three layers (one input, one output, and at least one hidden layer) and use backpropagation as learning mechanism. However, the comparatively old perceptron approaches perform surprisingly well.

The advantage of neural networks is that they can handle noisy or contradictory data very well. Furthermore some types of neural networks are able to comprehend fuzzy logic, but one has to change from backpropagation as learning mechanism to counterpropagation (for which worse categorization results are reported).

The advantage of the high flexibility of neural networks entails the disadvantage of very high computing costs. Another disadvantage is that neural networks are extremely difficult to understand for an average user; this may negatively influence the acceptance of these methods.

2.3.2.7. Regression-based Methods. For this method the training data are represented as a pair of input/output matrices where the input matrix is identical to our feature matrix A and the output matrix B consists of flags indicating the category membership of the corresponding document in matrix A. Thus B has the same number of rows like A (namely M) and c columns where c represents the total number of categories defined. The goal of the method is to find a matrix F that transforms A into B' (by simply computing $B'=A*F$) so that B' matches B as well as possible. The matrix F is determined by applying multivariate regression techniques. An advantage of this method is that morphological preprocessing (e.g. word stemming) of the documents can be avoided without losing categorization quality. Thus, regression-based approaches become truly language-independent. Another advantage is that these methods can easily be used for both single-category and multiple-category problems.

Unfortunately regression-based methods are not very popular in the categorization community and, therefore, investigations comparing regression-based methods with others are relatively rare.

2.3.2.8. Vector-based Methods. One of the simplest categorization methods is the centroid algorithm. During the learning stage only the average feature vector for each category is calculated and set as centroid-vector for the category. A new document is easily categorized by finding the centroid-vector closest to its feature vector. Unless

the document clusters overlap each other, this method does not need many training documents.

If, however, the document clusters overlap each other or the category consists of two or more different topics (clusters), the algorithm performs often poor. The method is also inappropriate if the number of categories is very large.

Support vector machines (SVM) need in addition to positive training documents also a certain number of negative training documents which are untypical for the category considered. SVM is then looking for the decision surface that best separates the positive from the negative examples in the n -dimensional space. The document representatives closest to the decision surface are called support vectors. The result of the algorithm remains unchanged if documents that do not belong to the support vectors are removed from the set of training data.

An advantage of SVM is its superior runtime-behavior during the categorization of new documents because only one dot product per new document has to be computed.

A disadvantage is the fact that a document could be assigned to several categories because the similarity is typically calculated individually for each category. Nevertheless, SVM is a very powerful method and has outperformed other methods in several studies.

2.4. Information Retrieval

2.4.1. Description

Information Retrieval (IR) is the process of retrieving information about a subject from a collection of data objects. Information Retrieval tries to satisfy what a user need which distinguishes it from Data Retrieval, which determines which documents of a collection contain the keywords of a user query in the context of documents [21].

The IR system extracts the contents of the information items in a collection and ranks them. Ranking is performed according to a degree of relevance to the user query. This extraction of document content focuses on syntactic and semantic information from the document text [22].

After the invention of the World Wide Web, the unique characteristics of web made the previous Information Retrieval techniques unsuitable or insufficient. Previous IR algorithms were developed for small data such as newspaper articles or book catalogs in a library. However; web is massive, changes more rapidly and is spread over geographically distributed computers [23].

One of the main challenges the open Web poses to search engines is "search engine spamming", i.e. malicious attempts to get an undeserved high ranking in the results. This has created a whole branch of Information Retrieval called "adversarial IR", which is related to retrieving information from collections in which a subset of the collection has been manipulated to infiuence the algorithms.

A solution to this problem is to use the hypertext structure of the Web, using links between pages as citations are used in academic literature to find the most important papers in an area. Link analysis, which is often not possible in traditional information repositories but is quite natural on the Web, can be used to exploit links and extract useful information from them.

The importance of the issue is mentioned by Lawrence Page, et al. [21] as: "Unlike academic papers which are scrupulously reviewed, web pages proliferate free of quality control or publishing costs. With a simple program, huge numbers of pages can be created easily, artificially infiating citation counts. Because the Web environment contains profit seeking ventures, attention getting strategies evolve in response to search engine algorithms. For this reason, any evaluation strategy which counts replicable features of web pages is prone to manipulation."

2.4.2. History

Until today, the field of Information Retrieval (IR) has matured considerably, since it was born in the 1950's out of this necessity. Several IR systems are used on an everyday basis by a wide variety of users.

The 1970's and 1980's saw many developments built on the advances of the 1960s. New models/techniques were experimentally proven to be effective on small text collections (several thousand articles) available to researchers at the time. However, due to lack of availability of large text collections, the question whether these models and techniques would scale to larger corpora remained unanswered. This changed in 1992 with the inception of Text Retrieval Conference, or TREC1.

TREC is a series of evaluation conferences sponsored by various US Government agencies under the auspices of NIST, which aims at encouraging research in IR from large text collections. With large text collections available under TREC, many old techniques were modified, and many new techniques were developed (and are still being developed) to do effective retrieval over large collections. TREC has also branched IR into related but important fields like retrieval of spoken information, non-English language retrieval, information filtering, user interactions with a retrieval system, and so on. The algorithms developed in IR were the first ones to be employed for searching the World Wide Web from 1996 to 1998 [24].

2.4.3. Techniques

IR systems rank documents by their estimation of the usefulness of a document for a user query. Most IR systems assign a numeric score to every document and rank documents by this score. Several models have been proposed for this process. The three most used models in IR research are the vector space model, the probabilistic models, and the inference network model.

In the vector space model text is represented by a vector of terms. The definition

of a term is not inherent in the model, but terms are typically words and phrases. If words are chosen as terms, then every word in the vocabulary becomes an independent dimension in a very high dimensional vector space. Any text can then be represented by a vector in this high dimensional space. If a term belongs to a text, it gets a non-zero value in the text-vector along the dimension corresponding to the term. Since any text contains a limited set of terms (the vocabulary can be millions of terms), most text vectors are very sparse. Since a word exists or not, it means a term is assigned to a positive value or set to zero, that is to say vector based systems operate in the positive quadrant of the vector space.

To assign a numeric score to a document for a query, the model measures the similarity between the query vector (since query is also just text and can be converted into a vector) and the document vector. The similarity between two vectors is once again not inherent in the model. Typically, the angle between two vectors is used as a measure of divergence between the vectors, and cosine of the angle is used as the numeric similarity (since cosine has the nice property that it is 1.0 for identical vectors and zero for orthogonal vectors).

For example, let's check the divergence between the word "cat" and "animal cat" text. The text has a value in the dimension of "animal" word and also a value in the "cat" word dimension. So, the angel between these two vectors is $45\,^{\circ}$. That is to say, the score is $cos(pi/4) = 0.7$. Since the units of the vector components in different dimensions are 0 or 1, the highest score is possible when all the words are identical. As an alternative, the inner-product (or dot-product) between two vectors is often used as a similarity measure. If all the vectors are forced to be unit length, then the cosine of the angle between two vectors is same as their dot-product.

Most IR systems use single words as the terms. However not all the words are vital terms in the meaning and can be ignored such as "the", "in", "of" that are considered as non-informative and also called stop-words. Joining various forms of the same word to its root form, called stemming in IR jargon, is also used by many systems. The main idea behind stemming is that users searching for information on retrieval will

also be interested in articles that have information about retrieve, retrieved, retrieving, retriever, and so on. This also makes the system susceptible to errors due to poor stemming. For example, a user interested in information retrieval might get an article titled Information on Golden Retrievers due to stemming. For document ranking, which differentiates modern systems from primitive ones, the most critical piece of information needed is a term's weight in a document. Various methods for weighting terms have been developed in the field, in which, three main factors come into play in the final term weight formulation [24].

(i) Term Frequency (or tf): Words that repeat multiple times in a document are considered salient. Term weights based on tf have been used in the vector space model since the 1960's.

(ii) Document Frequency: Words that appear in many documents are considered common and are not very indicative of document content. A weighting method based on this, called inverse document frequency (or idf) weighting, was proposed by Sparck-Jones early 1970's.

(iii) Document Length: When collections have documents of varying lengths, longer documents tend to score higher since they contain more words and word repetitions. This effect is usually compensated by normalizing for document lengths in the term weighting method.

Information Extraction Systems are developed in order to cope with the challenges regarding the fact that they should meet some requirements [25] such as:

- Accuracy: The system must extract the correct data.
- Resilience: The system must continue to work properly even when web pages change.
- Repair: The system should automatically repair its extraction rules when a web page changes.
- Generality: Information extraction rules need to work for most web sites in an

application domain.

- Extensibility: Information extraction rules should be easy to build for a variety of domains.
- Independency: The system should allow for platform-independent data exchange.

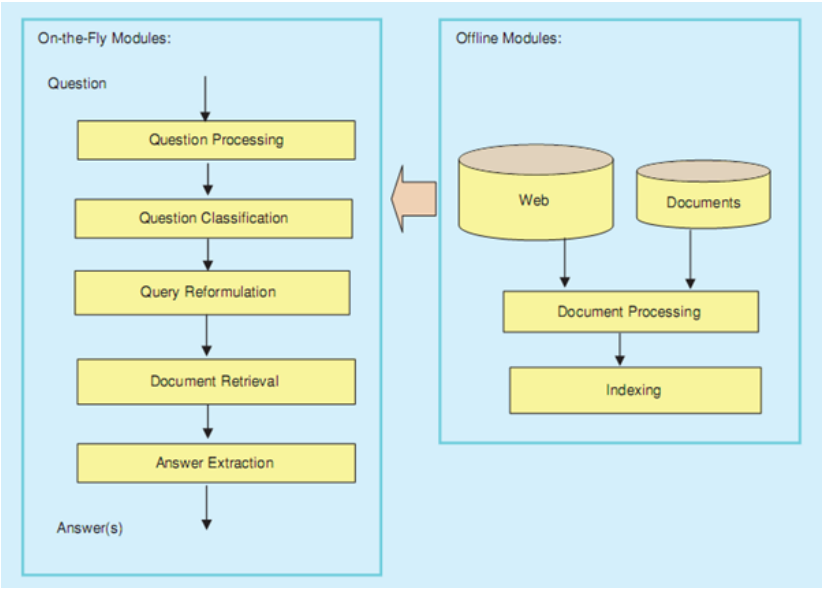

Figure 2.6. Question answering components [9].

2.4.4. Tf-Idf

So, what is tf-idf? The tf–idf weight (term frequency–inverse document frequency) is a weight often used in information retrieval and text mining. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document. Variations of the tf–idf weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query [26].

One of the simplest ranking functions is computed by summing the tf-idf for each query term; many more sophisticated ranking functions are variants of this simple

model.

$$tf_{ij} = \frac{N_{ij}}{\sum_k n_{kj}} \tag{2.2}$$

$$idf_i = log \frac{|D|}{|d : t_i \in d|} \tag{2.3}$$

$$(tf - idf)_{ij} = tf_{ij} * idf_i \tag{2.4}$$

Suppose we have a set of English text documents and wish to determine which document is most relevant to the query "the brown cow". A simple way to start out is by eliminating documents that do not contain all three words "the", "brown" and "cow" but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document and sum them all together; the number of times a term occurs in a document is called its term frequency. However, because the term "the" is so common, this will tend to incorrectly emphasize documents which happen to use the word "the" more frequently, without giving enough weight to the more meaningful terms "brown" and "cow". Also the term "the" is not a good keyword to distinguish relevant and non-relevant documents and terms. On the contrary, the words "brown" and "cow" that occur rarely are good keywords to distinguish relevant documents from the non-relevant documents. Hence an inverse document frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the collection and increases the weight of terms that occur rarely.

To sum up, the main idea is to count the keyword in the target document according to importance of the words. "A" or "the" words are not the vital parts of a statement, while other parts can carry more importance. Therefore, taking those words into consideration in a same weight as the others misleads the results and causes faulty

matching's. Determining every words weight and search in this way prevents this.

The term count in the given document is simply the number of times a given term appears in that document. This count is usually normalized to prevent a bias towards longer documents (which may have a higher term count regardless of the actual importance of that term in the document) to give a measure of the importance of the term within the particular document.

The tf-idf calculations are given in Equation 2.2, 2.3 and 2.4 [27]. Tf-idf weight is calculated as multiplying the term frequency term and inverse document frequency where term frequency is the count in a document over the total word count in that document and inverse document frequency expresses the proportion of the document count involving the keyword to the total document number.

A high weight in tf–idf is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms. The tf-idf value for a term will be greater than zero if and only if the ratio inside the idf's log function is greater than one. Depending on whether one is added to the denominator, a term in all documents will have either a zero or negative idf, and if one is added to the denominator a term that occurs in all but one document will have an idf equal to zero.

Consider a document containing 100 words wherein the word cow appears three times. Following the previously defined formulas, the term frequency (TF) for cow is then

$$3/100 = 0.03. \tag{2.5}$$

Now, assume we have 10 million documents and cow appears in one thousand of these. Then, the inverse document frequency is calculated as

$$log(10^7/1000) = 4. \tag{2.6}$$

The TF-IDF score is the product of these quantities

$$0.03 \times 4 = 0.12. \tag{2.7}$$

That is to say, the occurrence of the keyword in the current document contributes proportionally to the score, where the general frequency of the term in the whole documents decreases the score. In this way, a common word occurs at most of the documents is multiplied with a small value where a rare word is multiplied with a big value. For example if it occurs only in a single one over one million documents; it is multiplied with

$$log(10^7/1) = 7. \tag{2.8}$$

2.4.5. Web Crawling

Crawlers, also called robots and spiders, are programs that browse the World Wide Web autonomously. Web structure mining is the process of mining Web-link structure, web site structure, and web document structure. In Web-link structure mining, the web is viewed as a graph, where nodes represent web pages and edges are the hyperlinks between them. This graph itself is dynamic and constantly changing. Properties of this dynamic graph provide insights into how humans organize information globally, how robust and vulnerable the Internet is, and how the web evolves over time. Properties of the Web structure have been exploited to improve search and navigation [28].

In structure mining, a web site is treated as more than just a collection of web pages. It is a systematic organization of web pages into subdirectories. Related web pages are woven together through meaningful hyperlinks. A corporate web site, for example, is often professionally designed and developed based on a thorough analysis of the market needs, competitive landscape, and functional requirements.

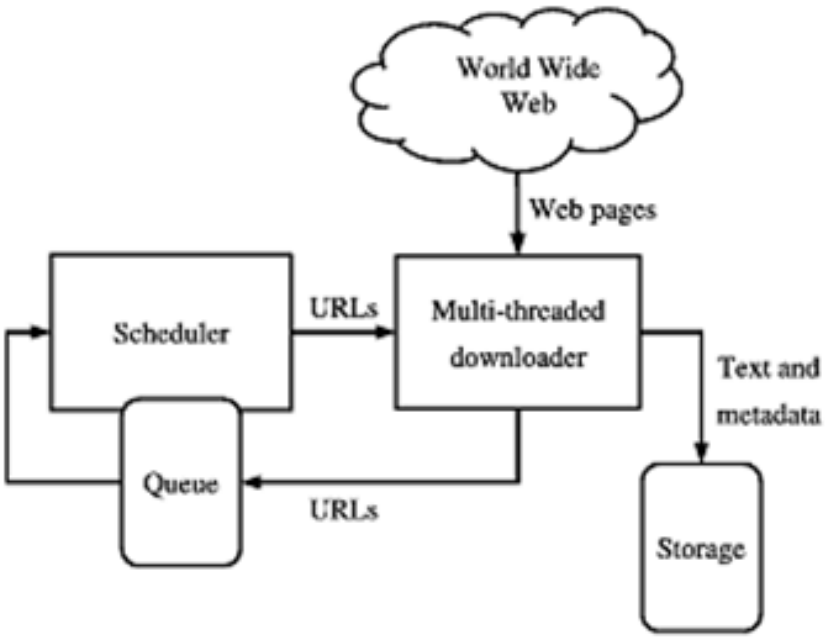

Figure 2.7. Web crawler [21].

The search algorithm mainly, uses hyperlinks (or citations) and given a web site, collects all hyperlinks with semantically meaningful anchor texts and weights each anchor text proportionally to the frequency it is used and the importance of the pages. Summaries are generated when conversing with users.

To reach the desired information, the clues on the web site can be used. For instance, pages on a website are often systematically organized into subdirectories and are linked to each other through meaningful hyperlinks. Most web pages have meaningful page titles. Text data units including information blocks falling into the categories like Table-Data, FAQ-Answer, Bulleted-List, Normal-Content, and Heading-Content, defined in WebTalk, have the potential to be answers for the questions [1].

The typical design of search engines is a "cascade", in which a web crawler creates a collection which is indexed and searched. Most of the designs of search engines consider the web crawler as just a first stage in web search, with little feedback from the ranking algorithms to the crawling process. The general approach for a search engine is modeled in a strict order: first crawling, then indexing, and then searching [21].

Web search process has two main parts: off-line and on-line. The off-line part is executed periodically by the search engine, and consists in downloading a sub-set of the web to build a collection of pages, which is then transformed into a searchable index. The online part is executed every time a user query is executed, and uses the index to select some candidate documents that are sorted according to an estimation on how relevant they are for the user's need.

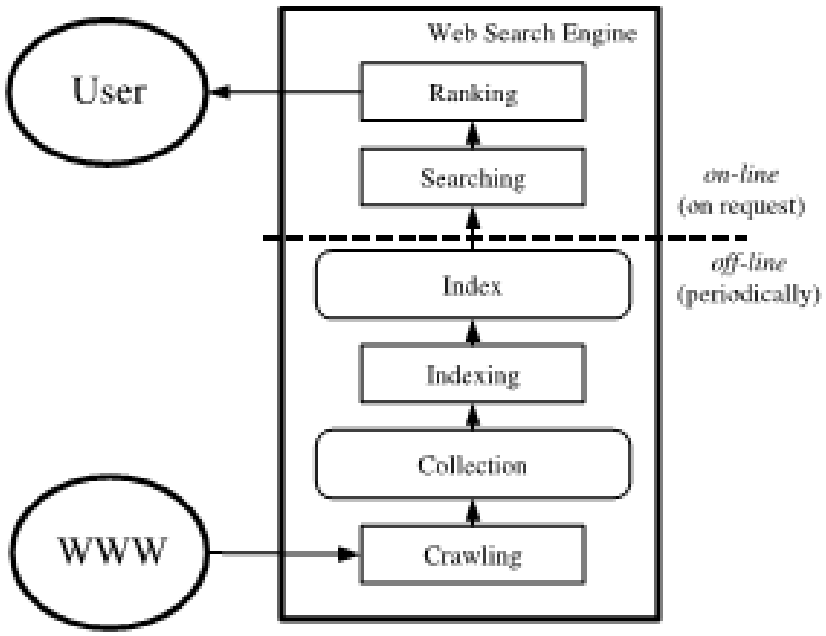

Figure 2.8. Periodic indexing of a web search engine [21].

Web pages come in many different formats such as plain text, HTML pages, PDF documents, and other proprietary formats. The first stage for indexing web pages is to extract a standard logical view from the documents. The most used logical view for documents in search engines is the "bag of words" model, in which each document is seen only as an unordered set of words. In modern web search engines, this view is extended with extra information concerning word frequencies and text formatting attributes, as well as meta-information about web pages including embedded descriptions and explicit keywords in the HTML markup.

There is a number of text normalization operations that are executed for extracting keywords, the most used ones are: tokenization, stopword removal and stemming. Tokenization involves dividing the stream of text into words. While in some languages like English this is very straightforward and involve just splitting the text using spaces and punctuation, in other languages like Chinese finding words can be very difficult. Stopwords are words that carry little semantic information, usually functional words that appear in a large fraction of the documents and therefore have little discriminating power for asserting relevance. In information retrieval stopwords are usually discarded also for efficiency reasons, as storing stopwords in an index takes considerable space because of their high frequency. Stemming extracts the morphological root of every word. In global search engines, the first problem with stemming is that it is language dependent, and while an English rule-based stemming works well, in some cases like Spanish, a dictionary-based stemmer has to be used, while in other languages as German and Arabic stemming is quite difficult.

Other, more complex operations such as synonym translation, detecting multiword expressions, phrase identification, named entity recognition; word sense disambiguation, etc. are used in some application do mains. However, some of these operations can be computationally expensive and if they have large error rates, then they can be useless and even harm retrieval precision.

For applications requiring extensive processing by the crawler, a single personal computer solution may be fast enough. An alternative is a distributed model in which a central control unit allocates tasks to many crawlers operating on individual personal computers. This is a practical solution in any organization in which many computers are left unused for periods of time, for example weekends. Many universities would also make ideal candidates, with whole laboratories of computers left idle during holiday periods.

Actually, in addition to the advantages of the network system, it restricts the users in a number of points of view. For example, the reports must not be very large in size, to prevent the traffic in the network. If the entire uncompressed web site were

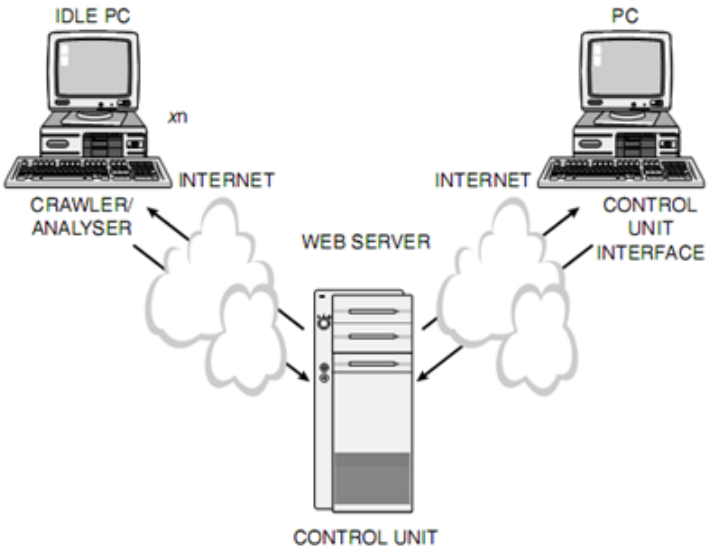

Figure 2.9. The distributed data mining crawler design [29].

to be 'reported', then this would represent an unacceptable doubling of network use.

2.4.6. Web Mining

The goal in Web content mining is to find useful information from text and multimedia content of Web pages. It includes information extraction and integration, knowledge discovery, and trend analysis.

Information extraction is a problem of extracting instances of predefined named entities and relations from text corpora. The underlying technologies include language processing techniques, Web-scale statistics, and unsupervised or semi-supervised learning methods.

Knowledge discovery is particularly relevant for mining blogs, customer reviews, and personal Web pages to infer opinions and preferences. In a case study on reputation systems abut opinion mining using econometrics, developed by A. Ghose, P. Ipeirotis, and A. Sundararajan [9], a method was proposed for econometrics-based opinion mining, which combined econometric techniques with language processing techniques to

derive the semantic orientation and strength of the opinion. This method identifies a “dollar value” of each opinion phrase and measures sentiment effectively. Typical features used include unigrams, bigrams, and trigrams conjoined with their part of speech.

Websites are not salt documentary pages and include heterogeneous context such as forms, links, and photos. Therefore, the content must be converted into a more structured format that would allow dialog modules to exploit. Junlan Feng and friends suggest building a web page semantic structure parser that segments each individual web page into smaller semantic units, and built a technique to create summaries for website directories.

Information conveyed on web pages is carried not only by their stream of texts, but also by the semantic structure of these pages, which are implicitly encoded in web documents. Therefore, segmenting a web page into smaller information blocks, and identifying the semantic categories of these smaller information units makes works simpler. A page can be segmented into semantic categories such as Page-Title, Form, Table-Data, FAQ-Answer, Menu, Bulleted-List, Normal-Content, Heading-List, Heading, Heading-Content, Picture-Label, and Other [1].

A website often consists of text data that is distributed in many web pages. Parsing a website involves reorganizing the textual data into a representation which can be manipulated by a machine.

As people spend more and more time on the Internet in today’s world of online shopping and banking; meanwhile, webpages get more complex in design and content. Web pages are mixed up with guides and menus attempting to improve the usability, but they often end up distracting from the actual content of interest. These “features” may include script and flash-driven animation, menus, pop-up ads, obtrusive banner advertisements, unnecessary images, or links scattered around the screen.

Most traditional approaches to removing clutter or making content more readable

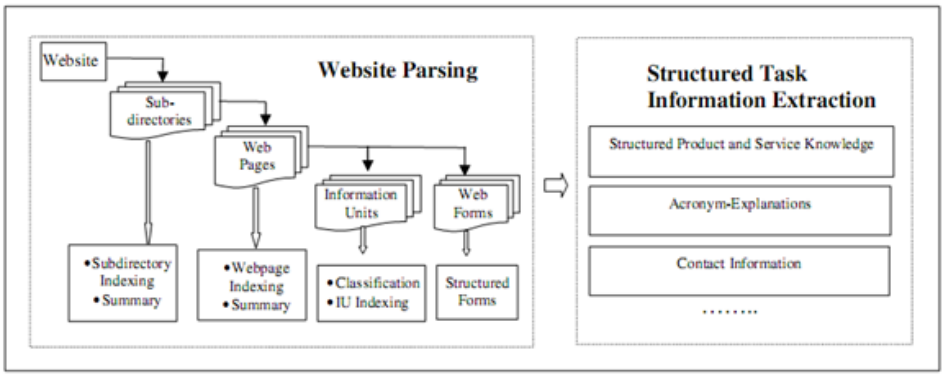

Figure 2.10. The diagram of website analyzer [4].

involve removing images, disabling JavaScript, etc., all of which eliminate the webpage's inherent look-and-feel. Opera, for example, utilizes their proprietary Small Screen Rendering technology that reformats web pages to fit inside the screen width [30].

For the experiments in the study of Gupta and friends [31], they chose 200 sites often visited by members of their group. Using these sites as the corpus, they worked on clustering them as specified in the above technique. They found that there is existed natural clustering in websites. They also found that most sites of similar genre followed similar layouts as they shared the same content and consequently the same target audience.

3. METHODOLOGY

The sum up the overall process, the main goal of the work is to handle the question or query of the user and overcome his/her request. The question is supposed to be asked in the specified use area. It can be a department, company, library or any other web site and the user can ask questions about the people, products, places and other information told in the site. In this work, the sites of Electrical & Electronics Engineering and Computer Engineering departments of Boğaziçi University are chosen as reference and the questions are built accordingly.

To understand the general operation and the flow of the system better, it may be a good way to go over the units seperately. The main units of the system which are responsible for a different task are Acoustic Recognizer, Classifier and Information Retriever which is composed of Crawler, Hunter, Ranker and Indexer.

3.1. System Overview

The classification algorithm is designed in a manner that gives the opportunity to be easily integrated to another system.

Figure 3.1 describes a Kiosk system in which the user communicates with the system verbally or by means of sign language. The automatic speech recognizer and automatic sign language recognizer are operating separately and not working on the same platform. In order to organize the separate parts, a messaging server is used. After the conversion of the verbal or sign language inputs to text, server delivers the output text to the interface unit. GUI performs the relevant operations to the incoming text and delivers it to either a TTS (text-to-speech) unit or to directly the user according to the system design.

The classifier part can be integrated to this system before returning the answer to the user. It takes the recognized text and decides to the most relevant class from

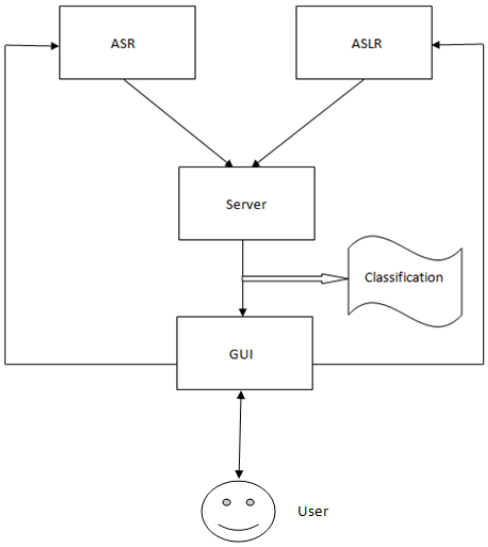

Figure 3.1. Block diagram of the overall system.

a previously constructed set, and then according to the found class, associates it with a keyword, in other words the label of the class. Then, the final step is to pick the best web page that would satisfy the users' expectance. To perform the last step, an indexing operation has to be done previously because the online check would take much more time. The site, which is the proper one for the Kiosk application (i.e. department site), has to be crawled and all the pages' links have to be extracted. Then the keyword is checked in all found links. According to the matching algorithm (tf-idf or any other, described in Section 2.4, the best links which can be a candidate answer to the user's answer or query are marked. The flowchart can be seen in Figure 3.2.

3.2. Recognizer

For the recognition process; as previously discussed in the Section 2.1.3, both a free form recognition and a grammar based recognition network can be used. The grammar based network increases the recognition accuracy values, where the free form network lets users to be free to ask any form of question without any limitation. The grammar consists of a question which can be about the courses, laboratories, instructors, options and the semesters.

Table 3.1. Form of the grammar.

```
Question = (year + year_q) or
           (course + course_q) or
           (option + option_q) or
           (lab + lab_q) or
           (inst + inst_q)

Year_q   = "Sınıf danışmanı kimdir" or
           "Sınıf dersleri nelerdir"

Course_q = "Dersi kaç kredidir" or
           "Dersinin labaratuvarı hangisidir" or
           "Dersinin önşartı var mı"

Option_q = "Opsiyon danışmanı kimdir" or
           "Opsiyon dersleri nelerdir" or
           "Opsiyon labaratuvarları hangileridir"

Lab_q    = "Labaratuvar görevlileri kimlerdir" or
           "Labaratuvar araştırma alanları nelerdir" or
           "Labaratuvar sponsorları kimlerdir"

Inst_q   = "Hangi dersleri veriyor" or
           "Hangi labaratuvarda çalışıyor"  or
           "Yayınları nelerdir"
```

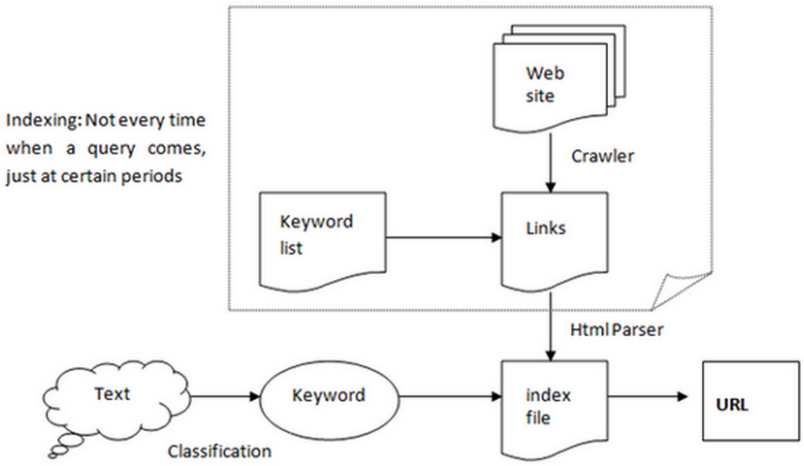

Figure 3.2. Block diagram of the response task.

The full grammar is shown in Appendix A.

3.3. Classifier

Open source machine learning tool MALLET is used for the classification process. Mallet is a Java-based package for statistical natural language processing, document classification, clustering, topic modeling, information extraction, and other machine learning applications to text [32]. It includes sophisticated tools for document classification: efficient routines for converting text to "features" and a wide variety of algorithms including Naïve Bayes, Maximum Entropy, and Decision Trees.

Before constructing the classifiers, Mallet does some data preprocessing. In this scope, it uses Instances to represent data, and preprocessing is the process of transforming the data to instance list.

MALLET instance is composed of the following 4 fields [33]:

- Name: This field acts as the Name of the instance and use for the identification of instances.

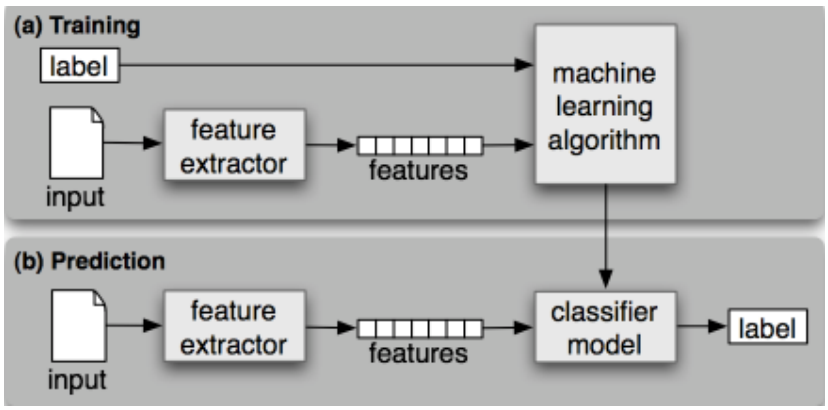

Figure 3.3. Overview of the classification process [33].

- Label: Label is primarily used to classify the instances. Labels represent the classes in the classification module.
- Data: The data is generally a Feature Vector or Feature Sequence (for example sequence of words). Classifier does the classification of the instances on the basis of the data of instances and provide the Label to it, if it is unclassified data.
- Source: It tells the information about the source of the Mallet Instance.

Table 3.2. Mallet training script.

```
mallet train-classifier -input path\importedData.mallet
-output-classifier path\outputClassifier.mallet -trainer MaxEnt
-trainer NaiveBayes -trainer C45 -trainer DecisionTree
-trainer MaxEntGE -trainer Winnow -trainer MaxEntL1
-trainer BalancedWinnow -trainer NaiveBayesEM -num-trials 10
-training-portion 0.9
```

A set of classifiers can be tested with partitioning the data to a training section and a test section. The script which trains the 90 % of data with all methods is shown in Table 3.2.

So, the first step is to import data to the system. As told above, the text data is kept as instances in a mallet file. Then, the features of the data are extracted and the classifier is constructed using the desired method according to the learnt information about the classes as a consequence of the training process. Finally, the text is classified with the previously constructed classifier.

3.4. Information Retriever

After classification of the question, first list of addresses has to be extracted which will be checked, than each one is visited and the most relevant is detected in response to the question of the user. As a consequence of the design decision, the system does not try to find the exact answer to the question, but offer the whole page to the user in which possible answers exist.

A similar approach to tf-idf is used in the work; however since the keywords are single words and do not contain unimportant words, checking only the occurrence frequency meets the requirements without any weight consideration. Therefore, the only task in the presence of the system is just checking how many times the word has taken place in each document and pick the one on which it passes most.

For this goal, after the crawler and the hunter units have a part in the scenario, the ranker and the indexer modules come into prominence. The hunted words are counted in each page and the one that has the most occurence in it is indexed as the answer page of the question. The index file is updated periodically since the web is alive and the contents can change over time.

3.4.1. Crawler

SPHINX (or WebSphinx), is a Java toolkit and interactive development environment for Web crawlers. The focus point, that is the discriminating feature of Sphinx, is the developing crawlers that are Web-site-specific, personally customized, and relocatable which means SPHINX -Site-oriented Processors for HTML INforma-

tion eXtraction- allows site-specific crawling rules to be encapsulated and reused in content analyzers, known as classifiers. Personal crawling tasks can be performed (often without programming) in the Crawler Workbench, an interactive environment for crawler development and testing. For efficiency, relocatable crawlers developed using SPHINX can be uploaded and executed on a remote Web server [34].

SPHINX has a graphical user interface called the Crawler Workbench, seen in Figure 3.4, that supports developing, running, and visualizing crawlers in a Web browser.

SPHINX regards the web as a directed graph of pages and links, which are reflected in Java as Page and Link objects. A Page object represents a parsed Web page, with methods to retrieve the page's URL, title, and HTML parse tree. Also among the attributes of Page is a collection of Link objects representing the outgoing hyperlinks found on the page. A Link object describes not only the target of the link (a URL), but also the source (a page and an HTML element within that page). Each Link and Page object also stores a set of arbitrary string-valued attributes, which are used by classifiers for labeling.

With the WebSphinx, 182 links are determined for the Electrical and Electronics Engineering site.

3.4.2. Hunter

When the listed pages are read, it is seen that the data is scattered in some html codes. With the raw data, it is very inefficient to process because you read, check and compare unnecessary data. If we have only the real data instead of the html codes, to check the keyword would be much easier. As a consequence, what we need is an HTML parser.

Another reason for parsing the page is the necessity for weighting in the parts of it. That is to say; since the importances of different regions of the page are not all equivalent, a weighting operation can fix the possible misleading results. For instance,

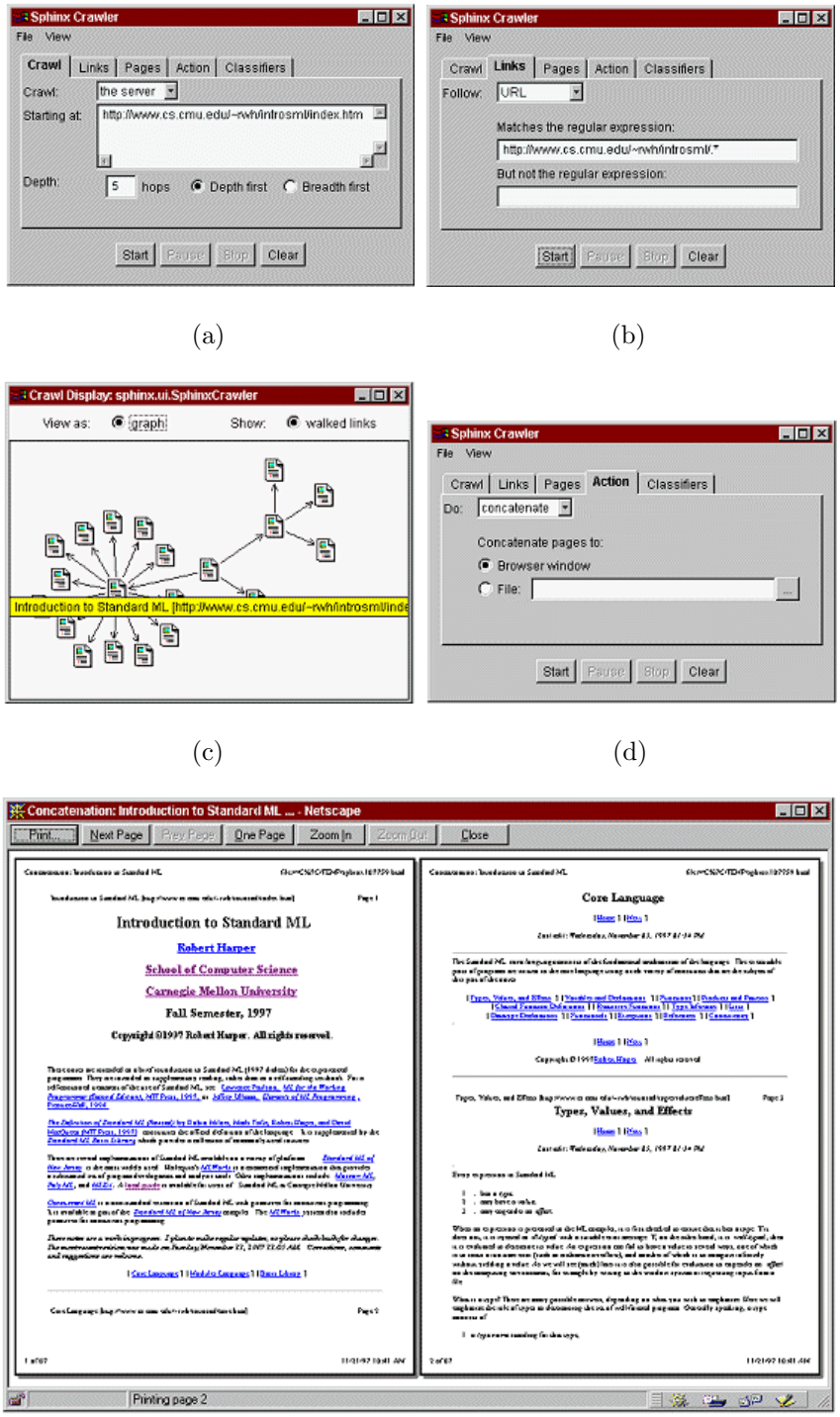

Figure 3.4. Crawler Workbench [34].

Table 3.3. Some of the links found after crawling.

http://www.ee.boun.edu.tr
http://www.ee.boun.edu.tr/Default.aspx
http://www.ee.boun.edu.tr/People/Faculty.aspx
http://www.ee.boun.edu.tr/People/Faculty/SelimSeker.aspx
http://www.ee.boun.edu.tr/People/Faculty/YagmurDenizhan.aspx
http://www.ee.boun.edu.tr/People/Faculty/OmerCerid.aspx
http://www.ee.boun.edu.tr/People/Faculty/LeventArslan.aspx

the title of the page can give more clues about the content of the page than the body of the text. In this situation, the factor of the title must be higher than the body text's one. This can be done thanks to a weighting function according to the regions of the page.

3.4.3. Ranker

After hunting the label word in each page, the next task is to check for closest one to the subject. As told before, it is generally procured by tf-idf techniques. By removing the unimportant words and operating a weighting function with an inverse document frequency procedure, it is checked that how many times a keyword has taken place in each document and pick the one on which it passes most.

The frequency is calculated as counting each word in a body text once and the words in the title twice. As told in the Hunter section; since the title of the page is more prominent than the body in guessing the content, its weight is specified as the double of the body. In this way, the contribution of the words in title to the content relevance is raised.

3.4.4. Indexer

As a result of the ranking, the most suitable pages for various questions are designated to offer to the user. The index file must be updated periodically to cope with the dynamism of the web. In this way, the changes applied into the context of the site are mirrored to the system. When a question is asked, the relevant page is called from the index file rather than going online and checking the web site. A preview of the index file is shown in Table 3.4.

Table 3.4. A preview of the index file.

```
okyay kaynak
http://mecha.ee.boun.edu.tr/kaynak.html
Count: 5
oğuzhan çiçekoğlu
http://www.ee.boun.edu.tr/People/Faculty/OguzhanCicekoglu.aspx
Count: 2
sabih tansal
http://www.ee.boun.edu.tr/People/Faculty/SabihTansal.aspx
Count: 2
selim şeker
http://www.ee.boun.edu.tr/People/Faculty/SelimSeker.aspx
Count: 3
viktor zekeriya
http://www.ee.boun.edu.tr/People/Faculty/ViktorZekeriya.aspx
Count: 3
yasemin kahya
http://www.ee.boun.edu.tr/People/Faculty/YaseminKahya.aspx
Count: 3
yasemin yücel
http://www.ee.boun.edu.tr/People/Faculty/DrYaseminYucel.aspx
Count: 3
yağmur denizhan
http://www.ee.boun.edu.tr/People/Faculty/YagmurDenizhan.aspx
Count: 13
```

4. EXPERIMENTS & RESULTS

In this section, the results of various experiments performed to observe the system performance and accuracy values are presented. The performances of all individual tasks are investigated and the success conditions are examined. In this extent, classification results, processing time values and recognition results are given. The effect of number of sentences added to the corpus in free form recognition is shown in a chart. In addition, the comparison between the web page selection results of weighted pages and not weighted pages is done. Finally the speaker based and gender based overall results are shared.

Accordingly, a number of questions asked to the system are in the categories of instructors, courses, laboratories, semesters and options of the departments. These five categories could be specified as the classes and the question could be classified in one of the five classes but in this situation, the system would not know what the question is exactly about. Therefore, we chose the classes in a deeper level, giving more information. For instance, it can be understood that a question is about "the instructor" instead of "an instructor". For this reason; classes are chosen in a lower level.

We have 110 classes totally, in accommodation with the grammar shown in Section 3.2. They are the labels of the instructors, laboratories, courses, semesters and the options. The whole list of the classes is shown in Appendix B.

To get a glance of the big picture, the system can be summarized as a combination of two main tasks: the acoustic recognition and the classification. The voice inputs are

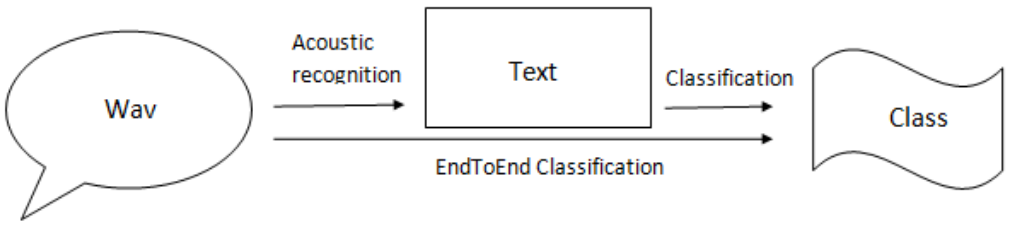

Figure 4.1. Main parts of the system.

recognized and converted to text and the text is classified after recognition.

4.1. Classification Results

The accuracy values of the classification methods can be seen in Table 4.1. The most successful one is observed as the Maximum Entropy. It takes a parameter to prevent overtraining. A smaller variance for the prior means that feature weights are expected to hover closer to zero, so extra evidence is required to set a higher weight. This parameter controls when we decide that we have converged.

Table 4.1. Accuracy results of different classifier trainers in percentages.

Trainer	ME Var=1	ME Var=1.79	ME GE	NB	NB EM	Winnow	Winnow BW	C45	Decision Tree
Acc.	97.80	99.00	0.86	52.00	52.00	43.00	98.00	98.00	0.50

4.2. Process Times

If the keyword search is done online in real time, it takes a while and users are generally not that patient; that is why indexing is used. The time values needed to

check the keywords are shown in Table 4.2.

The mean value is 74 seconds for 182 links, which is actually not a bad value; however one does not want to wait 74 seconds for each question. For this reason, the best results for each keyword is checked periodically and kept in an index file. In the index file, each keyword and the corresponding URL address are kept. A recall of the URL of a keyword from the index file takes only a few seconds.

4.3. Recognition Results

For the recognition process, a vocabulary which consists 50000 words is used and different amounts of sample sentences of the records are appended to the vocabulary to test the accuracy performance in the free form recognition. In this extent; 100, 1000, 10000 and 100000 sentences are mixed to the general corpus which includes mainly news records. As expected, with increased number of sample sentences from the recordings, the word error rate decreases.

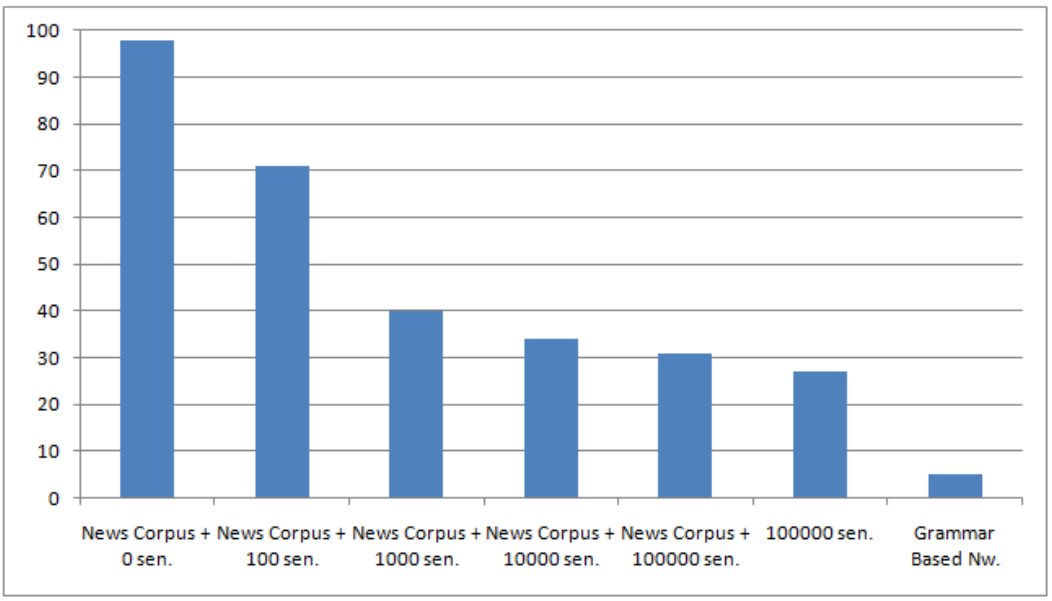

Figure 4.2. Sample sentences vs WER.

Table 4.2. Process times needed to check a keyword in all the pages.

Keyword	Trial	Time (secs)
“Ahmet Öncü”	1	65
	2	87
	3	74
	4	71
	5	69
“Busim”	1	81
	2	70
	3	75
	4	79
	5	72
“Electronics”	1	72
	2	75
	3	79
	4	75
	5	76
“Sophomore”	1	74
	2	71
	3	75
	4	79
	5	73
“EE 433”	1	72
	2	78
	3	65
	4	74
	5	68

4.4. The Effect of Weighting

The selection of the relevant web page corresponding to a label word is performed after a weighting operation in the pages. In this way; the importance of each section is emphasized. That is to say, if the title of a page is more important than the body in terms of the content, then the contribution of the title is increased with weighting. The effect of this operation to the accuracy of the selection can be seen in Figure 4.3.

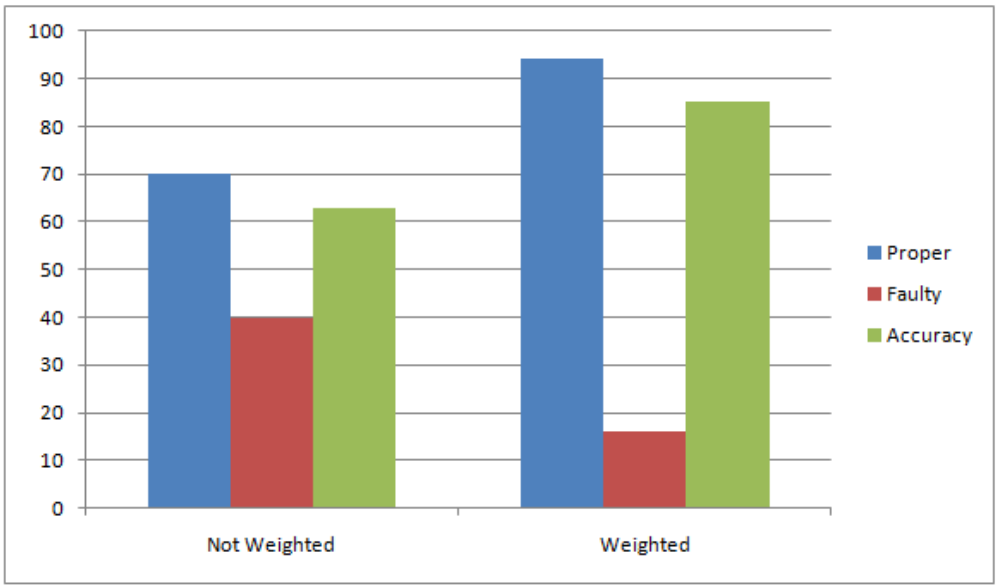

Figure 4.3. The effect of weighting.

4.5. Overall Results

From eight speakers -five males and three females- 1201 samples were taken totally. The recognition values show the success of the system in converting WAV samples into text and classification values indicate the success of the system in classification of the texts. In the classification process, since there are 110 classes, there is a probability of 0.009 for each class in the case of homogeneous distribution. A threshold value of 0.25 which is comfortably reasonable is set to reject the classification. That is to say, if the classification results in a class with a probability lower than 0.25, the system rejects to classify the acoustic recognition.

Table 4.3. Accuracy results of the experiments.

	True (#)	False (#)	Total (#)	Acc. (%)
Recognition (WER)	5323	290	5613	95
Recognition (SER)	1018	183	1201	84
Classification	1179	20	1199	98
EndToEnd	1044	155	1199	87

Table 4.4. Accuracy results of the experiments (Speaker based).

Speaker	Recognition (SER)				Classification				EndToEnd			
	T (#)	F (#)	All (#)	Acc. (%)	T (#)	F (#)	All (#)	Acc. (%)	T (#)	F (#)	All (#)	Acc. (%)
BT	146	96	242	60	240	2	242	99	173	69	242	71
BV	224	39	263	85	258	5	263	98	219	44	263	83
ED	136	1	137	99	135	2	137	98	134	3	137	97
HS	89	18	107	83	106	1	107	99	99	8	107	92
MD	82	7	89	92	86	3	89	96	84	5	89	94
IS	157	12	166	94	162	4	166	97	154	12	166	92
OC	91	7	98	94	95	1	96	98	90	6	96	93
MK	93	6	99	93	97	2	99	97	91	8	99	91

Table 4.5. Accuracy results of the experiments (Gender based).

Gender	Recognition (SER)				Classification				EndToEnd			
	T (#)	F (#)	All (#)	Acc. (%)	T (#)	F (#)	All (#)	Acc. (%)	T (#)	F (#)	All (#)	Acc. (%)
M	677	161	838	80	814	13	827	98	709	129	838	84
F	338	25	363	93	354	7	361	98	337	26	363	92

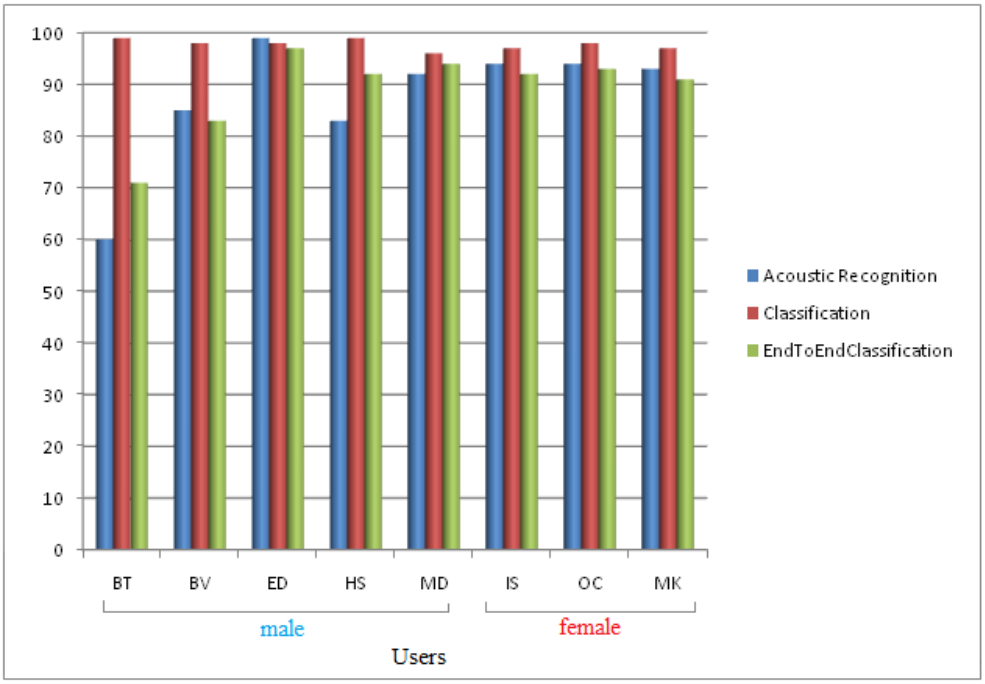

Figure 4.4. Speaker based accuracy results of the experiments.

For instance, if the acoustic recognition result is “ee”, then the probability distribution will be broad including all the courses; i.e. for ‘EE 101’ 0.095, for ‘EE 142’ 0.095. In this case, if the maximum probability is lower than 0,25; it decides not to finalize the decision: Reject! We see that 1199 total classification results, that is to say two rejections exist in the classification. They are the silence results in which the utterance is very silent compared to the environmental noise.

4.6. Discussion

In some pages, we see that the keyword passes only at the title and not any more. While the keyword of a subject is found in the relevant page at a level of occurence, in some cases other pages can also include the keyword at the same occurence frequency or even more frequently. The importance of the word is filtered thanks to weighting according to its position. In this way, if two pages have the same number of a keyword, the one containing it in the title wins the battle!

For example, by means of weighting, “Kadri Özçaldıran page” is found to be the

page related with the query “Kadri Özçaldıran”. Normally, this keyword is found in that page only one time at the title, where the faculty people page contains it two times. After the weighting operation, the correct page is selected.

As far as the experiment results are considered, the contribution of the classification entegration can easily be realized. As it can be seen in the Figure 4.5, while the recognition accuracy is 84 %, the end-to-end classification rate is 87 %. It means that, even if the utterance is recognized incorrect, it may be classified accurate.

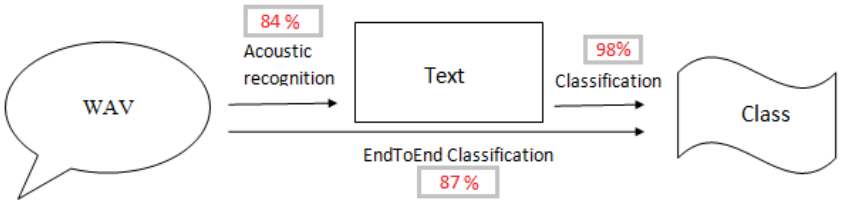

Figure 4.5. Results of experiments.

Some examples are given in Table 4.6 which demonstrate the situation where the utterance is recognized incorrect but classified correct.

Therefore, it is an undeniable fact that the injection of the classification to the system compensates some recognition errors and improves the accuracy rates from 84 % to 87 %.

When the recognition results are inspected, we see that female speakers’ results are better. In addition, there is a fluctuation between the results of speakers. It may stem from the tone of the speaker’s utterance or noise. However, except a few speaker, generally the sentence accuracy rates are around 94 %.

Table 4.6. The effect of classification.

Record718 Original : necs labaratuvar araştırma alanları nelerdir Recognized: necs labaratuvar Classified: NECS Attention : MisRecognized...!!!
Record730 Original : sabih Tansal hangi labaratuvarda çalışıyor Recognized: sabih tansal kitapları Classified: Sabih Tansal Attention : MisRecognized...!!!
Record979 Original : levent arslan projeleri nelerdir Recognized: levent arslan ödülleri nelerdir Classified: Levent Arslan Attention : MisRecognized...!!!
Record981 Original : murat saraçlar ödülleri nelerdir Recognized: murat saraçlar yayınları nelerdir Classified: Murat Saraçlar Attention : MisRecognized...!!!

5. CONCLUSION

In this thesis, the performance and the accuracy results of a system, which aims to carry out the recognition of a spoken question and finding relevant pages over a website, are investigated. In this way, users ask their questions and are directed to web pages on which they can find answers to their questions easily. Therefore, reaching information on a website gets easier by eliminating the manual searching steps.

In the recognition part, the performances of the free form recognition and using a grammar based recognition network are compared. In spite of the fact that grammar based network limits users to ask their questions in certain forms, it gives better results in terms of recognition accuracy. In free form recognition, adding larger volumes of sample sentences to the corpus is observed to improve the results.

As far as the classification results are investigated, it can be seen that 99 % accuracy is held with Maximum Entropy classifier. That is to say; if a sentence is not recognized incorrectly, then it is presumably classified correctly. As a result of this operation, the category of the question is determined.

After finding out the subject of the question, information about the subject is searched over the website. To perform the search, the website had to be crawled. That is to say, one should have an address list to visit each one. According to the address list, system goes to each link and search for the desired information.

Searching is applied thanks to keywords attached to the classes. The most fundamental factor affecting the search success is the weighting operation. By weighting, the importance of distinct regions in the web page are ranked in terms of the content. Thus, the contribution of an important region to the score is strengthened.

One of the most prominent operations in terms of process speed is indexing. Since online checking consumes a lot of time, after an execution, the results are recorded in

an index file and the results of queries are checked from the index file. The index should be updated periodically to follow the dynamic structure of the websites.

In short, with this application, users do not suffer from searching websites for information and reach what they want easily and quickly. In addition, with classification, even if the sentence is not recognized accurately, it may fall in the correct class and the relevant information is proposed to the user. In this way, possible misleadings are prevented. So, better results are taken in shorter times in the website search processes.

APPENDIX A: GRAMMAR

```
($question)

$question = $year $year_q or $course $course_q or $option $option_q or
$lab $lab_q or $instructor $instructor_q ;

$year = birinci or ikinci or UCUncU or dOrdUncU or master or doktora;
$year_q = sInIf danISmanI kimdir or sInIf dersleri nelerdir ;

$course = ee yUzbir or ee yUzkIrkiki or ee ikiyUzbir or ee ikiyUziki
or ee ikiyUzon or ee ikiyUzoniki or ee ikiyUzkIrk or ee ikiyUzkIrkbir
or ee ikiyUzkIrkiki or ee UCyUzdOrt or ee UCyUzon or ee UCyUzonUC
or ee UCyUzotuzUC or ee UCyUzotuzdOrt or ee UCyUzotuzbeS
or ee UCyUzotuzaltI or ee UCyUzelliiki or ee UCyUzaltmISUC
or ee UCyUzyetmiSUC or ee UCyUzyetmiSdOrt or ee dOrtyUzotuz
or ee dOrtyUzotuziki or ee dOrtyUzotuzUC or ee dOrtyUzotuzdOrt
or ee dOrtyUzotuzbeS or ee dOrtyUzotuzyedi or ee dOrtyUzotuzsekiz
or ee dOrtyUzotuzdokuz or ee dOrtyUzkIrkUC or ee dOrtyUzelli
or ee dOrtyUzellibir or ee dOrtyUzelliUC or ee dOrtyUzellidOrt
or ee dOrtyUzelliyedi or ee dOrtyUzaltmIS or ee dOrtyUzaltmISbir
or ee dOrtyUzyetmiS or ee dOrtyUzyetmiSUC or ee dOrtyUzyetmiSdOrt
or ee dOrtyUzyetmiSbeS or ee dOrtyUzyetmiSyedi
or ee dOrtyUzyetmiSsekiz or ee dOrtyUzyetmiSdokuz or ee dOrtyUzseksen
or ee dOrtyUzseksendOrt or ee dOrtyUzdoksan or ee dOrtyUzdoksanbir
or ee dOrtyUzdoksaniki or ee dOrtyUzdoksanUC or ee dOrtyUzdoksanyedi;
$course_q = dersi kaC kredidir or dersinde referans kitabI hangisidir
or dersinin labaratuvarI hangisidir or dersini kim veriyor
or dersinin OnSartI var mI or dersi konularI nelerdir  ;
```

```
$option = telekom or elektronik or kontrol or elektromanyetik
or iSaret iSleme;
$option_q = opsiyon danISmanI kimdir or
opsiyon dersleri nelerdir or
opsiyon labaratuvarlarI hangileridir;

$lab = beta or vavlab or busim or mnl or mecha or ds or isl or necs
or wcl or spg or lal or bumems  ;
$lab_q = labaratuvar gOrevlileri kimlerdir or
labaratuvar araStIrma alanlarI nelerdir or
labaratuvar sponsorlarI kimlerdir or
labaratuvar projeleri nelerdir or labaratuvar yayInlarI nelerdir or
labaratuvar OGrencileri kimlerdir;

$instructor = ahmet OncU or arda yalCInkaya or faik baSkaya or
gUnhan dUndar or oGuzhan CiCekoGlu or Omer cerid or Senol mutlu or
yasemin kahya or avni morgUl or sabih tansal or viktor zekeriya or
ISIl bozma  or kadri OzCaldIran or kemal cIlIz or mehmet akar or
okyay kaynak or yaGmur denizhan or selim hacIsalihzade or
aySin bartan ertUzUn or burak acar or bUlent sankur or
kIvanC mIhCak or levent arslan or murat saraClar or barhan Ozce or
ceyhun burak akgUl or hUseyin abut or okan ersoy or
ali emre pusane or emin anarIm or hakan deliC or mutlu koca or
gUlay Ozkan or mustafa ergen or yasemin yUcel or selim Seker or
heba yUksel ;
$instructor_q = hangi dersleri veriyor
or hangi labaratuvarda CalISIyor
or hangi alanda CalISIyor or OdUlleri nelerdir or projeleri nelerdir
or yayInlarI nelerdiror OGrencileri kimlerdir or kitaplarI nelerdir;
```

APPENDIX B: CLASSES

Table B.1. Classes.

BETA	BUMEMS	BUSIM	DS-Lab	ISL	LAL
Mecha	MNL	NECS	SPG	VAVlab	WCL
Freshman	Junior	Senior	Sophomore	M.S	Ph.D
Ahmet Öncü	Ali Emre Pusane	Arda Yalçınkaya	Avni Morgül	A. Baytan Ertüzün	Barhan Özce
Burak Acar	Bülent Sankur	C. Burak Akgül	Emin Anarım	Faik Başkaya	Gülay Özkan
Günhan Dündar	Hakan Deliç	Heba Yüksel	Hüseyin Abut	Işıl Bozma	Kadri Özçaldıran
Kemal Cılız	Kıvanç Mıhçak	Levent Arslan	Mehmet Akar	Murat Saraçlar	Mustafa Ergen
Mutlu Koca	Okan Ersoy	Okyay Kaynak	Oğuzhan ÇiÇekoğlu	Sabih Tansal	Selim Hsalihzade
Selim Şeker	Viktor Zekeriya	Yasemin Kahya	Yasemin Yücel	Yağmur Denizhan	Ömer Cerid
Ş. Mutlu	Comm	Magn	SP	Control	Electro
EE-101	EE-142	EE-201	EE-202	EE-210	EE-212
EE-240	EE-241	EE-242	EE-304	EE-310	EE-313
EE-333	EE-334	EE-335	EE-336	EE-352	EE-363
EE-373	EE-374	EE-430	EE-432	EE-433	EE-434
EE-435	EE-437	EE-438	EE-439	EE-443	EE-450
EE-451	EE-453	EE-454	EE-457	EE-460	EE-461
EE-470	EE-473	EE-474	EE-475	EE-477	EE-478
EE-479	EE-480	EE-484	EE-490	EE-491	EE-492
EE-493	EE-497				

REFERENCES

1. Feng, J., S. Reddy and M. Saraçlar, "WebTalk: Mining Websites for Interactively Answering Questions", *Interspeech 2005 - Eurospeech*, pp. 2485–2488, Proceedings of the 9th European Conference on Speech Communication and Technology, Lisbon, Portugal, 2005.

2. Karpov, A. and A. Ronzhin, "Information enquiry kiosk with multimodal user interface", *Pattern Recognition and Image Analysis*, Vol. 19, No. 3, pp. 546–558, 2009.

3. Fujie, S., D. Watanabe, Y. Ichikawa, H. Taniyama, K. Hosoya, Y. Matsuyama and T. Kobayashi, "Multi-modal Integration for Personalized Conversation: Towards a Humanoid in Daily Life", *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference*, pp. 617 – 622, 2008.

4. Feng, J., S. Bangalore and M. Rahim, "WebTalk: Mining Websites for Automatically Building Dialog Systems", *Proc. of IEEE ASRU*, pp. 168–173, 2003.

5. Feng, J., D. Hakkani-Tur, G. D. Fabbrizio, M. Gilbert and M. Beutnagel, "WebTalk: Towards Automatically Building Spoken Dialog Systems Through Mining Websites", *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference*, Vol. 1, pp. I – I, 2006.

6. Church, K., "Speech and Language Processing: Where Have We Been and Where Are We Going?", *Eurospeech*, pp. 1–4, Geneva, 2003.

7. Plotz, T. and G. Fink, "On the Use of Empirically Determined Impulse Responses for Improving Distant Talking Speech Recognition", *Hands-Free Speech Communication and Microphone Arrays, 2008.(HSCMA 2008)*, pp. 156 – 159, 2008.

8. McDonough, J. and M. Wolfel, "Distant Speech Recognition: Bridging the Gaps",

Hands-Free Speech Communication and Microphone Arrays, 2008.(HSCMA 2008), pp. 108 – 114, 2008.

9. Gilbert, M. and J. Feng, "Speech and language processing over the web", *Signal Processing Magazine, IEEE*, Vol. 25, pp. 18 – 28, 2008.

10. Young, S., D. Kershaw, J. Odell, D. Ollason, V. Valtchev and P. Woodland, *The HTK Book*, Cambridge University Press, 2000.

11. Reddy, R., "CMU Sphinx", `http://cmusphinx.sourceforge.net`, accessed at August 2011.

12. Drakos, N., "Networks, Dictionaries and Language Models", `http://www.ee.columbia.edu/ln/labrosa/doc/HTKBook21/node128.html`, accessed at August 2011.

13. Halverson, C., D. Horn, C. Karat and J. Karat, "The Beauty of Errors: Patterns of Error Correction in Desktop Speech Systems", *INTERACT '99*, 1999.

14. Rosenfeld, L., "Web Admin's Guide to Site Search Tools", `http://www.searchtools.com/guide/index.html`, accessed at August 2011.

15. Ehlen, P. and M. Johnston, "Location grounding in multimodal local search", *International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction*, ICMI-MLMI '10, pp. 32:1–32:4, ACM, New York, NY, USA, 2010.

16. Brüche, H., G. Knolmayer and M. Mittermayer, "Document Classification Methods for Organizing Explicit Knowledge", *Proceedings of the Third European Conference on Organizational Knowledge, Learning, and Capabilities*, Athens, Greece, 2002.

17. Witten, I. and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, second edn., Morgan Kaufmann, 2005.

18. Aly, M., "Survey on Multi-Class Classification Methods", http://www.vision.caltech.edu/malaa/research/aly05multiclass.pdf, accessed at August 2011.

19. Nigamy, K., J. Laertyy and A. McCallumzy, "Using Maximum Entropy for Text Classication", *IJCAI-99 Workshop on Machine Learning for Information Filtering*, pp. 61–67, Stockholm, Sweden, 1999.

20. Keramati, A. and N. Yousefi, "A Proposed Classification of Data Mining Techniques in Credit Scoring", *Proceedings of the 2011 International Conference on Industrial Engineering and Operations Management*, Kuala Lumpur, Malaysia, 2011.

21. Castillo, C., "Effective Web Crawling", *SIGIR Forum*, Vol. 39, No. 1, pp. 55–56, June 2005.

22. Yates, R. and B. Neto, *Modern Information Retrieval*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

23. Arasu, A., J. Cho, H. Molina, A. Paepcke and S. Raghavan, "Searching the Web", *ACM Transitions on Internet Technology*, Vol. 1, pp. 2–43, August 2001.

24. Singhal, A., "Modern information retrieval: a brief overview", *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, Vol. 24, pp. 35–43, 2001.

25. Gregg, D. and S. Walczak, "Adaptive web information extraction", *Communications of The ACM*, Vol. 49, No. 5, pp. 78–84, 2006.

26. Salton, G. and C. Buckley, "Term-weighting approaches in automatic text retrieval", *Information Processing and Management*, pp. 513–523, 1988.

27. Freeman, T., "Minig Music for Emotion Based on Lyrics and Audio Features",

http://seis.bris.ac.uk/ tf7960/data.html, accessed at August 2011.

28. Shkapenyuk, V. and T. Suel, "Design and Implementation of a High-Performance Distributed Web Crawler", *In Proceedings of the International Conference on Data Engineering*, pp. 357–368, 2002.

29. Thelwall, M., "A web crawler design for data mining", *Journal of Information Science*, Vol. 27, No. 5, pp. 319–325, 2001.

30. Gupta, S., G. Kaiser, D. Neistadt and P. Grimm, "DOM-based content extraction of HTML documents", *WWW '03 Proceedings of the 12th international conference on World Wide Web*, 2003.

31. Gupta, S., G. Kaiser and S. Stolfo, "Extracting context to improve accuracy for HTML content extraction", *WWW '05 Special interest tracks and posters of the 14th international conference on World Wide Web*, 2005.

32. McCallum, A., "MALLET: A Machine Learning for Language Toolkit", http://mallet.cs.umass.edu, accessed at August 2011.

33. Bird, S., E. Klein and E. Co, "Learning to Classify Text", http://nltk.googlecode.com/svn/trunk/doc/book/ch06.html, accessed at August 2011.

34. Miller, R. and K. Bharat, "SPHINX: A Framework for Creating Personal, Site-Specific Web Crawlers.", *Computer Network and ISDN Systems*, Vol. 30, pp. 119–130, 1998.