AN INTELLIGENT AGENT APPLICATION FOR BUYERS IN ELECTRONIC COMMERCE

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Computer Engineering, Computer Engineering Program

by
Ferkan KAPLANSEREN

March, 2008
İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"AN INTELLIGENT AGENT APPLICATION FOR BUYERS IN ELECTRONIC COMMERCE"** completed by **FERKAN KAPLANSEREN** under supervision of **Prof. Dr. TATYANA YAKHNO** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Tatyana YAKHNO

Supervisor

Prof. Dr. Alp R. KUT	Assit. Prof. Dr. Zafer DİCLE
Thesis Committee Member	Thesis Committee Member

Prof. Dr. Mehmet Ertuğrul ÇELEBİ	Assit. Prof. Dr. Adil ALPKOÇAK
Examining Committee Member	Examining Committee Member

Prof.Dr. Cahit HELVACI

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

First of all, I would like to thank Prof. Dr. Tatyana YAKHNO for being my supervisor. Thanks a lot for her patience, smiling face, guidance, advices, support and critics. Great thanks to my other supervisor Instructor. Dr. M. Kemal ŞİŞ who proposed the concept of my thesis. His experience and guidance helped me to think more analytically to concentrate on my thesis. It is a great chance to meet him.

Many thanks to my committee members Prof. Dr. Alp KUT and Assoc. Prof. Dr. Zafer Dicle. Although they are two of the busiest academicians at university, they allowed time to discuss on my thesis. Thanks for their critics, advices and support. I would like to thank Assoc. Prof. Dr. Adil ALPKOÇAK for his help with valuable suggestions and discussions that he has provided me.

I would like to express my deepest gratitude to my friend Instructor Korhan GÜNEL for his continuous support and help during every step of my studies from beginning to the end. It is a big chance to have a friend like Korhan. I would like to thank to Instructor Rıfat AŞLIYAN for his help about finding sequential word group frequencies. Great thanks to my office-friend Instructor Sabri ERDEM for his friendship, critics, support and advices about my thesis. I would also like to thank to the friends working at Quantitative Methods Division of Faculty of Business Administration and Department of Computer Engineering of Dokuz Eylül University and İzmir Meteorological Service for any support they have provided.

Words may not be enough to explain my feelings for my family. Shortly, thanks for everything to my brothers Seren and Yusuf, mother Nurten and father Sırrı. Lastly, many thanks to my other brother Emrah YURTSEVER whom this thesis is dedicated. His memory will live on forever in my heart.

Ferkan KAPLANSEREN

AN INTELLIGENT AGENT APPLICATION FOR BUYERS IN ELECTRONIC COMMERCE

ABSTRACT

One of the mostly used areas of information agents is electronic commerce. At business to customer commerce which is one of the parts of electronic commerce, web pages including information about properties of products such as price, color, size, etc. enable customers to be more conscious to shop. Because of browsing thousands of web pages will be impossible for customers, intelligent information agents play essential role for collecting, processing data in Internet and producing meaningful results for customers.

This thesis includes a new information extraction algorithm which is based on frequencies of sequential word groups, and design and application of an intelligent agent that acquire knowledge from Internet to automatically build a knowledge base.

To evaluate the algorithm, an intelligent agent which is called FERBot (Feature Extractor Recognizer Bot) is designed. FERbot collects the labeled or unlabeled information from structured or unstructured web pages to inform customers about the features of products and value range of these features for reasonable and cost efficient shopping.

This new algorithm is an alternative for the current information extraction algorithms that extract information from web. It plays an important role for text miners and natural language processors to process the extracted information.

Keywords: Electronic Commerce, Intelligent Agent, Information Extraction, Knowledge Acquisition, Frequencies of Sequential Word Groups.

ELEKTRONİK TİCARET ALICILARI İÇİN AKILLI BİR AJAN UYGULAMASI

ÖZ

Bilgi ajanlarının çoğunlukla kullanıldığı alanlardan birisi de elektronik ticarettir. Elektronik ticaretin bölümlerinden biri olan işletmelerden müşterilere yönelik ticarette; fiyat, renk, ebat, vb. gibi ürün özellikleri hakkında bilgi içeren web sayfaları, müşterilerin daha bilinçli alışveriş yapmalarını sağlar. Binlerce web sayfasını incelemek müşteriler için imkânsız olacağından, akıllı bilgi ajanları, İnternet üzerindeki verilerin toplanması, işlenmesi ve anlamlı sonuçlar olarak müşterilere sunulmasında başlıca rolü oynamaktadır.

Bu tez, ardışık kelime frekanslarına dayanan yeni bir bilgi çıkarımı algoritmasını ve İnternet üzerindeki bilgi kazanımını sağlayıp otomatik olarak bilgi tabanını inşa eden akıllı bir ajan tasarımını ile uygulamasını içerir.

Algoritmayı değerlendirmek için FERbot (Feature Extractor Recognizer Bot: Özellik Çıkarıcı Tanımlayıcı Robot) adında akıllı bir ajan tasarlandı. FERbot, İnternet üzerindeki yapılandırılmamış yada yarı yapılandırılmış internet sayfalarından etiketlendirilmiş yada etiketlendirilmemiş verileri toplayarak ürünlerin özelliklerini ve bu özelliklerin değer aralıklarını müşterilere bildirerek daha bilinçli ve tutarlı alışveriş yapmalarını sağlar.

Bu yeni algoritma, web üzerinden bilgi çıkarımı yapan mevcut algoritmalara bir alternatif olduğu gibi veri madencilerinin ve doğal dil işleyicilerinin, metinlerden çıkarılmış bilgileri işlemelerinde önemli rol oynayacaktır.

Anahtar Kelimeler: Elektronik Ticaret, Akıllı Ajan, Bilgi Çıkarımı, Bilgi Kazanımı, Ardışık Kelime Grupları Frekansları.

CONTENTS

Page

CHAPTER ONE
INTRODUCTION

1.1 Background of the Motivation

Recently, intelligent agents and electronic commerce (e-commerce) are two important and exciting areas of research and development in information technology. Day by day, the importance of the relationship between these areas is getting higher. World Wide Web is one of the platforms meeting electronic commerce and intelligent agents. Especially, the information agents searching for the information from Internet take an importance for this meeting. We know that huge amount of information which is unstructured is available at Internet. It is not as easy as it seems to reach the accurate information. Sometimes it takes more time than buyer offers to spend. Decreasing the time for searching information from web is definitely one of the most important tasks of e-commerce agents.

As a part of shopping through Internet, when a buyer wants to buy something for the first time and doesn't know anything about product, first of all he would want to get some information about that product. The information about a product can be the price of it, its model, brand, size, version, etc.

Customers may use a shopbot to find the lowest price for a product. If the code number of a product is uniquely known in Internet, while using a shopbot, there would be no need for a customer to search more than one advertisement, because, shopbot would find the lowest price. But, customers also wonder the general characteristics of products to fit the lowest price. Every product that is sold in an e-commerce store has several features that are related to that product. For example, resolution is a feature of computer monitor and television but it is not a feature of a shirt. For a buyer it is a good idea to know the features of a product as much as its price to decide whether it is worth to give that price. The importance of the features of a product changes according to different customers. Some of the customers

consider the price of the product to be important while some other customers consider the size, power etc. to be important.

Moreover features associated with a product change from time to time. New features are introduced and old features become trivial. For example GPRS was not a feature for mobile phone eight or ten years ago but it can now be considered as a standard feature. For that reason, new features of the product become more important than the old features. To inform the customers about standard and new features of products will be an important job of e-commerce buyer agents.

Brynjolfsson, Dick and Smith (2003) quantify consumer benefits to search and place an upper bound on consumer search costs. They analyze the consumer heterogeneity present in the data and what the implications are in terms of consumer behavior and search benefits and costs. Across the various consumer types, they find that first screen consumers are the most price-sensitive. Consumers that scroll down multiple screens, on the other hand, have low price sensitivity but brand appears to play a relatively important role for them, as presumably they choose to inspect lower screen because they care relatively more about other attributes besides price.

We take into account ideas of Brynjolfsson, Dick and Smith (2003) to find out some important factors regarding Internet commerce. They firstly say, the presence of search costs in this setting of nearly-perfect price and product information provides one possible explanation for the continuing presence of high levels of price dispersion in Internet markets. Secondly, the importance of non-price factors even for homogeneous physical products highlights the importance of retailer differentiation on the internet through service characteristics and reputation.

If a buyer doesn't aware of shopping (buyer) agents, one way of finding the information about a product is to use a search engine. Search engines help people in an organized way to find information they are looking for. In general, search engines refer to web search engines. Web crawlers and spiders of search engines visit Internet pages from links included by another page. They analyze huge amount of

unstructured material to look for data, information and knowledge. They summarize and index the pages into massive databases. Buyers visit a lot of Internet pages that come as a result of the work of a search engine. But they may not find enough time to visit thousand pages to read all of them. Most of people retry their queries by changing them after looking for a few sites if they can not see the information they are looking for.

E-commerce shopping (buyer) agents help customers by finding the products and services from Internet. On the other hand, most web users expect more human-like intelligence from their web applications. Recently, e-commerce is one of the most important and money making engine of web based applications. The e-commerce platform will eventually provide a platform where manufacturers, suppliers and customers can collaborate (Chaudhury, 2002). The expectation may be realized, apart from other factors, by subscribing more participants in this collaboration. The point of interest being e-commerce, trusting "one's personal computer" –it's application in reality- is a matter of using intelligent e-commerce agents" that can extract the features of desired commodity.

According to Leung and He (Leung, 2002), one of the things that an ecommerce agent can do is to monitor and retrieve useful information, and do transactions on behalf of their owners or analyze data in the global markets. Shopping bots (buyer agent) like eBay (ebay, 2006, http://www.ebay.com) find products and services that customers are searching for. eBay uses collaborative filtering, for that reason after a product search, at the bottom of the page of eBay, there is a list of similar products that other customers who did the same search looked at. Jango (Jango, 2006 http://www.jango.com) and adOne (adone, 2006, http://www.addone.com) are other examples for e-commerce agents (Chaudhury and Kilboer 2002). Jango does comparative price shopping. It is a wrapper visiting several sites for product queries and obtains prices. "adOne" searches classified advertisements like used cars. Another shopping bot that is used by akakce (online web store) is called as bilbot (bilbot, 2006, http://www.akakce.com) (Karatas, 2001) makes price comparison.

Search engines and intelligent agents use some techniques to mimic the intelligence that human beings expect. The scope of the data in web pages consists of text, video, images, sounds and etc. Nearly all web pages include text where as they don't have movies or images. According to Hui and Yu (2005), much of the information is embedded in text documents and relies on human interpretation and processing, at the same slow pace as before. If we narrow the scope of information available in World Wide Web (WWW) into text documents, the basic function of search engines and intelligent agents becomes understanding the semantics of text documents.

Semantic search is one of the main parts of this searching process. Semantic Web Consortium defines semantic web as providing a common framework that allows data to be shared and reused across application, enterprise, and community boundaries (Semantic Web, 2006, http://www.w3.org/2001/sw/). It defines a common format for interchange of data and explains how the data which refers real objects in real life should be recorded. Semantic web project has been around for over a decade, but it seems that this project will not be realized for near future. Managing this unstructured information implies discovering, organizing and monitoring the relevant knowledge to share it with other applications or users. Therefore, deriving semantics from unstructured or semi structured web will be as important as designing a semantic web.

To extract the semantics from text documents will be quite a complicated matter unless web pages are designed in a standard format. Moens (2006) defines Information Extraction (IE) as the identification, and consequent or concurrent classification and structuring into semantic classes, of specific information found in unstructured data sources, such as natural language text, making the information more suitable for information processing tasks. Relation between IE and the fields of Natural Language Processing (NLP), Knowledge Acquisition (KA), Machine Learning (ML) and Data Mining (DM) of Artificial Intelligence (AI) is important to process the text in WWW.

The World Wide Web is now the largest source of information; yet, because of its structure, it is difficult to use all that information in a systematic way. One problem for the information extraction systems and software systems that need information from WWW is the information explosion which increases day by day. Chuang and Chien (2005) fight with this huge amount of data by combining their study with search process of search engines to extract features from the retrieved highly ranked search-result snippets for each text segment. Their aim is to organize text segments into hierarchical structure of topic classes. CAN (Can, 2006) summarizes the information explosion problem and illustrates some recent developments for information retrieval research of Turkish Language. He emphasizes some methods like vector space models, term frequency, inverse document frequency and document query matching for more effective information retrieval against the information explosion.

Most of the information extraction systems need statistics to process the unstructured text systems. Filatova and Hatzivassiloglou (2003) presents a statistical system that detects, extracts, and labels atomic events at the sentence level without using any prior world or lexical knowledge. Freitag (2000) investigates the usefulness of information like Term frequency statistics, Typography, Meta-text (HTML tags), Formatting and layout by designing learning components that exploit them. Like Freitag, Burget (2005) proposes an alternative information extraction method that is based on modeling the visual appearance of the document. Burget (2004) also explains a hierarchical model for web documents that simply presents some structured data such as price lists, timetables, contact information, etc.

Objective of Mukherjee, Yang, Tan and Ramakrishnan (2003) is to take a HTML document generated by a template and automatically discover and generate a semantic partition tree where each partition will consist of items related to a semantic concept.

Hidden Markov Model (HMM) is one of the main methods that uses the probabilty for lexical analysis. Bikel, Schwartz and Weischedel (1999) present

IdentiFinder, a HMM that learns to recognize and classify names, dates, times, and numerical quantities. Borkar, Deshmukh and Sarawagi (2001) present a method for automatically segmenting unformatted text records into structured elements. Their tool enhances on HMM to build a powerful probabilistic model that corroborates multiple sources of information including, the sequence of elements, their length distribution, distinguishing words from the vocabulary and an optional external data dictionary.

Kerninghan and Pike (Kernighan, 1999) applied "Markov Chain" on *n*-words phrase to produce meaningful or non-meaningful texts from documents. They consider there are some prefixes and suffixes in an *n*-words phrase. That is, for a three words phrase, word 1 and word 2 are prefix and word 3 is the suffix. By changing the suffix of the phrase according to the frequency of the three words phrase, sentences are produced for a meaningful or non-meaningful text.

Two of the most popular software Topocalizer (Topocalizer, 2006, http://www.topicalizer.com) and Textalyser (Textalyser, 2006, http://www.textalyser.net) generate text analysis statistics of web pages and texts. They mainly perform lexical analysis (e.g., to find the number of words in a text or average number of words per sentence), phrasal analysis to find most frequent two-words or *n*-words phrases, finally, textual analysis to find the number of sentences and number of paragraphs in a document.

GTP (General Text Parser) (Berry, 2003) is other software that parses documents and provides matrix decomposition for use in information retrieval applications. GTP creates a term-by-document matrix (Berry 1999, 2003) where frequencies of terms are the rows and corresponding documents are columns.

Results of these analyses can be used for different purposes. Generally, statistical results of text analyzing are used for some classification algorithms. Zhou and Chou (2001) give brief information about some classification algorithms like *k*-nearest neighbor, naive bayesian, vector space models and hierarchical algorithms.

Radovanovic and Ivanovic (2006) describe some key aspects of their meta-search engine system CatS (including HTML parsing, classification and displaying of results), outlines the text categorization experiments performed in order to choose the right parameters for classification, and puts the system into the context of related work on (meta-)search engines. Kucukyilmaz, Candanoğlu, Aykanat and Can (2006) use k-nearist neighbour and Bayesian algorithms to predict the gender of chatting people. A specific example for text mining is MedTAS (Uramoto, 2004) which is built on top of the IBM-UIMA (Unstructured Information Management Architecture) framework which focuses on extracting knowledge from medical texts such as clinical notes, pathology reports and medical literature. But, search engine of UIMA supports a query language called XML.

Relations between data, information and knowledge also define the characteristics of information extraction systems and expert systems. Some knowledge engineers use information extraction techniques to build knowledge bases. Beveren (2002) presents 3 propositions about data-information-knowledge: P1, data and information are the only forms that are captured, transferred or stored outside the brain. P2, knowledge can only exist within individual human brains. P3, information is acquired through the sensors to be processed in the brain, and new knowledge is created from the processing of information.

Some knowledge engineers try to construct knowledge bases from web data based on information extraction using machine learning (Craven et al., 2000). To collect product knowledge, Lee (2004) works with specially designed interfaces to allow domain experts to easily embed their professional knowledge in the system. Gomes and Segami (2007) study the automatic construction of databases from texts for problem solvers and querying text databases in natural language, in their research. Also as Rao, Kweku and Bryson (2007) emphasis, the importance of the quality of the knowledge that is included by the knowledge management systems.

One of the important parts of knowledge base systems is the ontology to reuse and share the data between different applications systems constructed on same problem

domain. The system that is offered by Shamsfard and Barforoush (2004) starts from a small ontology kernel and constructs the ontology through text understanding automatically. Their study converts input texts to ontology and lexicon elements in the following steps.

a) Morphological and syntactic analysis and extracting new word's features.
b) Building sentence structures.
c) Extracting conceptual–relational knowledge (primary concepts).
d) Adding primary concepts to the ontology.
e) Ontology reorganization.

1.2 The Purpose of the Thesis

The purpose of this thesis is creating an intelligent agent that will inform customers about features of a product before the shop through Internet. The main idea is to visit all e-commerce web pages that sell the desired commodity to gather the information about that commodity.

This thesis is based on both theoretical and practical studies. For designing our agent, we developed a new algorithm based on sequential word group frequencies to extract the features of products from web pages. Also, to evaluate this algorithm we developed our own software called FERbot (Feature Extractor Recognizer bot).

In this study, we propose a new algorithm based on Information Extraction and Knowledge Discovery from e-commerce web pages. We study lexical contextual relations to show that relative frequencies of sequential word groups allow agent to derive semantics from a collection of e-commerce web pages. For that reason, the input set of this study is the set of web pages, $WP = \{wp_1, wp_2, \ldots, wp_n\}$, from e-commerce web sites in Internet. The purpose is to produce the output which is the definition of any product in terms of possible features and values. A product is defined as a set of features like $P = \{F_1, F_2, F_3, \ldots, F_n\}$ and a feature is defined as a set of values like $F_i = \{V_1, V_2, V_3, \ldots, V_k\}$. Syntactical rules of languages define the

orders of the words (adjective, noun phrase etc.) in a regular sentence or in a plain text. Statistics about these ordered words can be used to find out how the documents are organized. We demonstrate that structure and semantics of a text can be systematically extracted by lexical analysis.

This algorithm that we developed is an alternative for the current statistical and machine learning techniques for information detection and classification for the purpose of extracting information from web. Also, text miners can use our algorithm for knowledge discovery within the concept of statistical data mining techniques to operate the information extracted from texts.

We developed a software called FERbot (Feature Extractor Recognizer bot) to evaluate our approach. FERbot learns the features of products automatically by extracting sequential word groups (*SWG*) from web pages and finds out the values of these features to automatically construct a knowledge base for products.

Our software FERbot is designed as a centralized system collecting the all information in its own server and has three important jobs; collecting, modeling and presenting the data. Learning part of our software FERbot, firstly learns the features of products automatically by extracting sequential word groups from web pages. Then recognizer part of FERbot finds out the values of these features to automatically construct a knowledge base for products. We consider a top-down design to find the features and values of products.

FERbot visits web sites of products to index the products and their features before buyers. Thus, customers can see the classification of product features and the values of these features without visiting the Internet pages. Remaining is to make the customer to be educated about the product and to narrow down the features of the product from general to specific.

1.3 Thesis Organization

The structure of the thesis is organized as follows.

Chapter 2 is about the Information Extraction and Knowledge Acquisition. This chapter begins with the descriptions of Information Extraction in general and in relation to our thesis. The general architecture of an Information Extraction system is mentioned in this chapter. The other main concept of this chapter is Knowledge Acquisition. We give some general information about knowledge engineering, knowledge bases and ontology. Most important applications of Information Extraction and Knowledge Acquisition related to our studies are mentioned with their examples in chapter 2.

Chapter 3 includes information about e-commerce, intelligent agents and relations between them. Main features of e-commerce agents are mentioned in detailed form. Comparisons of e-commerce agents and their main jobs are given in chapter 3. We explain the differences and similarities between known intelligent agents and FERbot. We find a category of intelligent agents for FERbot in this chapter. Chapter 3 also includes information about main characteristics of FERbot.

Chapter 4 is the heart of this thesis. Based on the developments worked in previous chapters, we introduce our new algorithm and software FERbot that is an implementation for this algorithm. The details of the algorithm we developed to extract information and to build a knowledge base are explained. The architecture of FERbot which has three main components which are Information Gathering, Knowledge Modelling and User Interface are mentioned in chapter 4. Knowledge Modelling part of the architecture is the section that evaluates our approach. Our approach bases on Sequential Word Groups (SWG). Obtaining SWGs from web pages and usage of them are given in chapter 4. Algorithms and flow charts of our approach are given in this chapter.

Chapter 5 presents how FERbot produces different results according to different domains and different web pages. Some case studies are used to illustrate the effective use of FERbot. Examples are “araba”, “bmw”, “renault” and “villa”. FERbot is used for these studies and details of working process are given step by step. This chapter shows how results change if the thresholds of the parameters are changed. The comparision of results of examples are given in this chapter. Also, chapter 5 includes results of these studies.

Chapter 6 states our conclusion and discuss about the possible future extensions of this study. The main contributions of this thesis for Information Extraction, Knowledge Acquisition and Text Analyzing are given and the way of improving this algorithm and software of FERbot for future works are discussed. The possible usage areas of our intelligent agent FERbot except e-commerce field are mentioned in this chapter.

CHAPTER TWO
BASIS OF OUR APPROACH IN INFORMATION EXTRACTION AND KNOWLEDGE ACQUISITION

2.1 Information Extraction

Day by day, Information Extraction (IE) methods are improved against to the information explosion. A shopbot extracts the prices of products to compare them. It presents them in a more structural form. Such a system use methods like pattern matching, semantic analysis or natural language processing. The aim is to add a meaning to the text inside the web page. This text can be in unstructured or semi structured form. If the data is written in natural language form, this data is in unstructured format but some web pages including information about scientific labels, product prices, medicine information or discographies use semi structured forms. If data is converted into structured form, processing these raw data will be an easy job for computer applications like Data Mining and Information Retrieval.

IE is the subcategory of Artificial Intelligence. Most of the data is available in text or image format in World Wide Web (WWW). It is not easy or possible for human beings to read every document in WWW. Our computers also can not easily find out the information we need from this unstructured source if they don't use some IE methods.

Zhou and Zhang (2006) define main three objectives of IE, up to the scope of the NIST (National Institute of Standards and Technology) Automatic Content Extraction (ACE) program as Entity detection and tracking, Relation detection and characterization, and Event detection and characterization. According to Grishman (1997), the process of information extraction has two major parts; first the system extracts individual "facts" from the text of a document through local text analysis. Second, it integrates these facts, producing larger facts or new facts (through) inference. Stevenson (2007) emphasizes the importance of usage of multiple sentences for facts instead of single sentence.

Riloff and Lorenzen (1999) define Information Extraction systems as the systems extract domain-specific information from natural language text. The domain and types of information to be extracted must be defined in advance. IE systems often focus on object identification, such as references to people, places, companies, and physical objects. Cowie and Lehnert (1996) specify the goal of information extraction research is to build systems that find and link relevant information while ignoring extraneous and irrelevant information.

Moens (2006) defines Information Extraction as the identification, and consequent or concurrent classification and structuring into semantic classes, of specific information found in unstructured data sources, such as natural language text, making the information more suitable for information processing tasks. Moens continues: The use of the term extraction implies that the semantic target information is explicitly present in a text's linguistic organization, i.e., that it is readily available in the lexical elements (words and word groups), the grammatical constructions (phrases, sentences, temporal expressions, etc.) and the pragmatic ordering and rhetorical structure (paragraphs, chapters, etc.) of the source text.

Extracted data must be semantically classified into a group. The extracted information sometimes becomes a word, word compound, adjective phrase or semantically combined sentences. In our study we will use the sequential word groups to create a hierarchical classification. Our Information Extraction system uses some extraction patterns which are constructed automatically.

Information Extraction Systems are more powerful if they are used within natural language processing, data mining system, systems reason with knowledge or summarization systems. All these systems firstly need a structured text to process it. Natural Language Processing Systems analyze morphological, syntactic and semantic structure of language. Expert Systems need well constructed knowledge bases. Even they may share this knowledge as ontology with other applications.

A general architecture of an Information Extraction System which is designed by Moens (2006) is seen in Figure 2.1. The system basically gets three inputs which are training corpus, source text and external knowledge. In the training phase, the knowledge engineer or the system acquires the necessary extraction patterns. In deployment phase IE system identifies and classifies relevant semantic information in new text. Finally system produces the structured output.

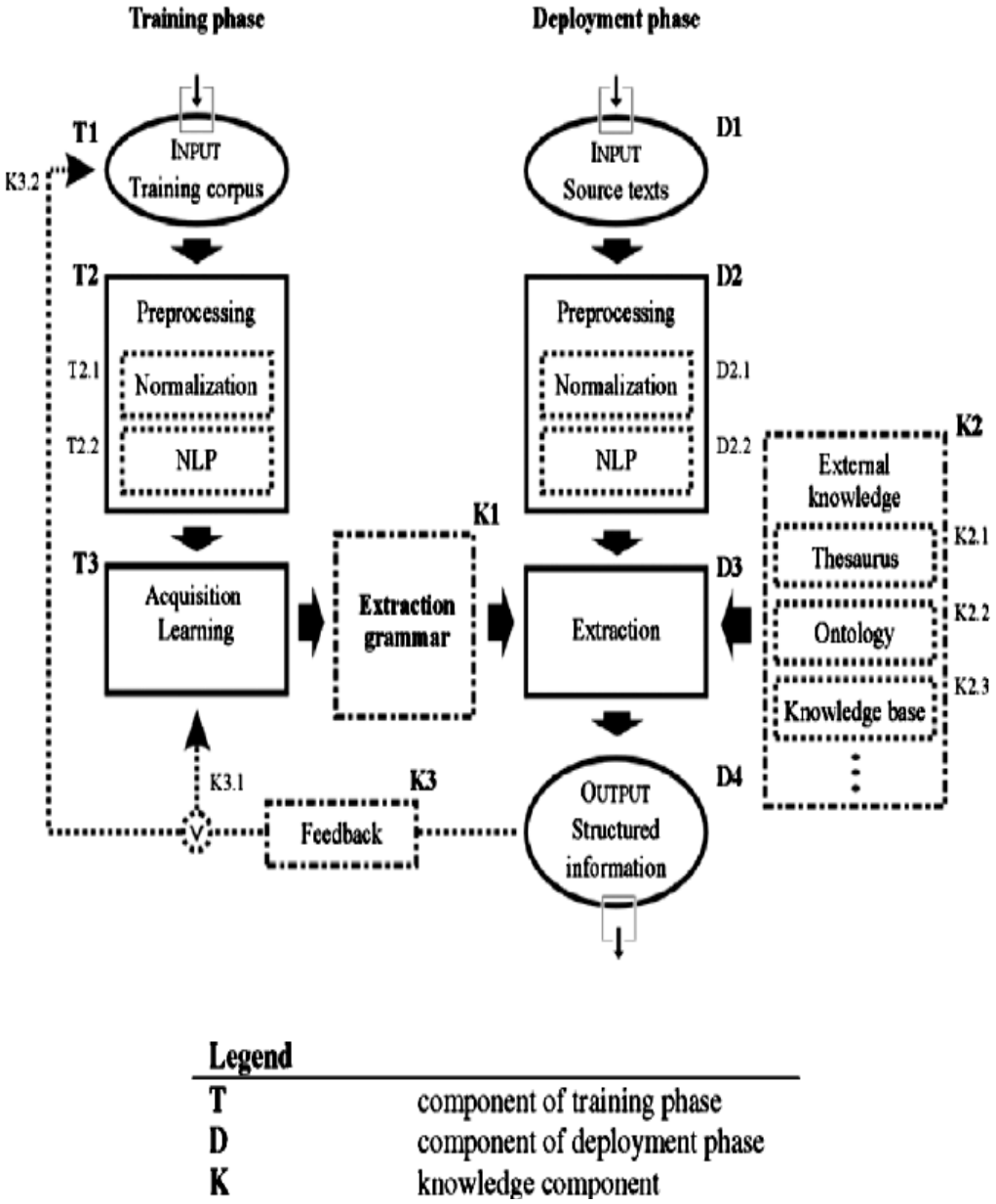


Figure 2.1 Typical Information Extraction System designed by Moens (2006).

In our information Extraction System, input set is the collection of e-commerce web pages. The processing part of our system creates Sequential Word Groups from e-commerce web pages and by extracting features and values of products from these e-commerce web pages, system constructs the Knowledge Base automatically. Finally, system shares the structured information with customers and other intelligent agent that may use the same ontology. Basic architecture of our Information Extraction System is seen in Figure 2.2.

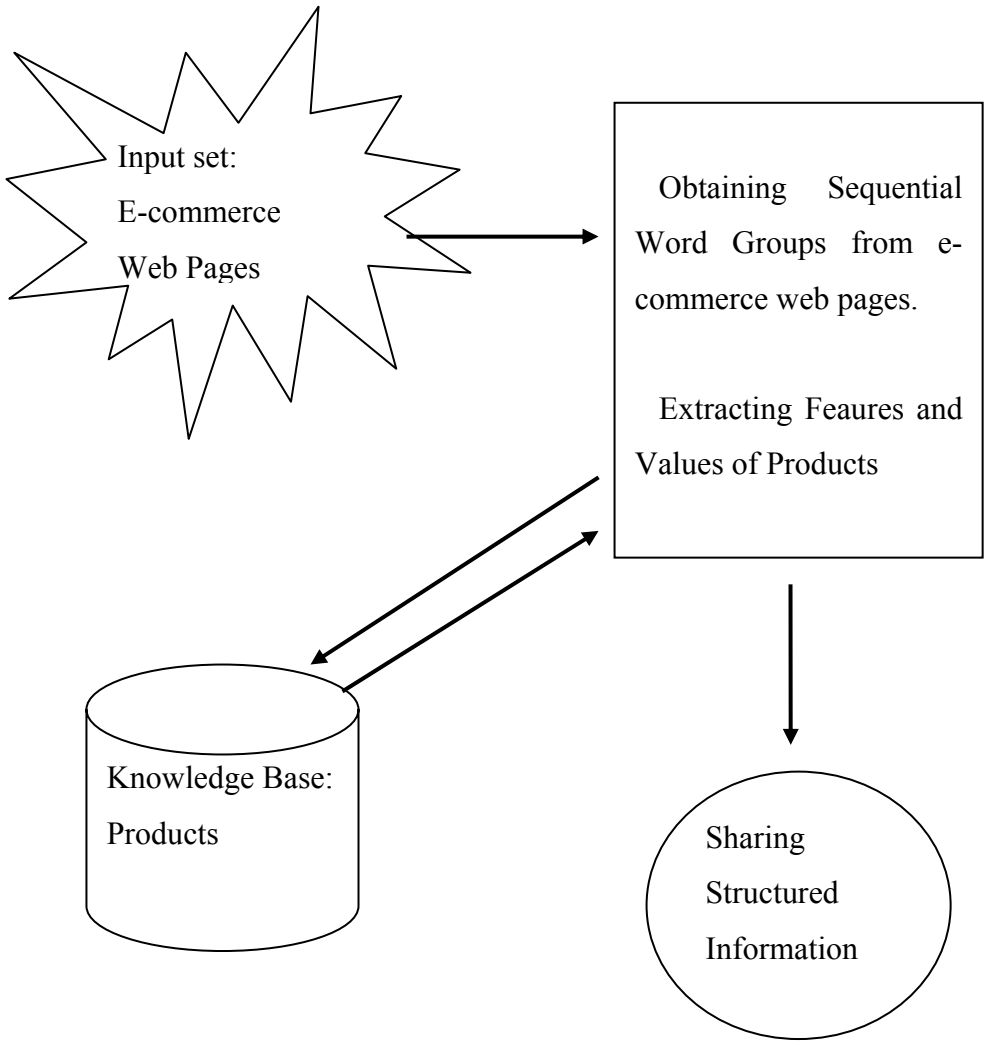


Figure 2.2 Basic architecture of our Information Extraction System.

Some of the information extraction tasks that we are interested in are named entity recognition, phrase (adjective or name) recognition, entity relation recognition. Named entity recognition is the process of finding and classifying the expressions that are labeled in a text. These labeled expressions can be product name, location name or a company name. Phrase recognition helps to seperate an adjective phrase or a noun phrases into adjectives and nouns. Entity relation recognition detects the relations between entities and these relations are represented in a semantic structure.

We use WWW as a knowledge discovery source and our Information Extraction system can be thought as converting information from text into database elements.

2.2 Pattern Recognition and Machine Learning Algorithms

Pattern recognition is the way of classifying the patterns into some groups based on a learned knowledge. Theodoridis and Koutroumbas (2003) define pattern recognition as classifying objects into a number of classes or categories based on the patterns that objects exhibit. In that respect, our use of pattern recognition would be in the field of Information Extraction. Because of Information Extraction is interested in detection and recognition of certain information, classified patterns are recognized as a set of features and their values. Of course the aim is classifying these patterns into semantic classes.

Recently feature vectors of Statistical and Machine Learning Algorithms are used to classify the objects or patterns into categories. Machine Learning algorithms change the orientations of knowledge acquisition from manual to automatic. Mostly, these kinds of algorithms like Bayesian classifiers, base on probability for information Retrieval applications.

Machine learning algorithms take an input which is the training data to be used for learning. Machine learning algorithms are mainly supervised, unsupervised, semi supervised, reinforcement, transduction and learning to learn algorithms. Supervised learning is domain specific learning method of machine learning but unsupervised

learning is generally implemented for domain independent works. Kohavi and John (1997) desribe the feature subset selection problem in supervised learning which involves identifying the relevant or useful features in a dataset and giving only that the subset to the learning algorithm.

In our approach, machine learning algorithm is applied over e-commerce web pages to learn the relations between products and features of products. Our algorithm may be applied on both domain specific and domain independent areas.

2.3 Knowledge Acquisition and Knowledge Base Consruction

Basically Knowledge Acquisition is obtaining, analyzing knowledge to represent it for the usage of a computer program. Knowledge is based on the idea when a human expert does for a problem solution.

Sowa (2000) define Knowledge Acquisition as the process of eliciting, analyzing and formalizing the patterns of thought underlying some subject matter. Sowa thinks the knowledge engineer must get the expert to articulate tacit knowledge in natural language for elicitation, In formalization the knowledge engineer must encode the knowledge elicited from the expert in rules and facts of some Artificial Intelligence language. Between those two stages lies conceptual analysis: the task of analyzing the concepts expressed in natural language and making their implicit relationships explicit. Lee and Yang (2000) define, Knowledge Acquisition as one of the 5 activities of Knowledge Management which are:

1. Knowledge Acquisition,
2. Knowledge Integration,
3. Knowledge Innovation,
4. Knowledge Protection,
5. Knowledge Dissemination.

Our purpose for this dissertation is to construct our Knowledge Base automatically for Knowledge Management. The input of this Knowledge base is obtained from WWW which may be entries from a database. E-commerce web sites mostly use a structured database working at the backside of interaction with users.

One way to represent the knowledge is using the frames. A frame is a structure that defines the classes of entities (relevant features) in a domain. We extract these features and values of these features from the text by using our algorithm that is based on Sequential Word Groups. A basic frame for a car is shown in Table 2.1.

Table 2.1 A basic frame for a car.

CAR	
FEATURE	VALUE
motor power (hp)	[60, 300]
fuel type	{benzene, diesel, lpg}
cylinder volume (cm3)	[800, 4500]
production date	[1950, 2007]
color	{ grey, beige, white, red, … , blue }

There are some knowledge representation languages like Web Ontology Language (OWL) or DARPA Agent Markup Language - Ontology Interface Layer (DAML-OIL) to code the frames by using Extensible Markup Language (XML). Most of the frames are constructed manually. This manually frame construction restrict the searching domain. To convert manual construction into automatic construction is expensive and huge amount of extra work.

In our study we try to construct product frames automatically by defining their features and values.

2.4 Recently Developed Systems for Information Extraction and Knowledge Acquisition

FERbot is a sytem which uses its own algorithm for Information Extraction and Knowledge Acquisition. Following works are the most related works to our purpose which are designed recently. The purposes of each work may overlap with the purposes of our approach but we use a different algorithm and different methods for extracting information from Internet.

Etzioni and Weld (1994) implemented an artificial intelligent agent called softbot (software robot) which allows a user to make a high level request and softbot searches and inferences knowledge to determine how to satisfy the request in Internet.

Perkowitz, Doorenbos, Etzioni, Weld (1997) published four categories of questions for automatic learning to provide helpful tools and to enable Artificial Intelligence Systems to scale with the growth of Internet.

1. Discovery: How does the learner find new and unknown information resources?
2. Extraction: What are the mechanics of accessing an information source and parsing the source?
3. Translation: Having parsed the response into tokens, how does the learner interpret the information in terms of its own concepts?
4. Evaluation: What is the accuracy, reliability, and scope of the information source?

According to answers of these questions, they describe two agents called ShopBot and Internet Learning Agent (ILA) for constructing autonomous Internet learning agent to discover and use information resources effectively. ShopBot is the agent that uses test queries to learn how to extract information from a web site. ShopBot autonomously learns how to shop at online stores. Internet Learning Agent (ILA) is

the agent that learns to translate from an information source's output to its own internal concepts through interaction with the source.

Rus and Subramanian (1997) present a customizable architecture for software agents that capture and access information in large, heterogeneous, distributed electronic repositories. The key idea is to exploit underlying structure at various levels of granularity to build high-level indices with task-specific interpretations. Information agents construct such indices and are configured as a network of reusable modules called structure detectors and segmenters. They illustrate their architecture with the design and implementation of smart information filters in two contexts: retrieving stock market data from Internet newsgroups, and retrieving technical reports from Internet ftp sites. Their main idea it to seperate a document into different segments which identify possibly-relevant fragments and analyse these fragments by structer detectors.

According to Smith, M.D. and E. Brynjolfsson (2001), shopbots collect and display information on a variety of product characteristics, list summary information for both well- and lesser-known retailers, and typically rank the retailers based on a characteristic of interest to the shopper such as price or shipping time. The resulting comparison tables reveal a great deal of variation across retailers in relative price levels, delivery times, and product availability.

Data Mining is the way of Discovering Knowledge from databases. Kohavi and Provost (2001) define five desired situations for an application of data mining to e-commerce.

1. Data with rich descriptions.
2. A large volume of data.
3. Controlled and reliable data collection
4. The ability to evaluate results.
5. Ease of integration with existing processes.

Ansari, Kohavi, Mason, and Zheng (2000) proposed an architecture that successfully integrates data mining with an e-commerce system. The proposed architecture consists of three main components: Business Data Definition, Customer interaction, and Analysis, which are connected using data transfer bridges. This integration effectively solves several major problems associated with horizontal data mining tools including the enormous effort required in preprocessing of the data before it can be used for mining, and making the results of mining actionable.

Crescenzi, Mecca and Merialdo (2001) investigate techniques for extracting data from HTML sites through the use of automatically generated wrappers. To automate the wrapper generation and the data extraction process, they develop a novel technique to compare HTML pages and generate a wrapper based on their similarities and differences. Their software RoadRunner tokenizes, compares and aligns the HTML token sequences tag by tag.

Arasu and Garcia-Molina (2003) study the problem of automatically extracting the database values from such template-generated web pages without any learning examples or other similar human input. They formally define a template, and propose a model that describes how values are encoded into pages using a template. Their algorithm that takes, as input, a set of template-generated pages, deduces the unknown template used to generate the pages, and extracts, as output, the values encoded in the pages.

Chang, Hsu and Lui (2003) propose a pattern discovery approach to the rapid generation of information extractors that can extract structured data from semi structured Web documents. They introduce a system called IEPAD (an acronym for Information Extraction based on PAttern Discovery), that discovers extraction patterns from Web pages without user-labeled examples. Their system applies pattern discovery techniques, including PAT-trees, multiple string alignments and pattern matching algorithms. Extractors generated by IEPAD can be generalized over unseen pages from the same Web data source.

Perrin, Petry, (2003) present an algorithm that systematically extracts the most relevant facts in the texts and labels them by their overall theme, dictated by local contextual information. It exploits domain independent lexical frequencies and mutual information measures to find the relevant contextual units in the texts. They report results from experiments in a real-world textual database of psychiatric evaluation reports.

Kang and Choi, (2003) developed MetaNews which is an information agent for gathering news articles on the Web. The goal of MetaNews is to extract news articles from periodically updated online newspapers and magazines. More specifically, MetaNews collects HTML documents from online newspaper sites, extracts articles by using the techniques of noise removal and pattern matching, and provides the user with the titles of extracted articles and the hyperlinks to their contents.

Flesca, Manco, Masciari, Rende and Tagarelli (2004) review the main techniques and tools for extracting information available on the Web, devising a taxonomy of existing systems.

The objective of Chan (2004) work is to contribute to the infrastructure needed for building shared, reusable knowledge bases by constructing a tool called the Knowledge Modeling System that facilitates creation of a domain ontology. The ontology of a system consists of its vocabulary and a set of constraints on the way terms can be combined to model a domain.

Rokach Chizi and Maimon (2006) define the feature selection as the process of identifying relevant features in the data set and discarding everything else as irrelevant and redundant. Their study presents a general framework for creating several feature subsets and then combines them into a single subset.

Ahmed, Vadrevu and Davulcu (2006) improved a system called Datarover which can automatically crawl and extract all products from online catalogs. Their system is based on pattern mining algorithms and domain specific heuristics which utilize the

navigational and presentation regularities to identify taxonomy, list of product and single product segments within an online catalog. Their purpose is to transform an online catalogue into database of categorized products.

Rokach, Romano, Chizi and Maimon (2006) examine a novel decision tree framework for extracting product attributes. Their algorithm has three stages. First, a large set of regular expression-based patterns are induced by employing a longest common subsequence algorithm. In the second stage they filter the initial set and leave only the most useful patterns. Finally, they present the extraction problem as a classification problem and employ an ensemble of decision trees for the last stage.

Chang, Kayed, Girgis and Shaalan (2006) survey the major Web data extraction approaches and compares them in three dimensions: the task domain, the automation degree, and the techniques used. The criteria of the first dimension explain why an IE system fails to handle some Web sites of particular structures. The criteria of the second dimension classify IE systems based on the techniques used. The criteria of the third dimension measure the degree of automation for IE systems. They believe these criteria provide qualitatively measures to evaluate various IE approaches.

CHAPTER THREE
INTELLIGENT AGENTS AND THEIR ROLES IN ELECTRONIC COMMERCE

3.1 Electronic Commerce

Electronic Commerce (e-commerce) is the way of advertising, buying, selling, marketing, and servicing products and services through electronic systems. These systems can be Internet as well computer networks. In general, there are two types of e-commerce, Business to Business (B2B) and Business to Customer (B2C). In this thesis, we will be interested in with the Business to Costumer part of e-commerce. In B2C e-commerce, customers directly use Internet especially Web to manage their main activities such as searching product, ordering or payment.The trade cycle that is used by Whiteley (2000) for e-commerce is seen at Table 3.1.

Our study will take a place at the pre-sale part of trade cycle where pre-sale is the first step of the trading cycle consisting of searching and negotiation.

Table 3.1 Electronic Markets and Trade Cycle.

Search	Pre-sale
Negotiate	
Order	Execution
Deliver	
Invoice	Settlement
Payment	
After Sales	After Sale

The primary goal of B2C e-commerce is to facilitate various commercial actions, such as product brokering, merchant brokering, and negotiation of the price or other terms of trade. Addition to this idea, Guttman, Moukas, and Maes (1998) defines a

model called “Consumer Buying Behaviour (CBB)” to capture consumer behaviour. There are six stages in the CBB model:

1. Need identification
2. Product brokering
3. Merchant brokering
4. Negotiation
5. Purchase and delivery
6. Product service and evaluation.

From the CBB model perspective, agents can act as mediators in consumer behavior in buying commodities in three stages: product brokering, merchant brokering, and negotiation. One of the important areas of the CBB is product brokering. At this stage agents can determine what to buy according to different product information. The main techniques in this stage are feature-based filtering, collaborative filtering, and constraint based filtering.

According to Leung and He (2002), at Feature-based filtering technique, given the feature keywords to be retrieved, the technique returns only those texts that contain the keywords. At Collaborative filtering technique, the idea is to give personalized recommendations based on the similarities between different user’s preference profiles. At Constraint-based filtering technique, a customer specifies constraints, to narrow down the products, until the best one satisfying his need is found. The differences among these techniques are summarized in Table 3.2 from five dimensions.

In our study we consider that customers certainly know the product they buy or they certainly know the product that they want to be informed about. For that reason, our intelligent agent FERbot uses feature based filtering for product brokering. The difference from other agents using feature based filtering, is bringing the product features and values as a result of query.

Table 3.2 Comparison of different Product brokering techniques (Leung and He, 2002).

	Feature-based	**Collaborative**	**Constraint-based**
When to use the technique	Know exactly user's needs	Not know what to buy	Know exactly user's needs
Requirements of the technique	Feature keywords for the goods	Profiles of users	The conditions that the goods satisfy
Interaction with users	A lot	Few	A lot
Results Returned	Goods satisfying required features	Suggestion of what to buy	Goods satisfying Specified Constraints
Goods type it suits for	More kinds of goods	Books, D/VCDs and so on	More kinds of goods

3.2 Intelligent Agents

If a buyer doesn't aware of shopping agents, one way finding information about a product is to use a search engine like "Google", "Yahoo", "Lexibot" and "Mamma". Each of them has some advantages and disadvantages. Web crawlers and spiders of these search engines visit Internet pages from links included by another page. They summary and index the pages into massive databases.

Google returns thousand pages including short summaries of those web pages according to query [http://www.google.com]. Yahoo is a kind of directory search engine categorizing all subjects [http://www.yahoo.com]. Lexibot is a deep web search engine searching web not like surface web search engine. It finds out deeper parts of web sites [http://www.lexibot.com]. Mamma falls into the "meta search engine" class. It returns a combination of several search engine results [http://www.mamma.com].

Sometimes definitions of agent, intelligent agent, e-commerce agent, crawlers and search engines are not so clear for people. Because, all these terms may overlap according to their definitions and for their usage.

Smith, Cypher and Spohrer (1994) define an agent as a persistent software entity dedicated to a specific purpose. “Persistent” distinguishes agents from subroutines; agents have their own ideas about how to accomplish tasks, their own agendas. “Special purpose” distinguishes them from entire multifunction applications; agents are typically much smaller.

Etzioni and Weld (1995) think that an agent enables a person to state what information he requires; the agent determines where to find the information and how to retrieve it.

Jennings and Wolldridge (1998) defines an agent as a hardware or (more usually) software entity with (some of) the following characteristics:

- Ongoing execution: Unlike usual software routines, which accomplish particular tasks and then terminate, agent function continuously for a relatively long period of time.
- Environmental awareness: Agents act on the environment where they are situated in two ways: (1) Reactivity: agents sense and perceive the environment through sensors, and react, through effectors, in a timely fashion to change the environment. (2) Proactiveness: Agents also are able to exhibit opportunistic, goal-directed behaviours and take the initiative where appropriate.
- Agent awareness: Agents may model other agents, reason about them, and interact with them via communication/coordination protocols in order to satisfy their design objective and to help one another with their activities.
- Autonomy: When events, anticipated or unanticipated, occur in open, nondeterministic environments, an agent is able to independently determine and carry out some set of actions without direct intervention from humans (or other agents). That is, it can exercise control over its own actions and internal states.
- Adaptiveness: Over time, based on their experiences, agents are able to adapt their behaviours to suit the preferences and behaviours of various users.

- Intelligence: Agents may embody sophisticated AI techniques, for instance, probabilistic reasoning, machine learning, or automated planning.
- Mobility: Agents may roam on the Internet according to their own decisions.
- Anthropomorphism: Agents may exhibit human-like mental qualities; they respond to queries about their beliefs or obligations, convey understanding or confusion through icons depicting facial expressions, or display an animation that connotes fear or friendliness.
- Reproduce: Agents may be able to reproduce themselves.

Feldman and Yu (1999) define an intelligent agent as the one which can learn the patterns of behavior, or the rules regarding certain actions and transactions, and then act appropriately on behalf of its boss.

Agents stand for different activities of different disciplines. The main things that agents can do for e-commerce systems can be grouped into four groups by Leung and He (2002).

1. Agents can monitor and retrieve useful information, and do transactions on behalf of their owners or analyze data in the global markets.
2. Agents can try to find the best deal for the customer with less cost, quicker response, and less user effort.
3. Agents can make rational decisions for humans, and negotiate the price of the trade with peer agents strategically.
4. Agents can also manage the supply chain network of a company at a low cost.

Kalakota and Whinstone (1996) think that the most important roles of agents are information gathering and filtering, and decision making.

As seen from above definitions, one part of e-commerce agents deal with the subject of information gathering which is the act of collecting information. This is the main idea of our study which is collecting information from Web and extracting meaningful relations between products and their properties to inform customers. For

that reason, FERbot will comply with the set of abilities given in the first group that is defined by Leung and He (2002) at the above.

3.3 Classification of Agents

Classification of agents may be done according to their general characteristics. Franklin and Graesser (1996) classify agents according to the subset of the properties in Table 3.3 that they enjoy. According to their definition, every agent satisfies the first four properties. Our agent FERbot satisfies the property of “Learning” additionaly. For that reason, it is considered as an intelligent agent.

Table 3.3 Classification of agents according to Franklin and Graesser (1996).

Property	**Other Names**	**Meaning**
Reactive	sensing and acting	Responds in a timely fashion to changes in the environment.
Autonomous		Exercises control over its own actions.
Goal-oriented	Pro-active, Purposeful	Does not simply act in response to the environment.
Temporally continuous		Is a continuously running process
Communicative	Socially able	Communicates with other agents, perhaps including people.
Learning	Adaptive	Changes its behavior based on its previous experience.
Mobile		Able to transport itself from one machine to another.
Flexible		Actions are not scripted.
Character		Believable "personality" and emotional state.

Another agent classification is made by Nwana (1996) in Figure 3.1.Two of the agent types that FERbot falls in are Information Agents and Smart Agents. Information agents have varying characteristics: they may be static or mobile; they may be non-cooperative or social; and they may or may not learn (Nwana 1996). The most important point from this definition is the character of an agent being learning or not learning. By the ability of learning, an agent becomes an intelligent agent.

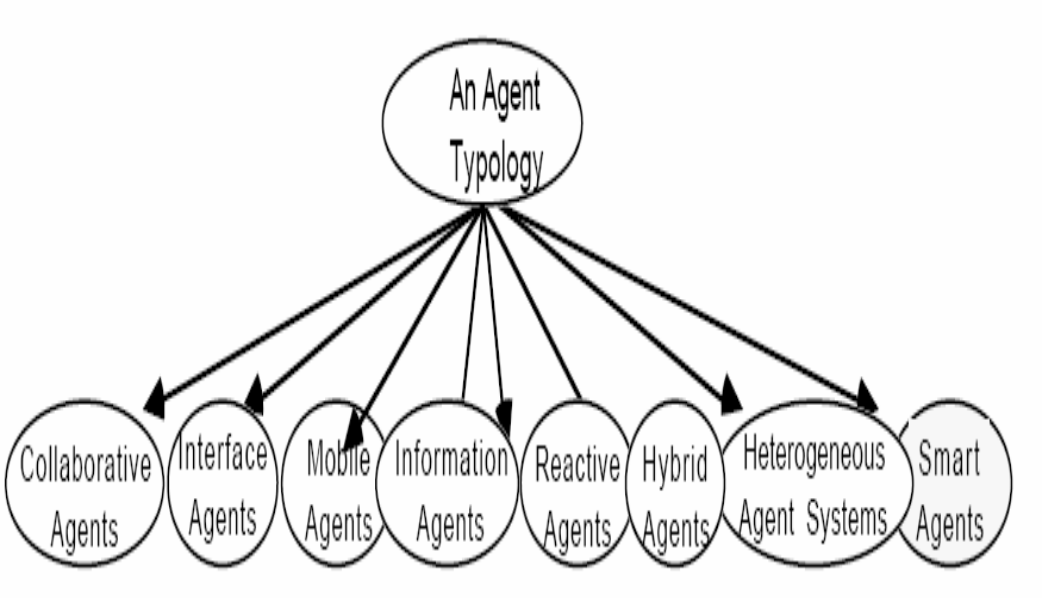


Figure 3.1 Classification of Software Agents (Nwana, 1996).

According to the classification in Figure 3.1, FERbot is both an Information agent and Smart agent. Agents may sometimes be classified by their roles (preferably, if the roles are major ones), e.g. World Wide Web information agents (Nwana, 1996). This category of agents usually exploits Internet search engines such as WebCrawlers, Lycos and Spiders. Essentially, they help manage the vast amount of information in wide area networks like the Internet. This kind of agents is called as information or Internet agents. Information agents may be static, mobile or deliberative. Clearly, it is also pointless making classes of other minor roles as in report agents, presentation agents, analysis and design agents, testing agents, packaging agents and help agents or else, the list of classes will be large.

Sometimes agents fall into the same group with crawlers. According to Answers Corporation [http://www.answers.com], a crawler is defined as follows: “Crawler also known as a web crawler, spider, ant, robot (bot) and intelligent agent, a crawler is a program that searches for information on the Web”. It is used to locate HTML pages by content or by following hypertext links from page to page. Search engines use crawlers to find new Web pages that are summarized and added to their indexes. Web robots, spiders, and crawlers, collectively called bots, are automated programs that visit websites (Heaton, 2002).

Bots are mainly used by the areas of data mining, e-commerce, games, Internet chatting, mailing etc. Mainly there are two types of Internet bots. First group of Internet bots are useful software programs that perform some specific tasks automatically over Internet. Crawler or wrappers are in the first group of Internet bots. A Web crawler is a type of Internet bot used by Web search engines, such as Google and Excite, to surf the web and methodically catalog information from Web sites (http://www.mysecurecyberspace.com/encyclopedia/index/Internet-bot.html).

Second group Internet bots are dangerous software programs that can be installed into person computer to do some jobs without taking permission from the person. In the case of E-commerce, a malicious programmer can design a bot whose task is to aggregate prices from other E-commerce sites (Rui and Liu, 2004). According to Rui and Liu, based on the collected prices, the malicious programmer can make his/her price a little cheaper, thus stealing away other sites’ customers. For that reason sometimes these kinds of bots are useful for costumers but are not good for some e-commerce owners.

3.4 Intelligent Agent Design Issues

Before designing an agent program, there must be a good idea of possible percepts and actions, what goals or performance measure the agent is supposed to achieve, and what sort of environment it will operate in. For the acronymically minded, it is called as PAGE (Percept, Actions, Goals, Environment) description (Russell and

Norvig, 1995). Figure 3.4 shows the basic elements for a selection of agent types. So, according to the PAGE definition, our agent FERbot can be basically structured as shown in Table 3.5:

Table 3.4 Examples of agent types and their PAGE descriptions.

Agent Type	Percepts	Actions	Goals	Environment
Medical diagnosis system	Symptoms, findings, patient's answers.	Questions, tests, treatments.	Healthy patient, minimize costs.	Patient, hospital.
Satellite image analysis system	Pixels of varying intensity, color.	Print a categorization of scene.	Correct categorization.	Images from orbiting satellite.
Part-picking robot	Pixels of varying intensity.	Pick up parts and sort into bins	Place parts in correct bins	Conveyor belt with parts.
Refinery controller	Temperature, pressure readings.	Open, close valves; adjust temperature.	Maximize purity, yield, safety	Refinery.
Interactive English tutor	Typed words	Print exercises, suggestions, corrections.	Maximize student's score on test.	Set of students.

Table 3.5 FERbot and its PAGE description.

Agent Type	E-commerce Agent (intelligent agent, shopping agent, information agent, bot).
Percepts	Web Pages (commercial sites), Customer Queries.
Actions	Information gathering, discovering knowledge, filtering, indexing, analyzing, modeling, constructing knowledge base.
Goals	To satisfy the customer needs by the way of defining the features and values of a product.
Environment	Internet, customers.

Against to the advent of the Web, Doorenbos, Etzioni and Weld, (1997) defines some fundamental questions for the designers of intelligent software agents. They are:

- Ability: To what extent can intelligent agents understand information published at Web sites?
- Utility: Is an agent's ability great enough to provide substantial added value over a sophisticated Web browser coupled with directories and indices such as Yahoo, Lycos and Google?
- Scalability: Existing agents rely on a hand-coded interface to Internet services and Web sites. Is it possible for an agent to approach an unfamiliar Web site and automatically extract information from the site?
- Environmental Constraint: What properties of Web sites underlie the agent's competence? Is “sophisticated natural language understanding” necessary? How much domain-specific knowledge is needed?

According to our agent design, the answers that can be given for these fundamental questions are: FERbot is able to understand values and features of products. FERbot uses Googles search engine at back side to restrict the size of the web pages. FERbot may extract information from the unfamiliar web sites automatically. This is the main advantage of FERbot. FERbot works on domain specific and domain independent data.

The implementation tools for e-commerce agents, according to Leung and He (2002), can be summarized in Table 3.6.

Table 3.6 Tools for agent-mediated e-commerce (Leung and He, 2002).

<table>
<tr><td rowspan="5">Communication Protocols and Languages</td><td>TCP/IP</td><td colspan="2">A common transport protocol which allows information transfer through the Internet</td></tr>
<tr><td>EDI</td><td colspan="2">An information exchanging protocol for interbusiness transactions</td></tr>
<tr><td>HTML</td><td colspan="2">Data format language which enables browser software to interpret pages on the WWW</td></tr>
<tr><td>XML</td><td colspan="2">Data content meta-language which provides a file format for representing data, a schema for describing data structure and a mechanism for extending and annotating HTML semantic information</td></tr>
<tr><td>XML Based E-com. Specific.</td><td colspan="2">XML/EDI, ICE, OBI, SET, OPT, OFE, ...</td></tr>
<tr><td rowspan="2">InterAgent Languages</td><td>Commu. Languages</td><td>KQML, FIPA, ACL, MIL</td><td rowspan="2">Content Language provides a syntax or semantics for the content of the message and different content languages correspond to different domains</td></tr>
<tr><td>Content Language</td><td>KIF, FIPA, SL, MIF & Ontoligua</td></tr>
<tr><td colspan="2">Agent Developing Tools and Tool-Kits</td><td colspan="2">ZEUS, Agent Builder, Agent Building Environment, Agent Building Shell, Agent Development Environment, Multi-Agent System Tool, ...</td></tr>
<tr><td colspan="2">Mobile Agent Platforms</td><td>All of these platforms Voyager, Concordia, Odyssey, Aglets, Jumping Beans, ...</td><td>All of these platforms are java-based</td></tr>
</table>

3.5 FERbot Instead of Wrappers

FERbot is a kind of intelligent information agent that behaves like wrappers. It works over unstructured web pages. If semi structured web pages are the input set of FERbot, the performance of FERbot becomes excellent. FERbot can be thought as a wrapper but, by the way of using its own algorithm which is different than the traditional algorithms, FERbot differs from other wrappers that extract information from XML web pages.

Information agents generally rely on wrappers to extract information from semi structured web pages (a document is semi structured if the location of the relevant information can be described based on a concise, formal grammar). Each wrapper consists of a set of extraction rules and the code required to apply those rules (Muslea, Minton and Knoblock, 1999).

According to Seo, Yang and Choi (2001), the information extraction systems usually rely on extraction rules tailored to a particular information source, generally called wrappers, in order to cope with structural heterogeneity inherent in many different sources. They define a wrapper as a program or a rule that understands information provided by a specific source and translates it into a regular form that can be reused by other agents.

The main issue that is raised commonly from many applications such as online shopping and meta-search systems is how to integrate semi structured and heterogeneous Web information sources and provide a uniform way of accessing them (Kang and Choi, 2003).

Classification of web data extraction of Baumgartner, Eiter, Gottlob and Herzog (2005) as follows:

- Languages for wrapper development.
- HTML-aware tools.
- NLP-based (Natural Language Processing) tools.
- Wrapper induction tools.
- Modeling-based tools.
- Ontology-based tools.

Muslea (1999) surveys the various types of extraction patterns that are generated by machine learning algorithms. He identifies three main categories of patterns, which cover a variety of application domains, and he compares and contrasts the patterns from each category.

Extraction rules for Information Extraction from free text are based on syntactic and semantic constraints that help to identify the relevant information within a document. In order to apply the extraction patterns, the original text is pre-processed with a syntactic analyzer and a semantic tagger. Some systems for constructing extraction patterns are AutoSlog, LIEP, PALKA, CRYSTAL, and HASTEN.

AutoSlog (Riloff 1993) builds a dictionary of extraction patterns that are called concepts or concept nodes.

LIEP (Huffman 1995) is a learning system that generates multi-slot extraction rules.

The PALKA system (Kim & Moldovan 1995) learns extraction patterns that are expressed as frame-phrasal pattern structures (for short, FP-structures).

CRYSTAL (Soderland et al. 1995) generates multi-slot concept nodes that allow both semantic and exact word constraints on any component phrase. Webfoot (Soderland 1997) is a preprocessing step that enables CRYSTAL to also work on Web pages: the Web pages are passed throughWebfoot, which uses page layout cues

to parse the pages into logically coherent text segments. Then one can apply CRYSTAL to text segments as if they were sentences coming from a free text document.

Also, The extraction patterns generated by HASTEN (Krupka 1995) are called Egraphs, and they can be seen as lists of (SemanticLabel, StructuralElement) pairs. Eventhoug all these systems use syntactic and semantic constraints to identify the items of interest; there are several important differences between them as shown in Table.3.7.

Table 3.7 Systems for constructing extraction patterns from free text and their differences.

System	**Granularity of Extraction**	**Semantic Constraints**	**Single slot rules or Multi slot rules**
AutoSlog,	Determining syntactic fields with target phrase.	Semantic class constraints only onto slots to be extracted.	Generating single slot rules
LIEP	Identifying exact phrase of interest	Semantic class constraints only onto slots to be extracted.	Generating multi slot rules
PALKA	Determining syntactic fields with target phrase.	Semantic class constraints only onto slots to be extracted.	Generating single and multi slot rules
CRYSTAL	Determining syntactic fields with target phrase.	Semantic class constraints on subject and preposional phrase.	Generating single and multi slot rules
HASTEN	Identifying exact phrase of interest	Semantic class constraints only onto slots to be extracted.	Generating single and multi slot rules

In order to handle Information Extraction from online documents, Muslea (1999) presents the three types of extraction rules for combining syntactic/semantic constraints with delimiters that "bound" the text to be extracted. Some of the systems for Information Extraction from Online Documents are WHISK, RAPIER and SRV.

WHISK (Soderland 1998) is a learning system that generates extraction rules for a wide variety of documents ranging from rigidly formatted to free text.

The RAPIER system (Califf and Mooney 1997) learns singleslot extraction patterns that use limited syntactic information (e.g., the output of a part-of-speech tagger).

Semantic class information SRV (Freitag 1998) generates first-order logic extraction patterns that are based on attribute-value tests and the relational structure of the documents.

The systems WHISK, RAPIER and SRV that extracts patterns from online documents and their main differences are represented in Table 3.8.

Table 3.8 Systems for Constructing Extraction Patterns from online documents and their differences.

System	**Number of constraints**	**If Length of phrase changes dramatically from document to document**	**Single slot rules or Multi slot rules**
WHISK	Impose set of constraints	May not extract either too many or too few words	Generating single and multi slot rules
RAPIER	Imposes a richer set of constraint	May extract either too many or too few words	Generating single slot rules
SRV	Imposes a richer set of constraint	May extract either too many or too few words	Generating single slot rules

In order to apply the extraction patterns FERbot processes the input text with syntactic and semantic tagging. Input set of FERbot is the collection of web pages. In this point, this study is about information extraction from documents but after defining features and values of products in fact they refer to extraction patterns, FERbot certainly extracts information from online documents. FERbot identifies exact phrase of interest. FERbot generates single or multi slot rules. FERbot allows semantic class constraints only onto slots to be extracted. FERbot may extract too many or too few words.

Knoblock, Lerman, Minton and Muslea (2000) defines a wrapper induction system as shown in Figure 3.2, where the wrapper induction system takes a set of web pages labeled with examples of the data to be extracted. The user provides the initial set of labeled examples and the system can suggest additional pages for the user to label in order to build wrappers that are very accurate. The output of the wrapper induction system is a set of extraction rules that describe how to locate the desired information on a Web page. After the system creates a wrapper, the wrapper verification system uses the functioning wrapper to learn patterns that describe the data being extracted. If a change is detected, the system can automatically repair a wrapper by using the same patterns to locate examples on the changed pages and re-running the wrapper induction system.

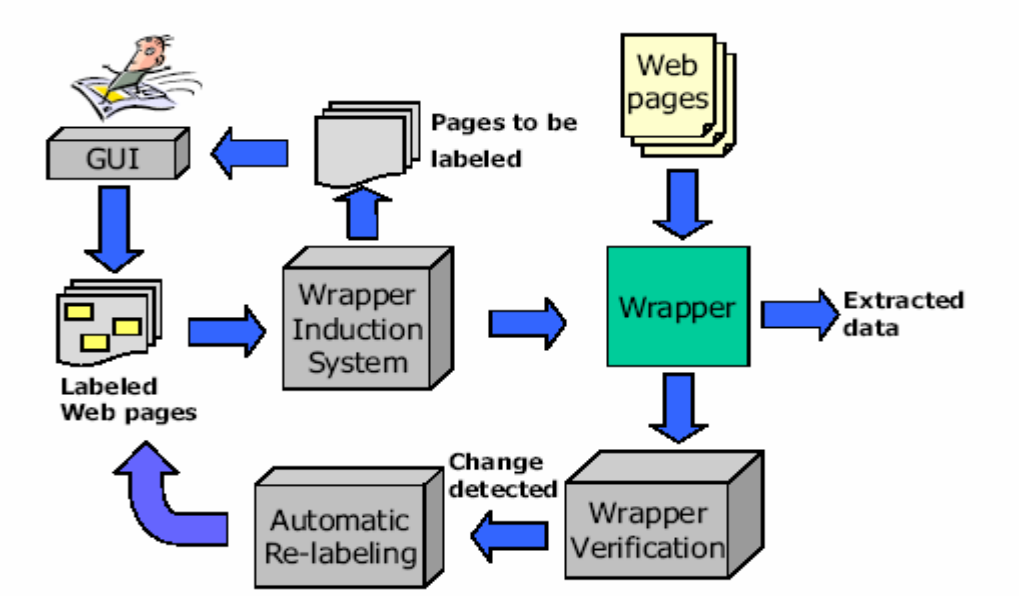


Figure 3.2 Lifecycle of a wrapper (Knoblock, Lerman, Minton and Muslea 2000).

Yang, Oh, Doh and Choi (2002) have presented an intelligent Web information extraction system XTROS that represents the domain knowledge and the wrapper by XML documents, and automatically generates wrappers from semi structured labeled documents. One of XTROS's current limitations is that it only works for the labeled documents, not functioning with non-labeled ones such as table-type descriptions. (Yang, Oh, Doh and Choi, 2002)

As seen in Figure 3.2, extracted patterns that will be extracted are labeled by XML within ontology. For that reason if a wrapper is constructed to extract information from this kind of labeled document, the bold colored words as seen in Figure 3.2 would be extracted. The main disadvantage of wrappers is to be applied on just labeled documents.

Figure 3.3 shows the wrapper that is constructed for semi structured domains like http://www.homes.com. These kinds of wrappers may easily extract information from semi structured web pages because, these wrappers are constructed for domain specific resources and they work on labeled documents.

To make a knowledge-based information-extraction system to be more general enough to handle both labeled and unlabeled documents, Yang and Choi (2003) propose an enhanced scheme of knowledge-based wrapper generation by using token-based morphological patterns. Each document is represented as a sequence of tokens, rather than a sequence of logical lines, in order to capture the meaning of data fragments more correctly and recognize the target information contextually.

```
<KNOWLEDGE>

   <OBJECTS>
      <OBJECT>PRICE</OBJECT>
      <OBJECT>BED</OBJECT>
      <OBJECT>BATH</OBJECT>
      <OBJECT>CITY</OBJECT>
      <OBJECT>MLS</OBJECT>
      <OBJECT>DETAIL</OBJECT>
      <OBJECT>IMG</OBJECT>
   </OBJECTS>

   <PRICE>
      <ONTOLOGY>
         <TERM>PRICE</TERM>
         <TERM>$</TERM>
      </ONTOLOGY>
      <FORMAT>
         <FORM>[ONTOLOGY]DIGITS</FORM>
         <FORM>DIGITS[ONTOLOGY]</FORM>
      </FORMAT>
   </PRICE>

   <BED>
      <ONTOLOGY>
         <TERM>BEDROOMS</TERM>
         <TERM>BEDROOM</TERM>
         <TERM>BEDS</TERM>
         <TERM>BED</TERM>
         <TERM>BR</TERM>
         <TERM>BD</TERM>
      </ONTOLOGY>
      <FORMAT>
         <FORM>[ONTOLOGY]DIGITS</FORM>
         <FORM>DIGITS[ONTOLOGY]</FORM>
      </FORMAT>
   </BED>
Omitted.

…..
…..
</KNOWLEDGE>
```

Figure 3.2 A section from XML-based domain knowledge for the real estate domain (Yang, Oh, Doh and Choi, 2002).

```
<WRAPPER>

   <PRICE>
      <ONTOLOGY>$</ONTOLOGY>
      <Ident>NULL</Ident>
      <Format>Digits</Format>
      <Operation>Ontology*Format</Operation>
   </PRICE>

   <BED>
      <ONTOLOGY>BR</ONTOLOGY>
      <Ident>NULL</Ident>
      <Format>Digits</Format>
      <Operation> Format*Ontology </Operation>
   </BED>

   <BATH>
      <ONTOLOGY>BA</ONTOLOGY>
      <Ident>NULL</Ident>
      <Format>Digits</Format>
      <Operation> Format*Ontology </Operation>
   </BATH>

Omitted.
…..
…..
</WRAPPER>
```

Figure 3.3 A section of the wrapper for homes from http://www.homes.com (Yang, Oh, Doh and Choi, 2002).

Procedure for the wrapper generator that produces IE rules is depicted in Figure 3.4. The wrapper generator consults the domain knowledge to produce a wrapper through learning from training examples. In the preprocessing phase, training Web pages are converted into a sequence of tokens with morphological tags. In the rule-induction phase, candidate IE rules are produced by analyzing the tokens with the features in the domain knowledge. The generated rules are evaluated and represented in the XML form.

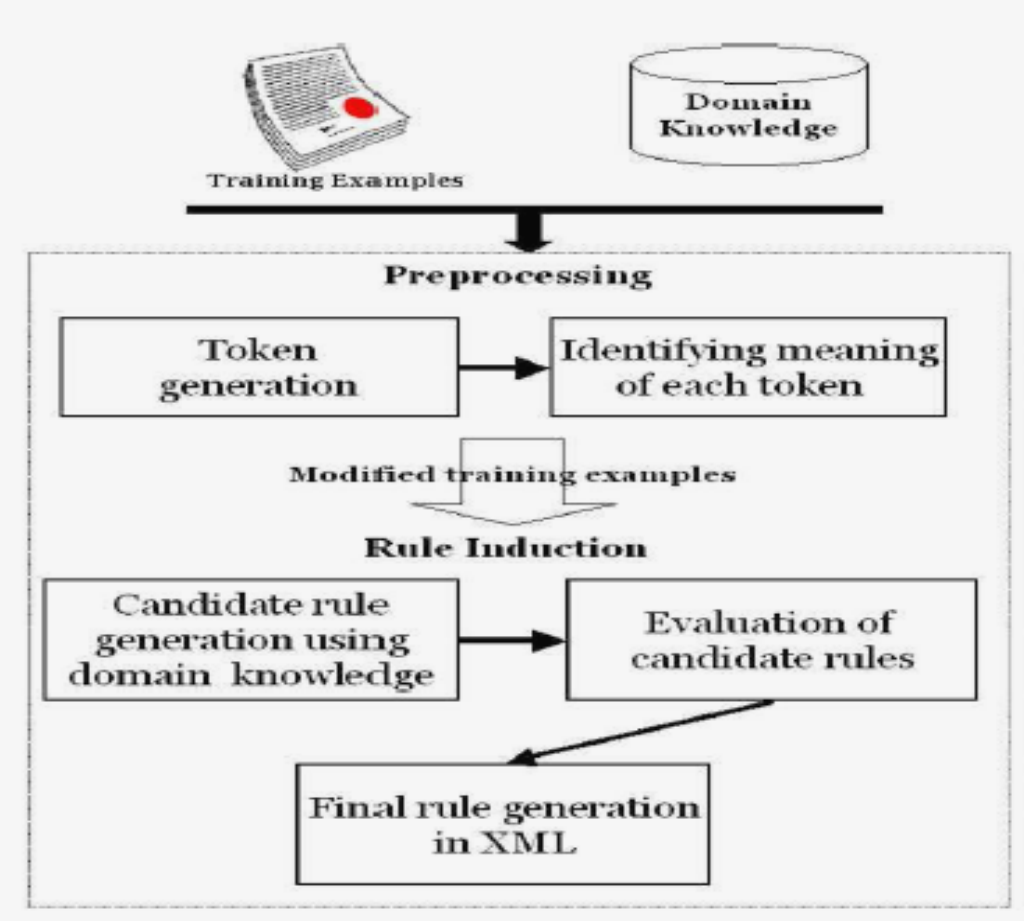


Figure 3.4 Overall procedure for generating information extraction rules (Yang and Choi, 2003).

FERbot is a kind of information extraction system which has a same purpose as wrappers have. The first advantage of FERbot is its own algorithm. It can be used for domain specific or domain independent resources. Second advantage is working with the labeled or unlabeled web pages. The input set of FERbot can be any kind of web pages whether it is not designed by XML. FERbot understands the features of

products by finding their frequencies. FERbot uses labels to extract the keywords of web pages to construct the keyword-document matrix for refining the results of features of products. Finally, we may say that most of the wrappers are constructed for domain dependent and labeled documents but FERbot does not imply these kinds of restrictions.

CHAPTER FOUR
FERbot (FEATURE EXTRACTOR RECOGNIZER BOT)

4.1 Problem Definition

Every product is defined by its features. An advertisement about a product in an e-commerce site firstly includes price of that product and some additional information like size, brand, power and etc. Because every e-commerce web page does not include the same standard notation for information about same product, it will be interesting to define the general characteristics of a product. Every product has some features and every feature has some values. For example, a car has some features like color, fuel type and motor power, etc. A color feature of a car has some values like red, blue and white.

Semantic relations of these features and values can occur between nouns, nouns and adjectives, nouns and verbs or verb adverbs. Level of a relation in a text may be seen in word-level, phrase level (noun-verb) or concept level. Relation between values and features and relation between features and products are similar, and it is the relation Has.

The input set of this study is the set of web pages, $WP = \{wp_1, wp_2, \ldots, wp_n\}$, from e-commerce web sites in Internet. The purpose is to produce the output which is the definition of any product in terms of possible features and values. If we denote a product by P and if F_i and V_j refer to the i^{th} feature and j^{th} value, respectively, we define a product as a set of features, $P = \{F_1, \ldots, F_n\}$ and a feature as a set of values,

$F_i = \{V_{i_1}, \ldots, V_{i_m}\}$, $i, j, n, m \in Z^+$ Finally, a product can be represented as

$P = \{ F_1 = \{V_{1_1}, \ldots, V_{1_k}\}, \ldots, F_n = \{V_{n_1}, \ldots, V_{n_t}\} \}$, $k, t \in Z^+$.

Every product may have different number of features and every feature may have different number of values. For example, feature and value representation of product car can be shown as:

Car = {color={red, blue, green}, fuel type={benzine, diesel, lpg}, production year={1946, 2004,2005,2006}}.

One way to find information that you are looking for is to use Internet. May be using a search engine is the shortest way to reach the data but, we know that there is a huge amount of data on Internet. Day by day it is increasing. Let us suppose we reached the data, now there are some different questions occur for changing the data into information. Because of this reason, there are some questions certainly must be answered. If I list the important questions about reaching the data and transforming it into the information, I would clearly specify the problem.

- What kind of information (or data) does the system need?
- Where does system find this information?
- Does system need a new search engine? If it does, which search engine does the system use to decrease the amount of work?
- Does system analyze all the data on the web or are there any restrictions?
- How does system find the addresses of the web sites?
- Does system need an agent located into the web sites that will be searched?
- Are the any standardized data on the web?
- How does system transform the data into the meaningful information?
- How does system detect and recognize the information?
- What kind of database does the system need?

Answers to above questions may be basically summarized as follow. The Internet is the digital area to find the data. But the desired system is especially interested in the e-commerce data. So it will be nonsense to visit the whole internet web pages. Instead of implementing a new search engine, our agent may use any existing one which is able to do specific searches. Data in Internet may be in structured,

unstructured or semi structured format. For the beginning, it can be considered all data is unstructured. Addresses of web pages are obtained from a search engine and that addresses are visited for downloading to work over them. Some Information Extraction algorithms must be used to extract the needed information. A relational database can be used for storing the data.

To achieve all these jobs, we developed our own software called FERbot (Feature Extractor Recognizer Bot) that is based on sequential word group frequencies. If we summarize the answer of question “What is FERbot?” we may explain FERbot basically.

- FERbot is a software suit.
- FERbot is a crawler.
- FERbot is a download manager.
- FERbot is an html converter.
- FERbot is a software that normalizes texts.
- FERbot is a text analyzer.
- FERbot is an Information Extractor.
- FERbot constructs Knowledge Bases automatically.
- FERbot is an Intelligent Agent.

FERbot is a kind of download manager that downloads Internet files from Internet. Download managers scan a webpage and download all files (image, sound, movie, achieve files etc.) linked to it, with customizable filters. Most popular download managers are listed in Table 4.1. Their main purposes are enabling fast download, downloading files whether there is a poor connection, downloading files with respect to different categories and searching for mirror sites to download files from different sources at the same time.

Table 4.1 List of some download managers (http://en.wikipedia.org/wiki/List_of_download_managers, 23.04.2007).

Aria	FileOwner 2005	HiDownload	NetAnts	True Downloader
BlackWidow	FlashGet	InstantGet	NetPumper	VisualWget
D4X	Free Download Manager.	Interarchy	Offline Downloader	WackGet
Download Accelerator Plus	Fresh Download	Internet Download Accelerator	Orbit Downloader	WebStripper
Download Dolphin	Getleft	Internet Download Manager	Quick Downloader	WellGet
Download Express	GetRight	KGet	ReGet	WinGet
Download Studio	Gigaget	LeechGet	Speed Download	WinWGet
DownThem All!	Go!Zilla	Mass Downloader	Star Downloader	wxDownload Fast
FileCopia	Gwget	Net Transport	Teleport Pro	Offline Explorer

FERbot is an html converter that gets html pages as input and produces a plain text as output. Like other html converters the aim of FERbot is extracting data or text from html pages to import data into a database or to analyze the text. Detagger (http://www.jafsoft.com/detagger/html-to-text.html) is a dual-purpose tool that can convert html to text or selectively remove html markup., ABC Amber Text Converter (http://www.processtext.com/abctxt.html) is the award-winning, powerful batch tool to convert documents to PDF, HTML, CHM, RTF, HLP, TXT (ANSI and

Unicode), DOC, XLS, MCW, WRI, WPD, WK4, WPS, SAM, RFT, WSD, PWI, PSW, PWD, and more.

Text normalization is the preprocess of using text for database storage, lexical analysis, language translation and etc. The main jobs of FERbot as a text normalizer are removing punctuation, converting letters into case sensitive form,

Details of modules of FERbot such as text analyzing, Information Extraction to convert data of WWW into database entries, Knowledge Acquisition for constructing Knowledge Base and Intelligence to learn the features and values of products will be mentioned in following sections.

4.2 Architecture of FERbot

FERbot is designed and implemented for the algorithm we developed for Information Extraction and Knowledge Acquisition. FERbot is a system that extracts the features of products from web pages, which comes as a result of search from a search engine (eliminating irrelevant sites and pages). After applying algorithms to organize this unstructured data, FERbot classifies features and their values into different categories. There are three main parts of FERbot system as shown in Figure 4.1 where IG represents the Information Gathering part, UI represents the User Interface part and KM represents the Knowledge Modelling part.

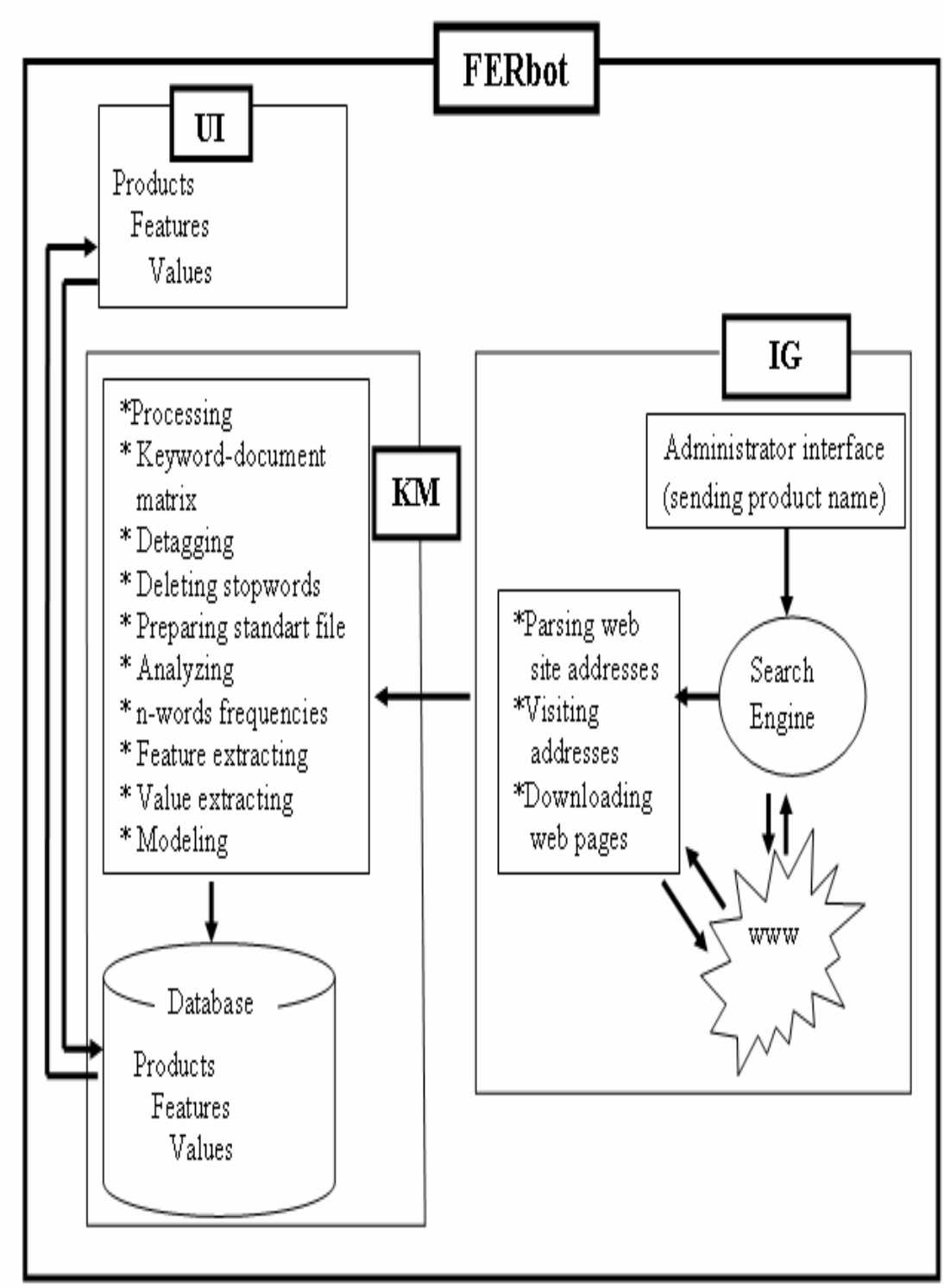


Figure 4.1 Architecture of FERbot: IG (Information Gathering), KM (Knowledge Modeling), UI (user interface).

At the Information Gathering part, FERbot visits Internet pages and stores them into a local machine to work over these web pages. FERbot knows which page to visit by filtering the addresses of web pages from Google's main page. To obtain more specific result page, FERbot sends the product name as a query to Google Search engine by customizing some searching options.

At the Knowledge Modelling part which is the most important, FERbot parses these downloaded html files to have a structured form for data. FERbot applies some statistical analysis and hierarchical algorithms about the collected data to define the product features. The algorithm we developed which is based on sequential word groups will be the core of the system FERbot.

At the user interface part, FERbot allows customers to present their queries. Customers may select the query from the list or they may write their queries. After accepting query, related answer is retrieved from database and sent back to user.

4.2.1 Information Gathering

Information agents are tools that we use for managing and manipulating the explosive growth of information currently. They will be used for the same purpose for future. Customers use Internet to discover the price or any general property of a product much more easily than discovering them in a conventional way. As seen in Figure 4.2, the first step of the FERbot is to find the web pages related to products from Internet.

Since the advent of e-commerce, the web has been increased in size at a phenomenal rate. It means that, there are lots of documents including unstructured material such as tables, indexes, video, etc. Search engines use softwares that roam the web to collect information from URL names, meta-tags and document titles. At the information gathering part of the searching mechanism, the main questions are "Which sites must be visited?" and "Which information must be collected from each

site?" Everyday the number of the web sites to be visited is increasing. Processing every document in www is often expensive and unnecessary.

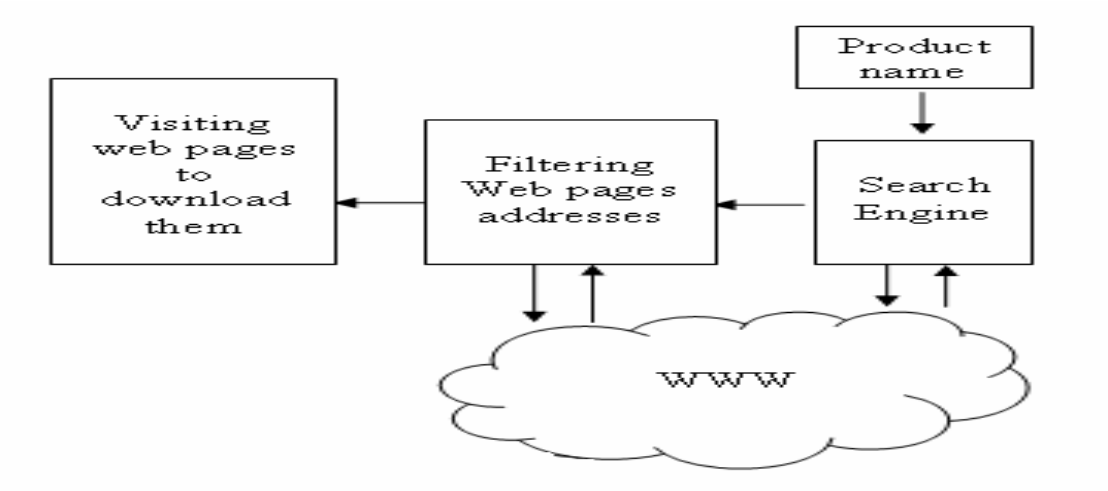


Figure 4.2 Information gathering part of FERbot .

Internet agents could be mobile, i.e. they may be able to traverse the WWW, gather information and report what they retrieve to a home location (Nwana 1996). Nwana (1996) shows how an information agent, typically within some browser like Netscape, uses a host of Internet management tools such as Spiders and search engines in order to gather the information in Figure 4.3. The information agent is associated with some particular indexer(s), e.g. a Spider. He defines a Spider as an indexer able to search the WWW, depth-first, and store the topology of the WWW in a DataBase Management System (DBMS) and the full index of URLs in the Wide Area Information Servers (WAIS).

The problem of retrieving relevant documents, for some definition of relevance from large text collections has been studied for decades in the information retrieval field. One way to restrict this huge number of web sites is using a search engine. FERbot uses Google search engine to visit just the e-commerce sites that sell products. For that reason web site addresses including domain name extensions like "edu", "mil", "org", "net", "gov", "aero", "biz", "coop", "info", "pro" are eliminated not to visit that sites.

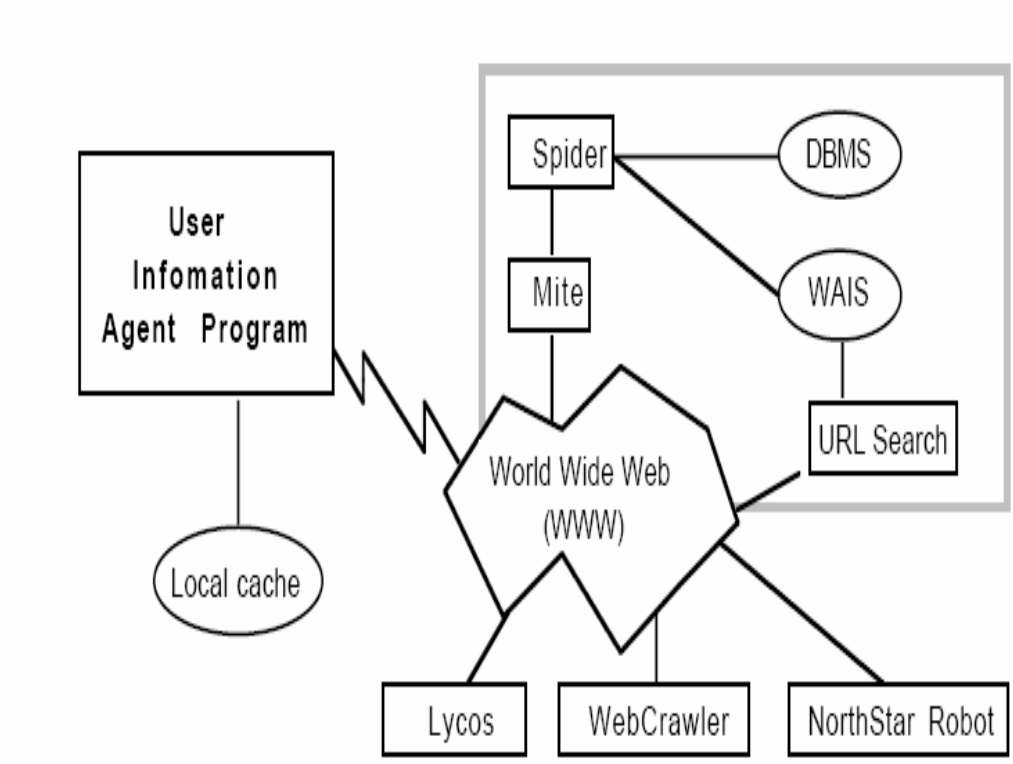


Figure 4.3 A view of how Information Agents Work (Nwana ,1996 (Adapted from Indermaur, 1995).

Because of FERbot analyzes text, files having some extension names must be eliminated not to visit the image or movie file even they are within a commercial site. Also some document files (.doc .txt .pdf) include long explanations more than 2MB. It is not a good idea to search these files. Finally, files having file extension names like “.pdf”, “.doc”, “.ppt”, “.xls”, ”.jpg”, “.jpeg”, “.gif” “.pict” “.bmp” “.tif” “.png” “.ico” are eliminated not to download.

Google search engine enables maximum 1000 results for a query for customers to see from the Google’s main page. Google applies Latent Semantic Index (LSI) to every query. LSI can be used to find relevant documents that may not even share any search terms provided by the user (Bery, Dumais and Brien, 1995). Google search results include both the product names and the relevant terms of product names. Every search-result of Google includes the web site name, web site address and a small summary including the queried word(s) as seen in Figure 4.4.

Figure 4.4 An example of Google Search Engine result including the web site name, address and a small summary about the query in Turkish web sites.

Google search engine uses different styles to insert hyperlinks for web site names, addresses and summaries. To leave alone the links showing the web site address of the query, Google's search results are parsed. FERbot applies pattern matching method to search regular expressions beginning with "<A class=l". The web page address is the link to visit it. These patterns are in italic form in Figure 4.5.

Parsed hyperlinks of web site addresses are written into a text file to keep them for reuse. Every link includes an identifier showing its order. The identifier can be considered as the pointer of that link. Sometimes Google produces less than 1000 results. At that situation the highest numbered pointer is the total results. Also it is possible to see same addresses at the result page. After downloading links, same web site addresses are deleted. A section from the file web page addresses can be seen in Figure 4.6.

According to Google Search Engine results, most popular 1000 links are refined and ready to be visited. Since FERbot is a centralized system, it begins to download web sites from the first link to the last one. There are some reasons that prevent FERbot to download web pages. First, some web pages are protected for downloading. Second, some web pages are not available and third, some downloadings are not completed. At these situations FERbot skips the web page and downloads the next web page.

FERbot interacts with e-commerce information resources that are designed to be used by customers. Semi structured information resources increase the performance of FERbot. An information resource gives information as an answer to a query of someone. If the information is in structured format like embeded data into tables and hierarchical manners, FERbot easily extract information from there. FERbot downloads web pages which have different types of page layouts and text formats as seen in Figure 4.7, Figure 4.11 and Figure 4.15.

results">http://www.google.com/search?hl=tr&rls=GGLG,GGLG:2005-48,GGLG:en&q=+site:www.araba.com+araba&sa=X&oi=smap&resnum=1&ct=more-results</A></SPAN></FONT></TD></TR></TBODY></TABLE></DIV>
<DIV class=g>
<H2 class=r>

<A class=l onmousedown="return clk(this.href,'','','res','2','')" href="http://www.arabadergisi.com/">

Arabadergisi.com - Türkiye'nin En Büyük Otomobil Dergisi <B>...</B></A></H2>
<TABLE cellSpacing=0 cellPadding=0 border=0>
 <TBODY>
 <TR>
  <TD class=j><FONT size=-1>arabadergisi.com, <B>araba</B>, dergi, otomobil, otomobil fiyatlari, fiyat, 2.el, otomobil resimleri, resim, teknik bilgiler, bmw, mercedes, ferrari, alfa romeo, <B>...</B>
<SPAN class=a>www.<B>araba</B>dergisi.com/ - 22k - </SPAN><NOBR> <A class=fl href="http://209.85.135.104/search?q=cache:_mw0nhXplNkJ:www.arabadergisi.com/+araba&hl=tr&ct=clnk&cd=2">Önbellek</A> - <A class=fl href="http://www.google.com/search?hl =tr&rls=GGLG,GGLG:2005-4

Figure 4.5 FERbot parses Google's result page which includes web page addresses.

1 http://www.araba.com/
2 http://www.arabadergisi.com/
3 http://www.amerikanaraba.com/
4 http://www.kiralikaraba.com/
5 http://www.arabafilms.com/
6 http://www.komikler.com/komikresim/kategori.php?catid=26
7 http://www.komikler.com/komiktv/kategori.php?catid=3
8 http://www.karpuz.com/db/oyn/araba/araba.html
9 http://www.showtvnet.com/Hobby/Yapalim/modelcilik/model_araba/index.shtml

10 http://www.showtvnet.com/hobby/eglenelim/sihir/maske/araba.shtml (same)
11 http://www.showtvnet.com/hobby/eglenelim/sihir/maske/araba.shtml (same)

12 http://www.komikler.com/komiktv/kategori.php?catid=3
13 http://www.arabakira.com/
…
…
…
988 http://www.uluslararasiegitim.com/ulkeler/usa/yasam/4/arkira.asp

Figure 4.6 Hyperlinks are written into a text file and same links are eliminated.

Especially, wrappers separate documents into different segments. Like most of the web page designer, we think that meta tags of HTML codes give brief idea about a web page as seen Figure 4.8, Figure 4.12 and Figure 4.16. Some of the web pages use keyword but some of them don't. The keywords of these examples of web pages are in italic form in these Figures.

Figure 4.9, Figure 4.13 and Figure 4.17 are the examples of information units that must be extracted from web pages.

Figure 4.10, Figure 4.14 and Figure 4.18 are the examples of html codes of information units that must be extracted from web pages. The patterns that will be extracted are bold colored.

FERbot extracts all keywords and other information units by using the pattern matching and sequential word frequencies which are mentioned in section 4.2.2.

Figure 4.7 Web page of http://www.araba.com/.

```
<HTML><HEAD><TITLE>ARABA.COM.TR Ltd. Şti. - İkinci El Araba - Otomobil - Araç İlan Sitesi</TITLE>
<META http-equiv=Content-Type content="text/html; charset=ISO-8859-9">
<META content="İkinci El Araba - Otomobil - Araç İlan Sitesi" name=description>
<META content="yedek parça, fiyatları, kiralama , kiralik , pazari, aksesuarları , ilanlar , 2 el satış, unfallwagen, markt, aksesuarı, araba satıcıları , ılan, araba modelleri, araba alım, satılık , resimleri, siteleri, edaran , nasional , fiatları , lastik , 2 el kredisi , modifiye ikinci el , ilanları ilk , aksesuarları, hasarlı , www otomobil, otomobil markaları, otomobil satıcıları, otomobil tasarım, ikinci el otomobil siteleri"
name=KEYWORDS>
<META http-equiv=imagetoolbar content=no>
<META content=Admin name=author>
<META content="ARABA.COM.TR Ltd. Şti." name=copyright>
<META content="ARABA.COM.TR Ltd. Şti." name=publisher>
<META content=all name=robots>
<META content=index name=robots>
<META content=follow name=robots>
```

Figure 4.8 html code of http://www.araba.com/.

Figure 4.9 Information unit that is extracted from web page of http://www.araba.com/.

```
<TABLE cellSpacing=2 cellPadding=2 width="100%" align=center border=0>
<TBODY> <TR> <TD vAlign=top width="25%">
<DIV align=center><A href="http://www.araba.com/533231.html"> <IMG
alt="Hyundai Getz 1.3 GLS  (Hyundai / Getz)"
src="ARABA_COM_TR Ltd_ Şti_ - İkinci El Araba - Otomobil –
Araç İlan Sitesi_files/x245ba00efe0d7ebe1c3e7ab061180762.jpg"
width=75 border=0><BR>Hyundai Getz 1.3.<BR>16.500
YTL</A></DIV></TD><TD vAlign=top width="25%">
<DIV align=center><A  href="http://www.araba.com/525649.html"> <IMG
alt="Mercedes SL 500 AMG Class CABRIO (Mercedes / SL-Class)"
src="ARABA_COM_TR Ltd_ Şti_ - İkinci El Araba - Otomobil - Araç İlan
Sitesi_files/xbbcfb3ab2e6e2be6c6a45bba202d510d.jpg"
width=75 border=0><BR>Mercedes SL 500 .<BR>44.900
YTL</A></DIV></TD><TD vAlign=top width="25%">
<DIV align=center><A href="http://www.araba.com/513394.html"> <IMG
 alt="Mercedes 300 CE Coupe (Mercedes / E-Class)"
src="ARABA_COM_TR Ltd_ Şti_ - İkinci El Araba - Otomobil - Araç İlan
Sitesi_files/x5fc34dbaf871d66bc52fbe1fff4297f7.jpg"
width=75 border=0><BR>Mercedes 300 CE .<BR>18.000
YTL</A></DIV></TD><TD vAlign=top width="25%">
<DIV align=center><A
href="http://www.araba.com/529114.html"><IMG
alt="Mercedes S 320 Long (Mercedes / S-Class)"
src="ARABA_COM_TR Ltd_ Şti_ - İkinci El Araba - Otomobil - Araç İlan
Sitesi_files/x842d5264d7474838631faaa1d3292e2e.jpg"
width=75 border=0><BR>Mercedes S 320 L.<BR>26.500
YTL</A></DIV></TD></TR></TBODY></TABLE>
```

Figure 4.10 Html code of information units that must be extracted from http://www.araba.com/ are bold colored.

Figure 4.11 Web page of http://www.otobulten.com/.

```
<META http-equiv=Content-Type content="text/html; charset=iso-8859-9">
<META content=document name=resource-type>
<META content="MSHTML 6.00.2800.1106" name=GENERATOR>
<META
content="araba, araç, otomobil, oto, galeri, galerici, pazar, oto pazar, dergi, kiralık,
ilan, 2. el araç, 2.el oto, sigorta, ikinci el araba, 2. el araba"
name=classification>
<META
content="araba, araç, otomobil, oto, galeri, galerici, pazar, oto pazar, dergi, kiralık,
ilan, 2. el araç, 2.el oto, sigorta, ikinci el araba, 2. el araba"
name=description>
```

Figure 4.12 A section from html code of http://www.otobulten.com/.

Figure 4.13 Information unit that is extracted from web page of http://www.otobulten.com/.

Otomobil - Pazar Fiyatları Sitesi_files/_t3_21007.jpg" width=120>
<DIV class=f_fi>27,950 YTL</DIV>
<DIV class=f_syh>2. el Binek Araçlar
2006 **Renault**
Megane ıı Dynamıc</DIV>Dogu OtomotİV</A>
<DIV class=f_kc>13.03.2007</DIV></TD>
<TD
class=vt><Ahref="http://www.otobulten.com/index.php?sa=modelne&kn = 3&tn= 21053&mr=23"><IMG height=90
alt="2. el Binek Araçlar2004 **Mercedes C 180** Akay Otomotiv"
src="Araba Center - Otobülten 2_ El Arac - Otomobil - Pazar Fiyatları Sitesi_files/_t3_21053.jpg" width=120><DIV class=f_fi>36,000 €</DIV>
<DIV class=f_syh>2. el Binek Araçlar
2004 Mercedes C
180</DIV>Akay Otomotiv</A><DIV class=f_kc>14.03.2007</DIV></TD>
<TD class=vt><Ahref="http://www.otobulten.com/index.php? sa= modelne &kn=2&tn=5820&mr=40"><IMG height=90
alt="2. el Ticari Araçlar2005 **Volkswagen Caddy** Ünal Otomotiv"
src="Araba Center - Otobülten 2_ El Arac - Otomobil - Pazar Fiyatları Sitesi_files/_t2_5820.jpg" width=120>
<DIV class=f_fi>Fiyat Belirtilmemiş</DIV>
<DIV class=f_syh>2. el Ticari Araçlar
2005 Volkswagen
Caddy</DIV>Ünal Otomotiv</A>
<DIV class=f_kc>14.03.2007</DIV></TD>
<TD class=vt><A
href="http://www.otobulten.com/index.php?
sa=modelne&kn=3&tn=20980&mr=11"><IMG height=90
alt="2. el Binek Araçlar2004 **Fiat Albea** Multijet Şavlı Otomotiv"
src="Araba Center - Otobülten 2_ El Arac - Otomobil - Pazar Fiyatları Sitesi_files/_t3_20980.jpg" width=120><DIV class=f_fi>15,950 YTL</DIV>
<DIV class=f_syh>2. el Binek Araçlar
2004 Fiat Albea
Multijet</DIV>Şavlı Otomotiv</A><DIV

Figure 4.14 Html code of information units that must be extracted from http://www.otobulten.com are bold colored.

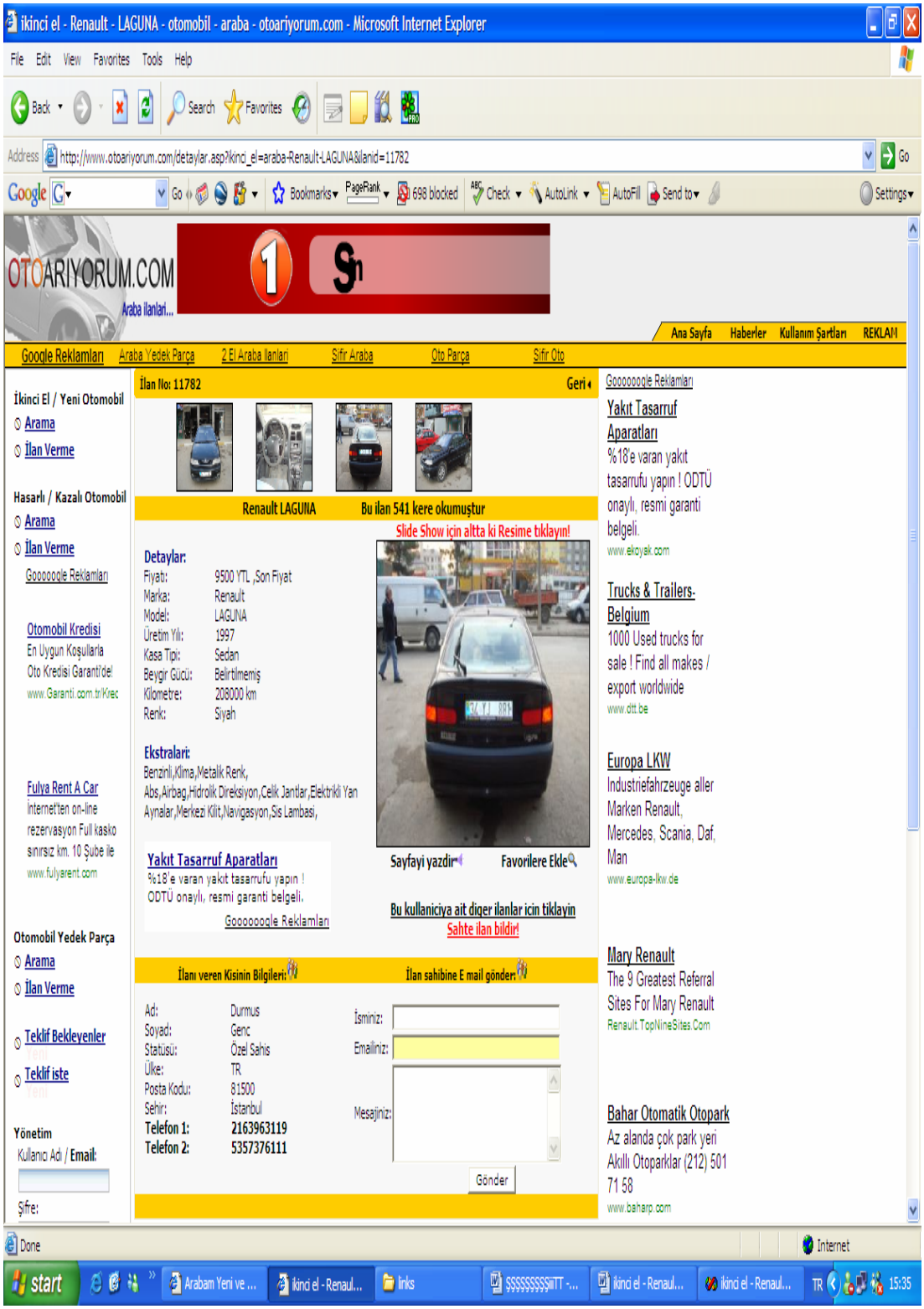


Figure 4.15 Web page of http://www.otoariyorum.com/detaylar.asp?ikinci_el=araba-Renault-LAGUNA &ilanid=11782.

<HTML><HEAD><TITLE>ikinci el - Renault - LAGUNA - otomobil - araba - otoariyorum.com</TITLE>

<META http-equiv=Content-Type content="text/html; charset=ISO-8859-9">
<META content="ARABA - Renault - LAGUNA - ikinci el otomobil - otoariyorum.com"
name=description>

<META content="Araba,Renault , LAGUNA, " name=keywords>

Figure 4.16 A section from html code of http://www.otoariyorum.com/detaylar.asp? ikinci_el=araba-Renault-LAGUNA&ilanid=11782.

Detaylar:	
Fiyatı:	9500 YTL ,Son Fiyat
Marka:	Renault
Model:	LAGUNA
Üretim Yılı:	1997
Kasa Tipi:	Sedan
Beygir Gücü:	Belirtilmemiş
Kilometre:	208000 km
Renk:	Siyah

Figure 4.17 Information units that must be extracted from http://www.otoariyorum.com/detaylar.asp? ikinci_el= araba-Renaul t- LAGUNA &ilanid = 11782.

```
<TABLE style="BORDER-COLLAPSE: collapse" cellPadding=0
width=272 border=0>
<TBODY><TR><TD width=89> </TD>
<TD> </TD></TR><TR> <TD width=89><FONT
color=#000080><B>Detaylar:</B></FONT></TD>
<TD> </TD></TR><TR>
<TD width=89>Fiyatı:</TD>
<TD>9500 YTL ,Son Fiyat</TD></TR>
<TR><TD width=89>Marka:</TD>
<TD>Renault</TD></TR>
<TR><TD width=89>Model:</TD>
<TD>LAGUNA</TD></TR>
<TR><TD width=89>Üretim Yılı:</TD>
<TD>1997</TD></TR>
<TR> <TD width=89>Kasa Tipi:</TD>
<TD>Sedan</TD></TR>
<TR><TD width=89>Beygir Gücü:</TD>
<TD>Belirtilmemiş</TD></TR>
<TR> <TD width=89>Kilometre:</TD>
<TD>208000 km</TD></TR>
<TR> <TD width=89>Renk:</TD>
<TD>Siyah</TD></TR></TBODY></TABLE>
<TABLE style="BORDER-COLLAPSE: collapse" cellPadding=0
```

Figure 4.18 Html code of information units of http://www.otoariyorum.com/detaylar.asp?ikinci_el=araba-Renault-LAGUNA&ilanid=11782 are bold colored.

4.2.2 Knowledge Modeling

After downloading all web pages, to find the frequencies of sequential word groups of the collected data, FERbot applies a regular process.

1. All html files have to be detagged. FERbot cleans all java, php, asp and etc. scripts, and deletes all markups of html codes. FERbot writes all text without markups and scripts into a text file having word-space-word format. Removing mark ups of html files is not the only thing to get an input file to work over it.
2. FERbot changes all letters into lower case in a document to make that document be more standardized.
3. FERbot converts all text formats into a same text format and deletes all punctuation marks.
4. There are some stopwords in a language to link sentences or phrases. For example in Turkish language they are “için, kim, bu” and etc. Cleaning these stopwords decreases the work on analysing whole data. FERbot cleans approximately 180 types of stopwords of Turkish language from documents.

It should be known that there is no definitive list of stopwords, the list of stopwords may change from search to another search. Stopword list that is used by FERbot for Turkish language is given in Table 4.2.

After applying these four steps over the collected data, there will be a standard file as shown in Figure 4.19. Stemming, or removing suffixes (and sometimes prefixes) to reduce a word to its root form, has relatively long tradition in the index building process (Berry and Browne, 1999). Stemming is useful if it is used to categorize similar words into same group but has some disadvantages if words have same writings but different meanings. Also, features of products sometimes are very similar if they are for different versions of same product. For this reason, FERbot does not use stemming algorithms.

Table 4.2 Stopword list that is used by FERbot for Turkish language.

acaba	bu	gibi	milyar	sekiz
altmış	buna	göre	milyon	seksen
altı	bunda	tr	mitte	sen
ama	bundan	header	mu	senden
ana	bunu	hem	mutlu	seni
anasayfa	bunun	hep	mü	senin
and	car	hepsi	müşteri	sık
arama	cat	her	mı	siz
bana	com	hiç	nasıl	sizden
banner	copyright	http	ne	sizi
başvuru	çok	iki	neden	sizin
ya	çünki	ile	nerde	yani
bazı	çünkü	ise	nerede	sorulan
belki	da	için	nereye	sorular
ben	daha	iletişim	niye	start
benden	dahi	kadar	niçin	şey
beni	de	katrilyon	on	şeyden
benim	defa	kdv	ona	şeyi
beş	diger	kez	ondan	şeyler
bin	diğer	ki	onlar	şu
bir	diye	kim	onlardan	şuna
biri	diyene	kimden	onları	şunda
birkaç	doksan	kime	onların	şundan
birkez	dokuz	kimi	onu	şunu
birşey	dört	kişisel	otuz	şti
birşeyi	ekle	kırk	raquo	tic
biz	elli	kirk	rent	www
bizden	en	login	sanki	trilyon
bizi	end	ltd	sayfa	tüm
bizim	üye	mi	veya	üç

500 000 000 tl yılı 1997 km 106000 motor gücü 105 bg 77 kw renk mavi met silindir hacmi 1600 cm sup3 yakıt tipi benzin yer 81100 kozyatağı il istanbul toyota terra 1 6 special 16 500 ytl 16 500 000 000 tl yılı 2001 km 84000 motor gücü 110 bg 81 kw renk gümüş silindir hacmi 1598 cm sup3 yakıt tipi benzin yer 42550 akşehir il konya toyota corolla 1 6 xei 12 500 ytl 12 500 000 000 tl yılı 1997 km 90000 motor gücü 110 bg 81 kw renk kırmızı silindir hacmi 1598 cm sup3 yakıt tipi benzin yer 42550 akşehir il konya toyota corolla 1 6 xei ac 13 750 ytl 13 750 000 000 tl yılı 1995 km 84800 motor gücü 110 bg 81 kw renk beyaz silindir hacmi 1600 cm sup3 yakıt tipi benzin lpg yer 58040 merkez il sivas toyota corolla gli 1 6 efı 16v 17 000 ytl 17 000 000 000 tl yılı 1997 km 59000 motor gücü 115 bg 85 kw renk gümüş silindir hacmi 1600 cm sup3 yakıt tipi benzin yer 60100 merkez il tokat toyota corolla 1 6 xeii 13 400 ytl 13 400 000 000 tl yılı 1997 km 100000 motor gücü 114 bg 84 kw renk bej silindir hacmi 1600 cm sup3 yakıt tipi benzin

Figure 4.19 A view from the text file which is ready to be analyzed.

4.2.2.1 Feature and Value Extraction

According to Grishman (1997), the process of information extraction has two major parts; first the system extracts individual "facts" from the text of a document through local text analysis. Second, it integrates these facts, producing larger facts or new facts (through) inference. Stevenson (2006) emphasizes the importance of usage of multiple sentences for facts instead of single sentence. Karanikas (2002) thinks that primary objective of the feature extraction operation is to identify facts and relations in text. Some knowledge engineers try to construct knowledge bases from web data based on information extraction using machine learning (Craven, 2000). To collect knowledge about a product, Lee (2004) works with specially designed interfaces to allow domain experts easily embed their professional knowledge in the system. Gomes and Segami (2007) study the automatic construction of databases from texts for problem solvers and querying text databases in natural language, in

their research. The system that is offered by Shamsfard and Barforoush (2004) starts from a small ontology kernel and constructs the ontology through text understanding automatically.

E-commerce web pages include advertisements about products to give information about features and values of these products. Different web pages may use different web page layout or different notation to show the information of a same product. For FERbot, it is not important whether a web page is labeled or not. FERbot extracts information into a hierarchical structure.

Every product has some features and every feature has some values. For example, “television”, “car” and “book” are products. “Color”, “fuel type” and “motor power” are features of product “car” where “red”, “blue” and “white” are the values of the color feature of “car”. Relation between values and features and relation between features and products are similar, and it is the relation Has. Has is a 1-to- *n* relation between products and features and features and values.

In our study, frequencies of *n*-words groups are used to extract features and values from the input document. We assume that firstly the name of the feature and then the values of that feature appear in a text to explain a product. Also we assume that a feature can be represented by a single word or by words and a value can be represented by a single word.

Automatic discovering of relations between features and values bases on lexical and syntactic analysis of the text. A sequential word group is constructed by words which can be in a meaningful or non-meaningful structure. For that reason, sometimes a noun phrase, adjective phrase or a verb phrase may refer to sequential word groups, for example, “green car” , “International Business Machine” or “Pay me five dollars”. Sometimes non-meaningful structures refer to a sequential word groups like “is the”, “dady 500 xlgs” or “2 4 on”.

Assume that we have a set of e-commerce web pages *WP*.

$WP = \{wp_1, wp_2, \ldots, wp_n\}$, $n \in Z^+$. Every wp_i is a pair of words and keywords, $wp_i = <W_i, K_i>$, where $W_i = [w_{i_1}, \ldots, w_{i_p}]$ and $K_i = \{k_{i_1}, \ldots, k_{i_s}\}$, $i, p, s \in Z^+$. W_i is the list of words where every w_{i_j} appears in the body part of i^{th} html page, K_i is the set of keywords of html pages where every k_{i_j} is defined in meta-keyword-tag of i^{th} html page.

If we combine all lists of words, W_i, successively we create a new list of words, $S = [W_1, W_2, W_3, \ldots, W_n]$. Order of words is important in list *S* and a word can appear more than one time in this list. If we collect all sets of keywords, K_i, in a single set, we create a new set of keywords, $K = \{K_1, K_2, K_3, \ldots, K_n\}$. We analyze the list *S* is in section 4.2.2.2 to find the features and values of a product and we work over the set *K* in section 4.2.2.3 to refine these features and values.

4.2.2.2 Sequential Word Groups

In our study, frequencies of *n*-words groups are used to extract features and values from the list *S*. We assume that firstly the name of the feature appears and then the value of this feature appears in a text to explain a product. Also we assume that a feature can be represented by a single word or by a sequence of words and a value can be represented by a single word.

Let SWG_i^r denote a sequential word group consisting of *r* words, starting from i^{th} word of list *S*.

The list of sequential word groups that have only 1-word can be shown by SWG^1 where $SWG^1 = [SWG_1^1, \ldots, SWG_n^1] = S$.

The list of sequential word groups that have 2-words can be shown as $SWG^2=[\,SWG_1^2, \ldots, SWG_n^2\,]$ where $SWG_i^2 = w_i^2\ w_{i+1}^2$

Finally, the list of sequential word groups that have *r*-words can be shown as $SWG^r = [SWG_1^r, \ldots, SWG_n^r]$ where $SWG_i^r = w_i^r\ w_{i+1}^r\ \ldots\ w_{i+r-1}^r$.

We analyze the list *S* to find the frequencies, $f_{SWG_i^n}$, of sequential word groups. We keep sequential word groups including *r*-word(s) and their frequencies in different text files. These text files are basically arrays of word(s) and their frequencies.

A feature of a product can only be a single word or it can be a noun phrase or an adjective phrase. For example, features of a car are color, fuel type, motor power and etc. For extracting features represented by two-words, we use three lists SWG^1, SWG^2 and SWG^3. We also use frequencies of every $SWG, f_{SWG_i^n}$, for comparison of frequencies of different *SWG*s.

Let us consider that first *n* unique *SWG*s of SWG^1 which have the highest frequencies are the features of the product *P*. Empirically *n* can be set to an integer number between 50 and 100. We compare frequency of first *n* unique *SWG of* SWG^1 and frequencies of *SWG* of SWG^2 beginning with the same word. If they are approximately equal, the feature is represented by two-words instead of single word. Formally, if $f_{SWG_i^1} = f_{SWG_j^2} + t$ then

$F_k = SWG_j^2 = w_j^2\ w_{j+1}^2$

where F_k shows a feature of product, $P = \{F_1 , ... , F_n \}$, $SWG_j^2 = SWG_i^1 \; w_{j+1}^2 = w_j^2 \; w_{j+1}^2$ and t is an integer number to show that frequencies are approximately equal.

After selecting a feature, it is time to find the values of this feature F_k. For every SWG from SWG^3, if the first two words of SWG are equal to the feature, F_k, then the third word that comes after these two words is set to a value of this feature, F_k. Formally, If $SWG_j^3 = F_k \; w_{j+2}^3$ then

$V_{k_t} = w_{j+2}^3$ where $F_k = w_j^3 \; w_{j+1}^3$ and V_{k_t} is one of the values of feature F_k ,

$V_{k_t} \in F = \{V_1 , V_2, V_3 ,\ldots,V_n \}$.

This process continues until all features that include two-words and values of these features are found.

To find the features including one-word, again we take into account of n unique SWG of SWG^1 which have the highest frequencies. We eliminate the $SWGs$ which are used for any feature including two words or used for a value of these features. Formally, if $SWG_i^1 \notin P$ or $SWG_i^1 \notin F$ then

$F_k = SWG_i^1$ where F_k becomes a feature of product P. Values of this feature F_k becomes the second words of SWG^2, beginning with the feature F_k.

Formally, if $SWG_j^2 = F_k \; w_{j+1}^2$ then

$V_{k_t} = w_{j+1}^2$ where $F_k = w_j^2$ and V_{k_t} is one of the values of feature F_k , $V_{k_t} \in F$.

This process continues until to all features that include one-word phrases and values of these features are found. A description of a product P,

$P = \{ F_1 = \{V_{1_1}, \ldots, V_{1_k}\}, \ \ldots \ , \ F_n = \{V_{n_1}, \ldots, V_{n_t}\}\}$, is completed ignoring features including more than two-words. Figure 4.20 shows how to find the features including more than two words and values of this feature.

```
(1)  Begin
(2)    Set i ,j ,k ,t to 1
(3)    While Not Eof( SWG_i^z )
(4)      If f_{SWG_i^z} = f_{SWG_i^{z+1}} Then
(5)        F_k = SWG_i^{z+1}
(6)      Endif
(7)      While Not Eof ( SWG_i^{z+2} )
(8)        If F_k = w_j^{z+2} w_{j+1}^{z+2} Then
(9)          V_{k_t} = w_{j+2}^2 , t=t+1
(10)         Endif
(11)         Locate F_k into set P
(12)         For i=1 to t-1
(13)           Locate V_{k_t} into set F_k
(14)         Next
(15)         k=k+1,  t=1
(16)       Endwhile
(17)       i=i+1, j=1
(18)     Endwhile
(19)   End.
```

Figure 4.20 Pseudo code of finding features including more than two words and values of these features.

The complexity of algorithm in terms of Big *O* Notation can be computed in following manner. Statement (13) takes $O(1)$ time but it is runned for n-1 times. So, statements (12), (13), (14) take O(n-1) time. Statements from (8) to (10) take $O(1)$ time and. Statement (11) and statement (15) takes $O(1)$ time, too. Statements from (8) to (15) takes O(max(1,1,n-1,1))=O(n-1) time and are run for n times so, Statements (7) to (16) take O(n*(n-1))= $O(n^2-n)= O(n^2)$ time. Statement (17) takes $O(1)$, statements from(8) to (10) take $O(1)$ time. Statements from (4) to (17) takes O(max(1, n^2,1))=$O(n^2)$ time and are runned for n times finally. Statemenst from (1) to (19) takes $O(n^2 * n)= O(n^3)$ time.

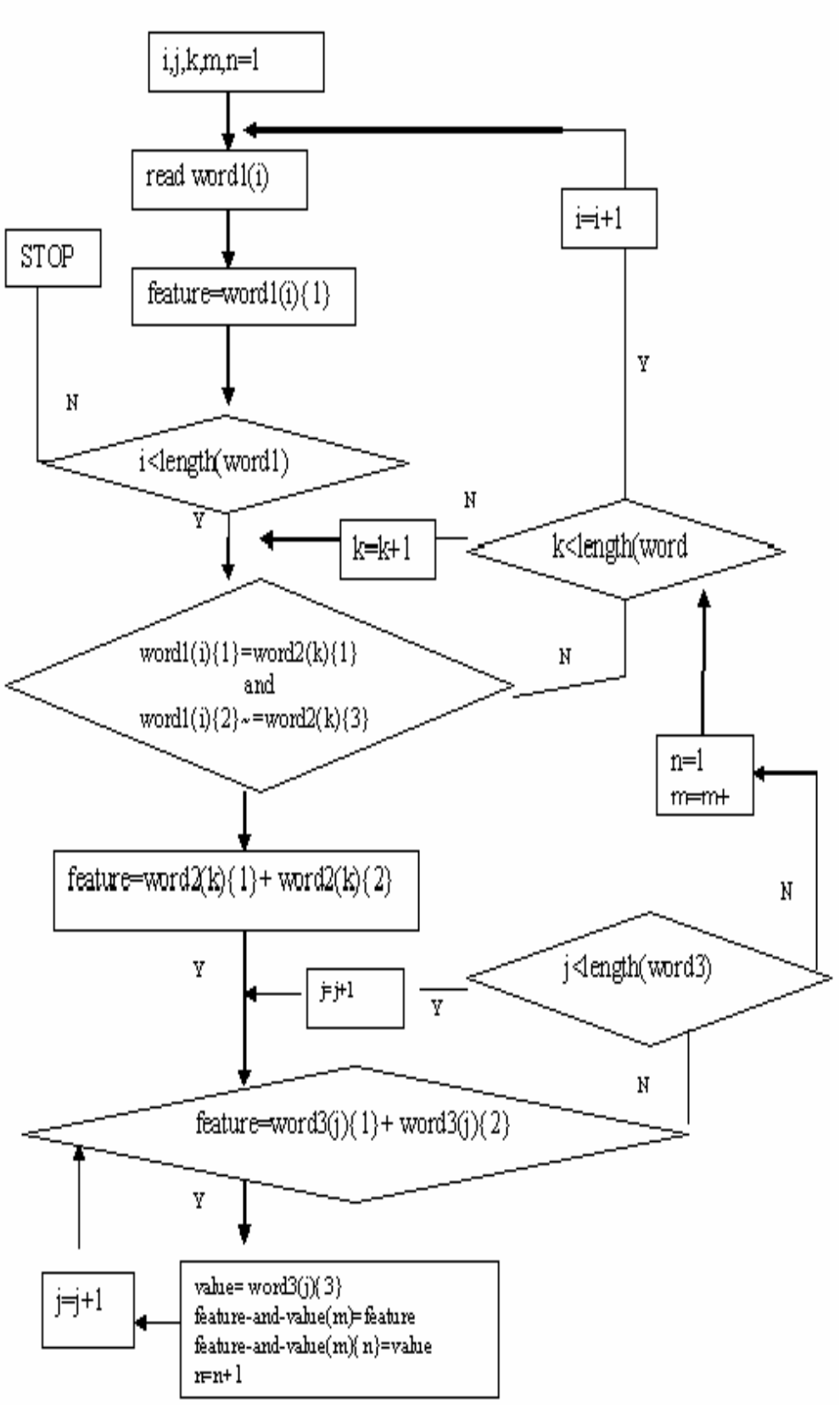


Figure 4.21 Flow chart to find the features which are represented by two-words and values of these features.

For programmers, the flowchart to find the features and their values for features that are represented by two-words phrases in array form can be seen in Figure 4.21 where

word1(i) = i^{th} element of SWG^1 .

word1(i){1} = first word of word1(i) .
word1(i){2} = frequency of the word of word1(i).

word2(k) = k^{th} element of SWG^2 .

word2(k){1}= first word of word2(k).
word2(k){2}= second word of word2(k).
word2(k){3}=frequency of the two-words group of word2(k).

word3(j) = j^{th} element of SWG^3 .

word3(j){1} = first word of word3(j).
word3(j){2} = second word of word3(j).
word3(j){3} = third word of word3(j).
word3(j){4} =frequency of the three-words group of word3(j).

4.2.2.3 Defining Mostly Used Keywords to Refine Features and Values

There is a collection of web sites having “html” or “htm” file name extensions. Parsing web pages helps us to find out important parts of any page. Most of the web page designer thinks that the title, description, and keyword meta tags gives a brief idea for that page. FERbot uses a keyword-document matrix to see if any keyword is used by any web page. In this matrix, rows represent the keywords and columns represent the web pages. When a keyword exists in a web page then the intersection of column and row takes 1, otherwise it takes 0. Number of 1s in a row gives the frequency of the keyword used in web pages. All frequencies of keywords are computed and then keywords are eliminated from the set K, if they have a frequency lower than the standard deviation of the average of keyword frequencies. FERbot considers the rest of the keywords have a strong relationship with the product name. After parsing web pages it was seen that a lot of designers did not use a keyword or

even a title. Another important negative status is, designers locate whatever they find into a keyword or description meta tags. This makes that page uninteresting to be parsed for the keyword-document matrix. FERbot also searches for the words written in meta tags (bold, strong, etc) to find the emphasized words in a web page. FERbot uses all these results to refine the results of the feature and value extraction.

Features and values of a product can be refined using the set of keywords, $K=\{k_1, k_2, \ldots, k_n\}$, of html pages where k_i appears in meta-keyword-tag of html page. $D = [d_{ij}]_{mxn}$ is a binary matrix shows which web pages use the keyword, k_i , in meta-keyword-tag. m shows number of keywords and n shows number of web pages. A keyword may include more than one word. Value of each entry of matrix D can be computed by the following conditions.

$$d_{ij} = \begin{cases} 1 & , \text{ if } i^{\text{th}} \text{ keyword occurs in } j^{\text{th}} \text{ document} \\ 0 & , \text{ if } i^{\text{th}} \text{ keyword doesn't occur in } j^{\text{th}} \text{ document} \end{cases}$$

For example, in the following D matrix, d_{21} has a value of 1 and means that second keyword of set K is the keyword of the first web page, formally, $k_2 \in wp_1$.

$$D = \begin{bmatrix} 0 & 1 & 1 & \ldots & 0 \\ 1 & 1 & 1 & \ldots & 1 \\ 1 & 0 & 0 & \ldots & 1 \\ \ldots & \ldots & \ldots & \ldots & . \\ \ldots & \ldots & \ldots & \ldots & . \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix}_{mxn}$$

Number of 1s in a row gives the frequency, f_{k_i} , of i^{th} keyword, k_i , used in web pages, wp. Frequency of a keyword can be computed as

$$f_{k_i} = \sum_{j=1}^{n} d_{ij} \ .$$

A section from Keyword-File Matrix of product “araba”can be seen in Table 4.3.

Table 4.3 A section from Keyword-File Matrix of product “araba”.

Keywords	Html files including keyword, k_i, represented by “1” otherwise by “0”.
Araba ilan	00000011111101010100000001111000000…
araba kampanya	00010000010000000010000001000111000…
araba kiralama	00000001000001000100000000100000001…
araba model	00000000100000100100100010010000000…
araba fiyatlari	11001100001000010010101010101111111…
araba modifiye	00000001000000001000000001000000001…
…	…

All frequencies of keywords are computed and then keywords are eliminated from the set K if they have a frequency under the standard deviation of the average of keyword frequencies. A new set which is the subset of K can be shown as $K' = \{k_i \mid \text{if } f_{k_i} > (A - \sigma)\}$ where A is the average of the frequencies of keywords and σ is the standard deviation of the frequencies of keywords. A is represented by the formula

$$A = \frac{\sum_{i=1}^{m} f_{k_i}}{m}$$

where m shows the number of keywords and σ is computed by the formula

$$\sigma = \sqrt{\frac{\sum_{i=1}^{m} (f_{k_i} - A)^2}{m-1}}.$$

Table 4.4 shows an example for the result of the keyword frequencies. Higher frequencies mean that these keywords are strongly related to that product.

Table 4.4 An example of keyword frequencies for the product “araba”, from 250 web pages

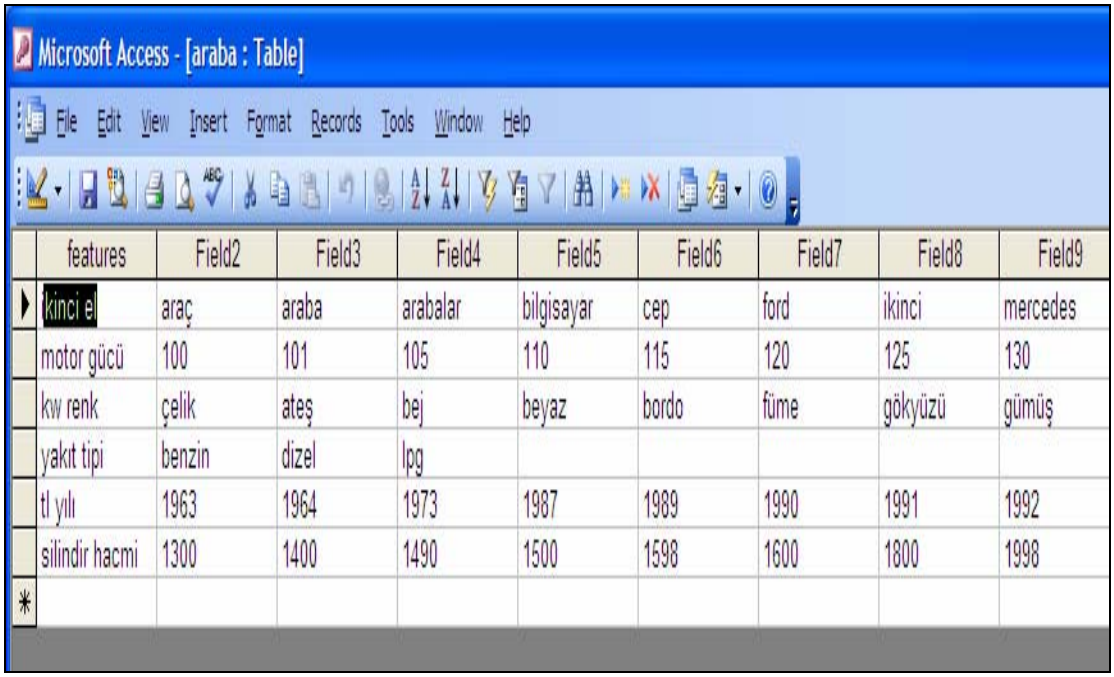
Mostly related	Related	Loosely Related
2 el araba satış 108	araba servisi 14	at yarışı tahminleri 6
araba aksesuarı 108	araba sevdası 13	at yarışı tüyo 6
araba aksesuarları 108	araba siteleri 17	at yarışı 6
araba fiyatları 108	araba vergi 17	atari 6
araba ilanlar 108	araba vergileri 13	atm 6
araba kiralama 108	araba vergisi 13	atv 6 (type of motor?, or tv?)
araba modelleri 108		atyarışı 6
araba fiatları 13		atölye kazaz 6
araba films 13		
araba fiyatlari 19		

Members of the set P are compared with the members of set K’. The words in the intersection set of these sets define the importance of the features and values. If ki∈ K’ and ki∈ P then ki is certainly a feature of product P. If ki∈ K’ and ki∈ F then ki is certainly a value of feature F.

Main programming codes of important parts of FERbot can be seen at Appendices section of thesis.

4.2.2.4 Database of FERbot

Database of FERbot includes all web page addresses and web pages, keyword files, keyword document matrix file, detagged web pages, input file and files of frequencies of sequential word groups. After extracting features and values of product, FERbot stores indexed data into tables. Columns of a table represent the features of a product and rows of a table represent the values of a feature of a product. Also, frequencies of these features and values are kept in other tables. Figure 4.22 shows a section from a table of the database where extracted data are the entries of database.



features	Field2	Field3	Field4	Field5	Field6	Field7	Field8	Field9
ikinci el	araç	araba	arabalar	bilgisayar	cep	ford	ikinci	mercedes
motor gücü	100	101	105	110	115	120	125	130
kw renk	çelik	ateş	bej	beyaz	bordo	füme	gökyüzü	gümüş
yakıt tipi	benzin	dizel	lpg					
tl yılı	1963	1964	1973	1987	1989	1990	1991	1992
silindir hacmi	1300	1400	1490	1500	1598	1600	1800	1998

Figure 4.22 A section from the table of product “araba”.

4.2.3 User Interface of FERbot

User interface is the last part of FERbot to interact with the customers. Customers see a basic graphical user interface including, a combo box to select a product and a submit button to retrieve the features. After accepting query, FERbot retrieves related answer from the database and sent back to user. In the first retrieval, customers just see the features of the selected product. By selecting any feature from the list they may see the values of the selected feature. This user interface is just designed for an example. Customers also will be enabled to write their queries by using a text box.

User interface is the place for presenting the database. The main page shows the product names in a combo box that refer to the table names as shown in Figure 4.23 and 4.24.

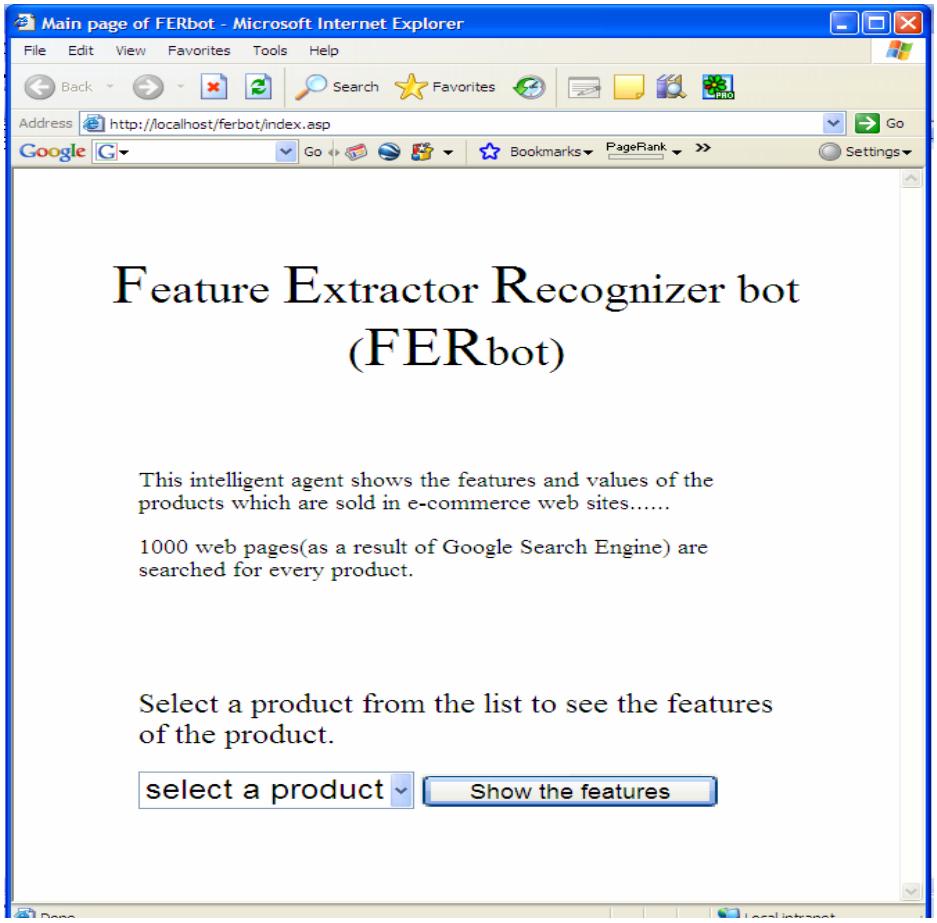


Figure 4.23 Main page of user interface for customers.

User selects a product to see the features of that product as seen in Figure 4.28. According to the product that is selected from main page, features of the product are displayed at the “features page” as shown in Figure 4.29. Product name is considered as a parameter which is used by a SQL query to select the rows and to project the fields. This query finds the row that refers to the feature name and gets the all fields that show the values of this feature.

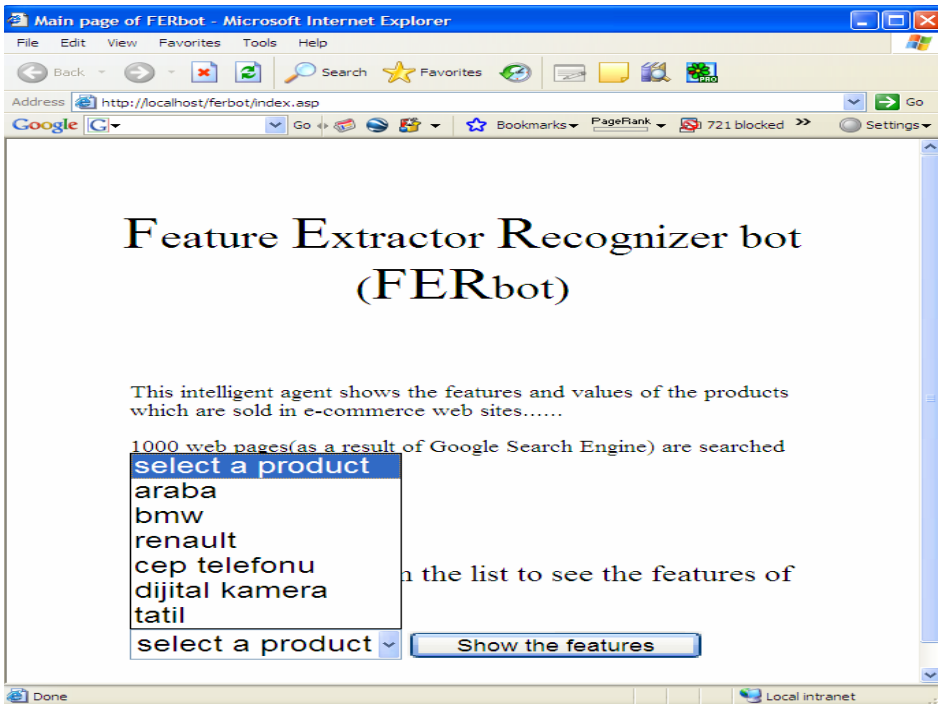


Figure 4.24 Customer selects a product from the list.

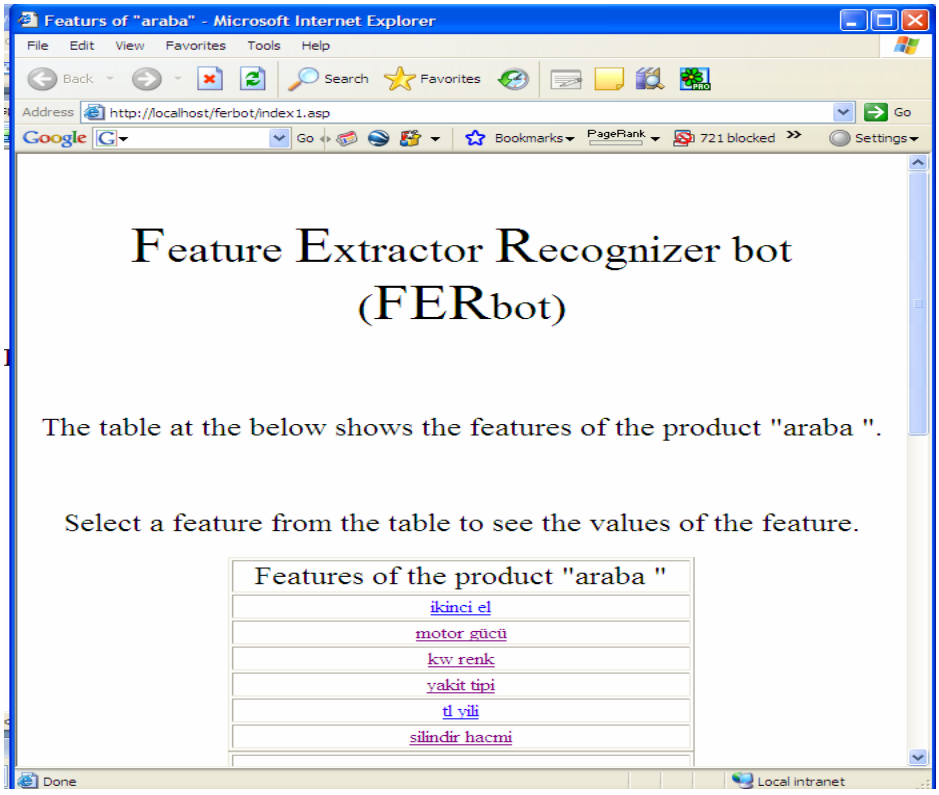


Figure 4.25 Customers see the features of selected product.

Last step for a customer to see the values of a feature is selecting a feature from the list in Figure 4.25. For example if a customer selects “yakit tipi” from the list, values of “yakit tipi” will be seen as “dizel” “benzin”, “lpg” as shown in Figure 4.26. If a user selects “renk” from the list, values of “renk” will be seen as “çelik” “beyaz”, “bej”,“bordo”, “fume” etc. as shown in Figure 4.27. If a customer selects “silindir hacmi” from the list, values of “silindir hacmi” will be seen as “1300” “1400”, “1490”,“1500”, “1598” etc. as shown in Figure 4.38.

Figure 4.26 Values of feature “yakit tipi”.

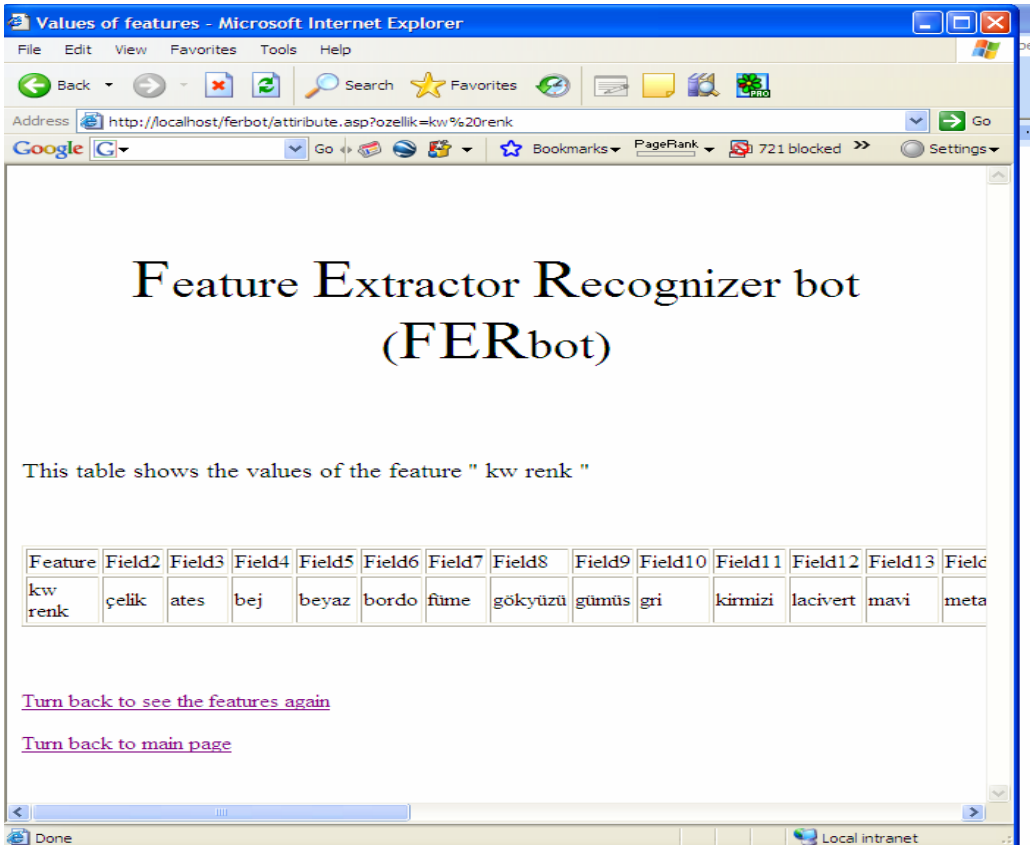


Figure 4.27 Values of feature “renk”.

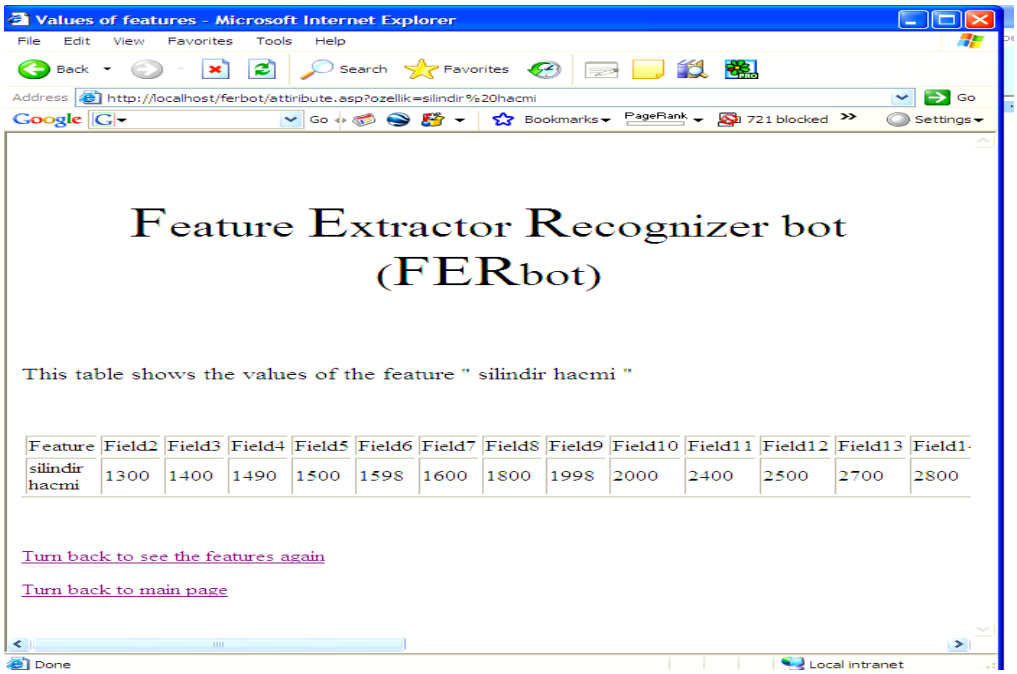


Figure 4.28 Values of feature “silindir hacmi”.

CHAPTER FIVE
FERbot FOR DIFFERENT RESOURCES

5.1 Domain Specific and Domain Independent Resources

In this section we try to show how FERbot produces different results according to the type of resource of web pages. Most of the wrappers are constructed for domain specific resources. For that reason, they get the semi structured or structured web pages as input set to extract the information. As we presented in section 3.5, they mainly segment the page layout and extract the labeled information. FERbot is ready for every kind of web pages but if FERbot gets semi structured or structured web pages as input set, it produces perfect results.

FERbot uses the web pages that are obtained from the addresses provided by Google search engine without defining type of the web pages. It means that FERbot certainly works over independent resources. In this respect, it is not important whether a web page is labeled or unlabeled. We evaluated our approach for some different types of products such as “araba, “cep telefonu”, “bmw”, “dijital kamera”, “renault”, “renault broadway”, “renault broadway 1.4 rn”, “mercedes”, “kitap”, “tatil”, “nokia”, “nokia 8810”, “televizyon”, “villa” and etc. Results of the products “araba”, “renault”, “bmw” and “villa” are given in following sections.

To see the difference between domain specific and domain independent resources we used FERbot for domain specific resources like www.emlak.turyap.com to obtain the features of product “villa”.

We explain the process that FERbot uses to obtain the features and values of “araba” in detail, but we just present some other results by tables.

5.2 Features and Values of Araba

FERbot collects the data from the web to extract the features of the product before presenting it to the customers. FERbot applies a regular process (collecting information, knowledge modeling and presenting data) for every product. An application of FERbot is about the product “araba”. “Araba” means car in Turkish language. Web pages that include information about “araba” were obtained from independent domains. Also web pages are in labeled or unlabeled form.

The first step of FERbot is to find the web pages related to “araba” from Internet using Google search engine. Every search-result of Google includes the web site name, web site address and a small summary including the queried word “araba”. Google uses different styles to insert hyperlinks for web site names, addresses and summaries. To leave alone the links showing the web site address of the query “araba”, Google’s search results are parsed. Hyperlinks of web page addresses about “araba” are written into a text file to visit and download them one by one.

FERbot creates a collection which consists of 925 web pages related to “araba”, to find the frequencies of sequential *n*-word(s) groups of these web pages. FERbot applies the regular process which is mentioned in section 4.2.1 and and section 4.2.2. After this process, FERbot creates an input file called S which is the list of words obtained from collected web pages.

FERbot analyzes the input file and writes all one-word, two-words,……, six-words groups (for example a two-words group means two sequential words considering order is important) and their frequencies into separate text files. Sections of three of these files about “araba” are presented in Table 5.1.

Table 5.1. Sections of SWG^1, SWG^2, SWG^3 and their frequencies, $f_{SWG\ 1}$, $f_{SWG\ 2}$, $f_{SWG\ 3}$.

A Section from SWG^1 and f_{SWG^1}		A Section from SWG^2 and f_{SWG^2}		A Section from SWG^3 and $f_{SWG\ 3}$	
SWG_i^1	$f_{SWG\ i}^1$	SWG_i^2	$f_{SWG_i^2}$	SWG_i^3	$f_{SWG\ i}^3$
motor	2358	motor gücü	2214	motor gücü 100	163
motora	1	motor i6	1	motor gücü 101	16
motorbike	2	motor ima	3	motor gücü 102	13
motorbisiklet	4	motor kategoride	1	motor gücü 103	15
…….	……	…….	……	…….	……

FERbot considers the first sixty unique words which have highest frequency as the most related words for product “araba”. It searches relevant features among these sixty words (to get better solutions number sixty can be increased). FERbot extracts the features which includes two-words by using the algorithm mentioned in Figure 4.25. For an example, the frequency of motor, $f_{SWG\ i}^1$, and the frequency of two-words group beginning with the word “motor”, $f_{SWG\ i}^2$, are compared, if they are approximately equal, feature name becomes two-words group (motor gücü) instead of “motor”. The frequency of “motor gücü”, $f_{SWG\ i}^2$, and the frequency of three-words group beginning with “motor gücü”, $f_{SWG\ i}^3$, are compared, if they are not equal or not approximately equal, feature name remains same as “motor gücü” and values of this feature becomes last words (..,100,101,102,103,…) of three-words group, SWG_i^3. Words having high frequencies that do not take a place in the “two-words feature group” are considered as single word feature and values of them were associated like values of two-words groups.

Sections from frequencies of sequential word groups that refer to features of “araba” and values of these features are presented in, Table 5.2, Table 5.3, Table 5.4 and Table 5.5. These sequential word groups are sorted in descending order. Higher frequency shows the strong relationship between the feature and its value. Values of features are bold colored in tables.

In Table 5.2, there are also two words groups different than “yakıt tipi”. It is seen that frequency of “yakıt türü benzin” which is in italic form has relatively higher frequency. “tipi” and “türü” have same meaning in Turkish that mean “type”. FERbot may extract these kind of relations from a text document by searching the order of sequential words.

Table 5.2 Section from two words groups starting with “yakıt tipi” and their values with frequencies in descending order.

Feature		**Value**	**Frequency**
yakıt	tipi	**benzin**	1627
yakıt	tipi	**dizel**	546
yakıt	tipi	**lpg**	18
yakıt	tipi	**kiralık**	3
yakıt	*türü*	***benzin***	*51*
yakıt	doldurabilir	**doldurmayabilirsiniz**	2
yakıt	depolama	**sahaları**	1
yakıt	siteye	**son**	1
yakıt	tüketimi	**100**	1
yakıt	türü	**farketmez**	1
yakıt	tasarrufu	**mini**	1
yakıt	benzin	**4**	1

Table 5.3 Section from two words groups starting with “silindir hacmi” and their values with frequencies in descending order.

Feature		Value	Frequency
silindir	hacmi	**1600**	160
silindir	hacmi	**2000**	63
silindir	hacmi	**2500**	44
silindir	hacmi	**1400**	41
silindir	hacmi	**1598**	40
silindir	hacmi	**1500**	39
silindir	hacmi	**1300**	31
silindir	hacmi	**1998**	28
silindir	hacmi	**1490**	27
silindir	hacmi	**1800**	25
silindir	hacmi	**2400**	16
silindir	hacmi	**2700**	16
silindir	hacmi	**2800**	16
silindir	hacmi	**1200**	14
silindir	hacmi	**1590**	14
silindir	hacmi	**1700**	13
silindir	hacmi	**5700**	12
silindir	hacmi	**1900**	11
silindir	hacmi	**1390**	10
silindir	hacmi	**1599**	10
silindir	hacmi	**4000**	10
silindir	hacmi	**4600**	10
silindir	hacmi	**5000**	10
silindir	hacmi	**1581**	9
silindir	hacmi	**1948**	9
silindir	hacmi	**3500**	9
silindir	hacmi	**6000**	8
silindir	hacmi	**1499**	7

Table 5.4 Section from two words groups starting with “motor gücü” and their values with frequencies in descending order.

Feature		Value	Frequency
motor	gücü	**75**	191
motor	gücü	**90**	164
motor	gücü	**100**	163
motor	gücü	**110**	155
motor	gücü	**80**	86
motor	gücü	**150**	75
motor	gücü	**120**	61
motor	gücü	**70**	58
motor	gücü	**105**	56
motor	gücü	**130**	52
motor	gücü	**85**	47
motor	gücü	**115**	45
motor	gücü	**136**	36
motor	gücü	**95**	29
motor	gücü	**65**	27
motor	gücü	**125**	26
motor	gücü	**225**	26
motor	gücü	**98**	26
motor	gücü	**140**	24
motor	gücü	**200**	23
motor	gücü	**86**	23
motor	gücü	**220**	22
motor	gücü	**300**	22
motor	gücü	**190**	21
motor	gücü	**163**	20
motor	gücü	**250**	19
motor	gücü	**60**	19
motor	gücü	**180**	18

Table 5.5 Section from one word groups starting with “renk” and their values with frequencies in descending order.

Feature	Value	Frequency
renk	**beyaz**	455
renk	**siyah**	245
renk	**gri**	233
renk	**kırmızı**	152
renk	**yeşil**	145
renk	**gümüş**	141
renk	**mavi**	136
renk	**lacivert**	130
renk	**bordo**	74
renk	**sarı**	64
renk	**bej**	58
renk	**füme**	57
renk	**metalik**	55
renk	**ateş**	52
renk	**farketmez**	52
renk	**çelik**	38
renk	**gökyüzü**	26
renk	**nil**	26
renk	**seçenekleri**	18
renk	**başak**	15
renk	**silindir**	15
renk	**kapri**	12
renk	**sahra**	12
renk	**yakut**	12
renk	**bronz**	9
renk	**buz**	9
renk	**çizgi**	6
renk	**koyu**	6

FERbot searches meta-keyword-tags to see the relation between the product name “araba” and keywords of web pages about “araba” as mentioned in Section 4.2.2.3. FERbot considers keywords having a frequency over the average of keyword frequencies are potential features or values of the product “araba”. These potential features and values are compared with the results that are found by the new algorithm to refine the results.

Table 5.6 Values and features of product “araba”.

FEATURE	**VALUE**
'ikinci el' (second hand)	araç (vehicle), araba (car), arabalar (cars), bilgisayar (computer), cep (mobile), ford (ford), ikinci (second), mercedes (mercedes), opel (opel), oto (auto), otomobil (automobile), otomotiv (automotive), Peugeot (peugeot), renault (renault), satılık (for sale)
'motor gücü' (motor power)	100, 101, 105, 110, 115, 120, 125, 130, 136, 140, 150, 163, 180, 190, 200, 220, 225, 235, 250, 300, 60, 65, 70, 75
'yakıt tipi' (fuel type)	benzin (benzine), dizel (diesel), lpg (lpg)
'silindir hacmi' (cylinder volume)	1300, 1400, 1490, 1500, 1598, 1600, 1800, 1998, 2000, 2400, 2500, 2700, 2800, cm
'yılı' (production date)	1963, 1964, 1973, 1987, 1989, 1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006
'renk' (color)	çelik (steel), ateş (fire), bej (beige), beyaz (white), bordo (maroon), fume (smoke-colored), gökyüzü (sky), gümüş (silver), gri (grey), kırmızı (red), lacivert (dark blue), mavi (blue), metalik (metallic), nil (Nile green), sarı (yellow), siyah (black), yeşil (green)
'ytl' (currency unit)	1, 10, 11, 12, 13, 14, 15, 16,17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 3, 30, 4, 5

After extracting and refining features and values of product, FERbot stores indexed data into tables. Columns of a table represent the features of a product and rows of a table represent the values of a feature. Also, frequencies of these features and values are kept in other tables.

Finally if any user selects a product like “araba” from FERbot’s user interface, he or she will see a result as in Table 5.6 where first column represents the main features of the product “araba” and second column represents the values of the features. Also frequencies of these features and values are kept in another file. By changing the threshold of any frequency, number of features and the range of the features (number of values) can be increased or decreased.

5.3 Features and Values of Renault and BMW

To show the performance of FERbot we show the features and values of Renault and BMW which are different types of product “araba” (car)in Table 5.7 and Table 5.8. These results obtained under same thresholds from independent domain resources. We used same features in both tables for better comparisons. “yakıt tipi” which means “fuel type in Turkish is the feature of both cars that has same values. This is the expected situation for all cars. We know that BMW has more powerfull motors and higher cylinder volumes than Renault has. It is observed for the values of features “motor power” and “cylinder volume” as shown in Table 5.7 and Table 5.8. Color (“renk”) features of these two kinds of cars almost have the same values but the frequencies may change.

FERbot may extract useful information about the relations between different products which are in same category. FERbot may show the range of the values of these products at the same time.

Table 5.7 Values and features of product “Renault”.

FEATURE	VALUE
motor gücü	100, 103, 105, 110, 115, 120, 135, 140, 60, 65, 70, 72, 75, 80, 85, 90, 95, 98
yakıt tipi	benzin, dizel, lpg
silindir hacmi	1149, 1200, 1300, 1390, 1397, 1398, 1400, 1461, 1490, 1500, 1590, 1598, 1600, 1700, 1721, 1870, 1898, 1900, 1995, 1998, 2000, 2500
renk	çelik, ateş, bej, beyaz,bordo, fume, gümüş, gri, kırmızı, lacivert, mavi, sarı

Table 5.8 Values and features of product “BMW”.

FEATURE	VALUE
motor gücü	100, 101, 105, 110, 115, 120, 125, 130, 136, 140, 163, 180, 190, 200, 220, 225, 235, 250, 300, 60,65
yakıt tipi	benzin, dizel, lpg
silindir hacmi	1596, 1598, 1600, 1796, 1798, 1800, 1991, 1995, 1998, 2000, 2200, 2498, 2500, 2800, 2993, 2998, 3000, 4400, 4500
renk	çelik, ateş, beyaz, bordo, füme, gümüş, gri, il, kırmızı, lacivert, mavi, metalik, opsiyonları, seçenekleri, siyah, yeşil

5.4 Features and Values of Villa

To show how FERbot learns features and values from domain specific resources, we only used web pages from the domain “www.emlak.turyap.com”. Results are presented in Table 5.9 and Table 5.10. The difference between these tables is the threshold of frequencies of values of features. When we decrease the threshold we obtain the results inTable 5.10 which are better than results in Table 5.11.

Table 5.9 Values and features of product “villa”.

FEATURE	**VALUE**
oda sayısı	2, 3, 4, 5, 6
salon sayısı	1, 2
kiralık	emlak, emlaklar, villa
marmaris	armutalan, banka, beldibi, içmeler, mahalle, muğla, turunç
site	haritası, içinde
kat	1, 2, 3, 4
bahçe	içinde, var
bölge	çengelköy, bolu, guzelyali, marmaris
mahalle	akpinar, bahcelievler, merkez

Table 5.10 Values and features of product “villa” with a different treshold.

FEATURE	**VALUE**
oda sayısı	3, 4, 5
salon sayısı	1, 2
kiralık	emlak, emlaklar, villa
adres	
fiyat	
ilçe	
kat	1, 2, 3, 4
mutfak	
banyo	
bahçe	var
alan	
mahalle	merkez

CHAPTER SIX
CONCLUSIONS AND FUTURE WORK

6.1 Tools for FERbot.

For applications of FERbot, generally Intel Pentium IV 3.0 GHz processor, and 512 Mb RAM were used. MatLab is the platform to write the codes of collecting the information from Internet to detag them, preparing the input file and defining the features and values. C programming language is used to find the word frequencies. Dreamweaver is used to design the web page to present the results. Active Server Pages (ASP) is used to interact with database. Microsoft Access is used for database management.

The time is approximately 30 hours for a product to define its features and values. Most of the time is wasted to remove stopwords. A word-space-word file for a product is almost between 3MB and 8MB. Approximately 20% of the files are consisting of stopwords.

6.2 Conclusion

If intelligent agents especially “shop-bots that derive semantics from semantic web sites” have capability to understand the non-standard web sites, they will complete their job to find the product features at the same time. Such agents may suggest new product features as important keywords to be taken into account for refined searches. If proposed new features are accepted by the user, the agent will search and shop in Internet accordingly. These agents searching the desired product in Internet, may record possible new features at the same time and inform the central server from which they are dispatched and where the product feature files are maintained.

In this thesis, we were interested in with the “Business to Costumer” part of e-commerce. Our study take a place at the pre-sale part of trade cycle where pre-sale is the first step of the trading cycle consisting of searching and negotiation.

Our new algorithm and work for extracting new features of products from e-shopping web-sites are realized as an agent based application. One part of e-commerce agents deal with the subject of information gathering which is the act of collecting information. This is the main idea of our study which is collecting information from Web and extracting meaningful relations between products and their properties to inform customers.

Our agent is a kind of e-commerce agent (intelligent agent, shopping agent, information agent, bot) where its percepts are commercial web pages and customer queries. The main actions that our agent performs are information gathering, discovering knowledge, filtering, indexing, analyzing, and modeling data, and constructing knowledge base. The environments for our agent are Internet and customers.

In this work, we demonstrated that structure and semantics of a text can be systematically extracted by lexical analysis. We presented a new algorithm to use the relative frequencies of sequential word groups for information extraction. We have shown that sequential word group (single, two-words,…., n-words) frequencies are effective to derive semantics from e-commerce sites to define product in terms of features and values. In our study we used the sequential word groups to create a hierarchical classification. Products are classified in hierarchical order where features are sub groups of products and values are sub groups of features.

In our information Extraction System, input set is the collection of e-commerce web pages. The processing part of our system creates Sequential Word Groups from e-commerce web pages and by extracting features and values of products from these e-commerce web pages, system constructs the Knowledge Base automatically. Finally, system shares the structured information with customers and other intelligent

agents that may use the same ontology. We use WWW as a knowledge discovery source and our Information Extraction system can be thought as converting information from text into database elements.

Extracted data must be semantically classified into a group. The extracted information sometimes becomes a word, word compound, adjective phrase or semantically combined sentences. In our study we used the sequential word groups to create a hierarchical classification. Our Information Extraction system uses some extraction patterns which are constructed automatically.

Automatic discovering of relations between features and values bases on lexical and syntactic analysis of the text. A sequential word group is constructed by words which can be in a meaningfull or non-meaningfull structure. In our approach, machine learning algorithm is applied over e-commerce web pages to learn the relations between products and features of products. Semantic relations of these features and values can occur between nouns, nouns and adjectives, nouns and verbs or verb adverbs. Level of a relation in a text may be seen in word-level, phrase level (noun-verb) or concept level. In order to apply the extraction patterns FERbot processes the input text with syntactic and semantic tagging.

In this study, to evaluate our new algorithm, we developed a software called FERbot (Feature Extractor Recognizer bot). FERbot learns the features of products automatically by extracting sequential word groups from hundreds of web pages and finds out the values of these features to automatically construct a knowledge base for products.

We may give the general properties of FERbot in the following list.

- FERbot is a software suit.
- FERbot is a crawler.
- FERbot is a download manager.
- FERbot is an html converter.

- FERbot is a software that normalizes texts.
- FERbot is a text analyzer.
- FERbot is an Information Extractor.
- FERbot constructs Knowledge Bases automatically.
- FERbot is an Intelligent Agent.

FERbot is a kind of download manager that downloads Internet files from Internet. Download managers scan a web page and download all files (image, sound, movie, achieve files etc.) linked to it, with customizable filters. FERbot is an html converter that gets html pages as input and produces a plain text as output. Like other html converters the aim of FERbot is extracting data or text from html pages to import data into a database or to analyze the text. The main jobs of FERbot as a text normalizer are removing punctuation, converting letters into case sensitive form.

There are three main parts of FERbot system which are Information Gathering part, User Interface and Knowledge Modelling. At the Information Gathering part, FERbot visits Internet pages and stores them into a local machine to work over these web pages. FERbot knows which page to visit by filtering the addresses of web pages from Google's main page. To obtain more specific result page, FERbot sends the product name as a query to Google Search engine by customizing some searching options. At the Knowledge Modelling part, FERbot parses these downloaded html files to have a structured form for data. FERbot applies some statistical analysis and hierarchical algorithms about the collected data to define the product features. The algorithm we developed which is based on sequential word groups will be the core of the system FERbot. At the user interface part, FERbot allows customers to present their queries. Customers may select the query from the list or they may write their queries. After accepting query, related answer is retrieved from database and sent back to user.

FERbot is a kind of wrapper that extracts information from labeled and unlabeled documents. The first advantage of FERbot is its own algorithm. It can be used for domain specific or domain independent resources. Second advantage is working with

the labeled or unlabeled web pages. The input set of FERbot can be any kind of web pages whether it is not designed by XML. FERbot understands the features of products by finding their frequencies. FERbot uses labels to extract the keywords of web pages to construct the keyword-document matrix for refining the results of features of products. We may say that most of the wrappers are constructed for domain dependent and labeled documents but FERbot does not imply these kinds of restrictions. FERbot extracts information into a hierarchical structure. This study also presents new possibilities towards information extracting from structured or unstructured html web sites.

FERbot interacts with e-commerce information resources that are designed to be used by customers. In our study we considered that customers certainly know the product they buy or they certainly know the product that they want to be informed about. For that reason, our intelligent agent FERbot uses feature based filtering for product brokering. The difference from other agents using feature based filtering, is bringing the product features and values as a result of query.

Most of the wrappers are constructed for domain specific resources. For that reason, they get the semi structured or structured web pages as input set to extract the information. They mainly segment the page layout and extract the labeled information. We have shown how FERbot produces different results according to the type of resource of web pages. FERbot is ready for every kind of web pages but, our algorithm produces perfect results if the input set is mostly full of semi structured web pages instead of unstructured web pages.

Our new algorithm is an alternative for the current statistical and machine learning techniques for information detection and classification. Also, text miners can use our algorithm for knowledge discovery within the concept of statistical data mining techniques to operate the information extracted from texts.

FERbot may extract useful information about the relations between different products which are in same category. FERbot may show the range of the values of these products at the same time.

FERbot may be used for cryptographic solutions of text analyzing. FERbot creates the frequencies of sequential word groups. For this reason, FERbot certainly knows which word comes after a word in a sentence by their group frequencies.

E-commerce applications change terminology of trading. For example, stock is not a feature of a product but most of the advertisements explain whether product is available in stock or not. FERbot considers “stock” can be a feature of any product if it is frequently used in the advertisement. Finally, FERbot produces different values for the features like “stock”.

6.3 Future work

One of the main future works will be based on k-Nearest Neighbor and Naive Bayesian algorithms to define the values of features without running the extraction algorithm. These algorithms are mostly used for classification of text documents. In Vector Space Models, features of a product are written as a vector of parent class. By using the matrix multiplication, weights of other features can be computed. The purpose is to make a hierarchical structure of data set in top-down order. By utilizing known (or artificial) hierarchical structure, the classification problem can be decomposed into a set of smaller problems corresponding to hierarchical splits in tree structure (Zhou and Cho H., (2001).

After constructing the table of features of a product and values of features, the purpose is updating values without doing all process of FERbot. For that reason, we have to obtain a value vector in the form of $V(v_1, v_2, v_3, \ldots\ldots, v_i,)$. This vector can be obtained from information etraction process of FERbot. After obtaining this vector, by using the combination of two algorithms k-NN and NB, we can find which Feature is the main class of this V vector.

For the algorithm of k-NN, a matrix is constructed where columns show the Features and rows show the all Values of a product. A vector of values is multiplied with the columns of matrix to find the Euclidian distance. The better result defines the feature class of this Value vector.

For Naïve Bayesian algorithm, If $P(v_i \mid F)$ shows the probability of value v_i given that Feature F is known

$$P(v_i \mid F)$$

$$P(V \mid F) = \prod_i P(v_i \mid F)$$

$$P(V \mid F) = \frac{P(V \cap F)}{P(F)} \text{ and } P(F \mid V) = \frac{P(V \cap F)}{P(V)}$$

$$P(F \mid V) = \frac{P(F)}{P(V)} P(V \mid F)$$

A value vector is in Feature F or it is not in Feature F.

$$P(V \mid F) = \prod_i P(v_i \mid F)$$

$$P(V \mid \overline{F}) = \prod_i P(v_i \mid \overline{F})$$

Use Bayes' formula again

$$P(F \mid V) = \frac{P(F)}{P(V)} \prod_i P(V \mid F)$$

$$P(\overline{F} \mid V) = \frac{P(\overline{F})}{P(V)} \prod_i P(V \mid \overline{F})$$

Divide first one by the second

$$\frac{P(F \mid V)}{P(\overline{F} \mid V)} = \frac{\frac{P(F)}{P(V)} \prod_i P(V \mid F)}{\frac{P(\overline{F})}{P(V)} \prod_i P(V \mid \overline{F})}$$

Find ln of both sides

$$\ln \frac{P(F \mid V)}{P(\overline{F} \mid V)} = \ln \frac{P(F)}{P(\overline{F})} + \sum \ln \frac{P(v_i \mid F)}{P(v_i \mid \overline{F})}$$

If $\ln \frac{P(F \mid V)}{P(\overline{F} \mid V)} > 0$ than this vector is in that feature

FERbot is designed as an intelligent agent to inform the customers about the product features and values. In this top-down design, product is located at the top and values are located at the bottom. FERbot may find a sub category of values to expand the hierarchical structure.

The semantic network of products can be constructed by FERbot. After defining the characteristics of products, the relations between these products will be obtained by FERbot.

The algorithm will be applied for electronic commerce web sites which are including a language different than Turkish language. To achieve this job, the stopword list should be changed and some syntactical arrangements should be done.

To increase the efficiency and performance of algorithm, methods such as parallel processing and computers having more powerful processors will be used.

REFERENCES

Ahmed S, T., Vadrevu S., Davulcu H., (2006). DataRover: An Automated System for Extracting Product Information, *In Advances in Web Intelligence and Data Mining,* Volume 23, pp. 1-10, Springer Berlin/Heidelberg Publishers.

Ansari S., Kohavi R., Mason L., Zheng Z., (2000). Integrating E-commerce and Data Mining: Architecture and Challenges, *WEBKDD'2000 workshop on Web Mining for E-Commerce -- Challenges and Opportunities.*

Arasu A., Garcia-Molina H., (2003). Extracting structured data from Web pages, *ACM SIGMOD,* pp. 337 – 348.

Baumgartner R., Eiter T., Gottlob G., Herzog M. Koch C., (2005) Information Extraction for the Semantic Web, *Lecture Notes in Computer Science – Reasoning Web*, Vol. 3564, pp. 275-289.

Berry M.W., Browne M., (1999). *Understanding Search Engines, Mathematical Modelling and Text Retrieval,* Philadelphia.

Berry M., Dumais S., Brien O.G., (1995). *Using linear algebra for intelligent information retrieval*, SIAM Review.

Berry M.W., Giles J.T., Wo L., (2003). GTP (General Text Parser) Software for Text Mining, *Statistical Data Mining and Knowledge Discovery, H. Bozdogan (Ed.),* CRC Press, Boca Raton, (2003), pp. 455-471.

Beveren J. V., (2002). A model of knowledge acquisition that refocuses knowledge management, *Journal of Knowledge Management,* Volume: 6, Issue: 1, pp. 18 – 22.

Bikel D. M., Schwartz R., Weischedel R. M., (1999). An algorithm that learns what's in a name, *Machine Learning*, vol. 34, pp. 211-231.

Borkar V., Deshmukh K., Sarawagi S., (2001). Automatic segmentation of text into structured records, *International Conference on Management of Data Archive Proceedings of the 2001 ACM SIGMOD international conference on Management of data*, ISSN:0163-5808, pp. 175 – 186.

Brynjolfsson E., Dick A.A., Smith M.D., (2003). Search and Product Differentiation at an Internet Shopbot, *A research and education initiative at the MIT Sloan School of Management*, Paper no: 194.

Burget R., (2004). Hierarchies in HTML Documents: Linking Text to Concepts, *15th International Workshop on Database and Expert Systems Applications*, pp.186-190.

Burget R., (2005). Visual HTML Document Modeling for Information Extraction, CZ, FEI VŠB, ISBN 80-248-0864-1, Ostrava pp.17-24.

Califf, M., Mooney, R. (1997). Relational learning of pattern-match rules for information extraction, *Working Papers of the ACL-97 Workshop in Natural Language Learning,* pp. 9–15.

Can F., (2006). Turkish Information Retrieval: Past Changes Future, *Advances in Information Systems*, Springer Berlin Heidelberg New York, pp.13-22.

Chan C.W., (2004). The Knowledge Modeling System and its application Electrical and Computer Engineering, *Canadian Conference on Volume 3*, Issue , 2- pp. 1353-1356.

Chang C. H., Hsu C. N., Lui S. C., (2003). Automatic information extraction from semi structured Web pages by pattern discovery, *Decision Support Systems 35*, pp. 129– 147.

Chang C. H., Kayed M., Girgis M.R., Shaalan K.F., (2006). A Survey of Web Information Extraction Systems, *Knowledge and Data Engineering*, Vol. 18, No. 10, pp. 1411-1428.

Chaudhury A., Kilboer J., (2002). E-business and E-commerce Infrastructure, *Technologies Supporting the E-business Initiative (Chapter 1)*, McGraw Hill.

Chuang S. L., Chien L.F., (2005). Taxonomy generation for text segments:A practical web-based approach, *ACM Transactions on Information Systems,* pp. 363-396.

Craven M., DiPasquo D., Freitag D., McCallum A., Mitchell T., Nigam K., Slattery S., (2000). Learning to construct knowledge bases from the World Wide Web, *Artificial Intelligence*, 118, pp. 69–113.

Crescenzi R., Mecca G., Merialdo P., (2001). RoadRunner: Towards Automatic Data Extraction from Large Web Sites, *27th International Conference on Very Large Data Bases.*

Doorenbos R. B., Etzioni O., Weld D.S., (1997). A Scalable Comparison-Shopping Agent for the World-Wide Web, Department of Computer Science and Engineering University of Washington.

Etzioni O., Weld D. S., (1994). A Softbot-Based Interface to the Internet, Communications of the ACM, pp. 72-76.

Etzioni O., Weld D. S., (1995). Intelligent Agents on the Internet: *Fact, Fiction, and Forecast IEEE Expert*, pp. 44-49.

Feldman S., Yu E., (1999). Intelligent agents: A Primer, *Searcher*, October 1999, Volume 7, No. 9.

Filatova E., Hatzivassiloglou V., (2003). Domain-Independent Detection, Extraction, and Labeling of Atomic Events, *Proceedings of the Fourth International Conference on Recent Advances in Natural Language Processing.*

Flesca S., Manco G., Masciari E., Rende E.,Tagarelli A., (2004). Web wrapper induction: a brief survey, *AI Communications*,Vol.17/2.

Franklin S., Graesser A., (1996). Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents, *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages*, Springer-Verlag, 1996.

Freitag D., (1998). Machine Learning for Information Extraction in Informal Domains, *Ph.D. thesis, Computer Science Department*, Carnegie Mellon University, Pittsburgh.

Freitag, D., (1998). Information extraction from html: Application of a general learning approach. *Proceedings of the 15th Conference on Artificial Intelligence (AAAI-98)*, pp. 517–523.

Freitag, D., (2000). Machine Learning for Information Extraction in Informal Domains. *Machine Learning,* 39(2/3), pp.169-202.

Grishman R., (1997). Information Extraction: Techniques and Challenges, *Materials for Information Extraction,* Springer-Verlag.

Gomes G., Segami C., (2007). Semantic Interpretation and Knowledge Extraction, *Knowledge-Based Systems,* Volume 20, Issue 1, pp. 51-60.

Guttman R., Moukas A,. Maes P., (1998). Agent-mediated electronic commerce: a survey, *Knowledge Engineering Review 13(2)*, pp. 147-159.

Heaton, J., (2002). *Programming Spiders, Bots, and Aggregators in Java,* Sybex Book.

Hui B., Yu E., (2005). Extracting conceptual relationships from specialized documents, *Data & Knowledge Engineering,* Volume 54 , Issue 1, pp. 29 – 55.

Jennings N., Wooldridge M., (1998). Applying agent technology. In Jennings N, Wooldridge M (eds). *Agent technology: foundations, applications, and markets*, Springer, Berlin, pp 3-28.

Kalakota R., Whinston A., (1996). Frontiers of electronic commerce. Addison-Wesley, Reading, MA, Ch 16, pp. 595-628.

Kang D., Choi2 J., (2003). MetaNews: An Information Agent for Gathering News Articles On the Web, *Lecture Notes in Computer Sciences,* pp. 179-186 ISBN: 978-3-540-20256-1, Springer Berlin.

Karataş K., (2001). The Anatomy of an Internet Robot : bilBot®.. Meteksan Sistem ve Bilgisayar Teknolojileri A.Ş. Bilkent, Ankara, *Türkiye'de Internet konferansları VII*., http://inet-tr.org.tr/inetconf7/program/97.html.

Karanikas H., Theodoulidis B., (2002). Knowledge Discovery in Text and Text Mining Software, *Technical report, UMIST-CRIM*, Manchester.

Kernighan B. W., Pike R., (1999). *The Practice of Programming (Chapter 3),* Addison Wesley Professional.

Kim, J., Moldovan, D., (1995). Acquisition of linguistic patterns for knowledge based information extraction, *IEEE Transactions on Knowledge and Data Engineering*, 7(5), pp. 713–724.

Knoblock C.A., Lerman K., Minton S., Muslea I., (2000). Accurately and Reliably Extracting Data from the Web: *A Machine Learning Approach, IEEE Data Engineering Bulletin*, 23(4), pp.33-41.

Kohavi R., John G., (1997). Wrappers for Feature Subset Selection, *Artificial Intelligence journal*, special issue on relevance, Vol. 97, pp. 273-324.

Kohavi R., Provost F., (2001). Applications of Data Mining to Electronic Commerce, *Data Mining and Knowledge Discovery journal.*

Krupka, G., (1995). Description of the sra system as used for muc-6. *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, pp. 221–235.

Kucukyilmaz T., Cambazoglu B.B., Aykanat C., Can F., (2006). Chat mining for Gender Prediction, *Advances in Information Systems,* Springer Berlin Heidelberg New York, pp.274-283.

Lee W.P., (2004). Applying domain knowledge and social information to product analysis and recommendations: an agent-based decision support system, *Expert systems,* Volume 21, Number 3, pp. 138-148, Blackwell.

Lee C., C., Yang J., (2000). Knowledge Value Chain, *Journal of Management Development,* Volume: 19 Issue: 9 DOI: 10.1108/02621710010378228, pp. 783 – 794, MCB UP Ltd.

Leung H., He M., (2002). Agents in E-commerce: State of the Art, *Knowledge and Information Systems,* Pp. 257-282.

Moens M.F., (2006). *Information Extraction: Algorithms and Prospects in a Retrieval Context, (Chapter 1)*, pp. 1-22, Springer.

Mukherjee S, Yang G., Tan W., Ramakrishnan I.V., (2003). Automatic Discovery of Semantic Structures in HTML Documents, *7th International Conference on Document Analysis and Recognition.*

Muslea I., (1999). Extraction Patterns for Information Extraction Tasks: A Survey, The AAAI-99, *Workshop on Machine Learning for Information Extraction.*

Muslea I., Minton S., Knoblock C. A., (2001). Hierarchical Wrapper Induction for Semistructured Information Sources, *Autonomous Agents and Multi-Agent Systems,* Volume 4 , Issue 1-2, p.p 93 – 114.

Nwana H., S., (1996). Software Agents: An Overview, *Knowledge Engineering Review,* Volume: 11, no: 2, pp. 205-244.

Perkowitz M., Doorenbos R. B., Etzioni O., Weld D. S., (1997). Learning to Understand Information on the Internet: An Example-Based Approach, Journal of *Intelligent Information Systems,* 8(2), pp. 133-153.

Perrin P., Petry F.E., (2003). Extraction and representation of contextual information for knowledge discovery in texts, *Information Sciences 151*, pp. 125–152.

Radovanović M., Ivanović M., (2006). CatS: A classification-powered meta-search engine, *Advances in Web Intelligence and Data Mining,* volume 23 of Studies in Computational Intelligence, pp. 191–200, Springer-Verlag.

Rao L., Muata K., Bryson O., (2007). Towards defining dimensions of knowledge systems quality, *Expert systems with applications*, pp. 368-378.

Riloff, E., (1993). Automatically constructing a dictionary for information extraction tasks. *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI.93)*, pp. 811–816.

Riloff, E., Lorenzen, J., (1999). Extraction-based Text Categorization: Generating Domain-specific Role Relationships Automatically, *Natural Language Information Retrieval,* Tomek Strzalkowski, ed., Kluwer Academic Publishers.

Rokach L., Chizi B., Maimon O., (2006). Feature Selection by Combining Multiple Methods, *2nd Workshop Algorithmic Techniques for Data Mining.*

Rokach L., Romano R., Chizi B., Maimon O., (2006). A Decision Tree Framework for Semi Automatic Extraction of Product Attributes from the Web, 4th Atlantic Web Intelligence Conference, *Advances in Web Intelligence and Data Mining,* pp. 201-210, Springer.

Rui Y.,Liu, Z., (2004). ARTIFACIAL: Automated Reverse Turing test using FACIAL features, *Multimedia Systems,* Volume 9, Number 6, June 2004, pp. 493-502(10), Springer.

Rus D., Subramanian D., (1997). Customizing Information Capture and Access, ACM Trans, *Information Systems,* 15(1), pp. 67-101.

Russell S., Norvig P., (1995). *Artificial Intelligence, A Modern approach,* Prentice Hall, New Jersey.

Seo H., Yang J., Choi J., (2001). Building Intelligent Systems for Mining Information Extraction Rules from Web Pages by Using Domain Knowledge, *IEEE International Symposium on Industrial Electronics,* pp. 322-327, Pusan, Korea.

Shamsfard M., Bardoroush A.A., (2004). Learning Ontologies from Natural Language Texts, *International Journal of Human-Computer Studies,* Volume 60, Issue 1, pp. 17-63.

Smith, D.C., Cypher A., Spohrer J., (1994). KidSim: Programming Agents Without a Programming Language, *Communications of the ACM,* 37, 7, pp. 55-67.

Smith, M.D. and E. Brynjolfsson (2001). Consumer Decision-Making at an Internet Shopbot: Brand Still Matters, *Journal of Industrial Economics,* Volume 49: pp. 541-557.

Soderland, S., Fisher, D., Aseltine, J., and Lehnert, W., (1995). Crystal: Inducing a conceptual dictionary, *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, pp. 1314–1319.

Soderland, S., (1997). Learning to extract text-based information from the world wide web, *Proceedings of Third International Conference on Knowledge Discovery and Data Mining (KDD-97)*.

Soderland, S., (1998). Learning information extraction rules for semi structured and free text. To appear in the *Journal of Machine Learning.*

Sowa J.F., (2000). Knowledge Representation: Logical, *Philosophical, and Computational Foundations,* Brooks Cole Publishing Co., Pacific Grove, CA, Chapter 7, pp. 452.

Stevenson M., (2006). Fact distribution in Information Extraction, *Language Resources and Evaluation,* Springer, pp. 183-201.

Stevenson M., (2007). Fact distribution in Information Extraction, *Language resources and evaluation,* Springer, Netherlands, pp.183-201.

Theodoridis S., Konstantinos K., (2003). *Pattern Recognition*, Amsterdam, The Netherlands: Academic Press.

Uramoto N., Matsuzawa H., Nagano T., Murakami A., Takeuchi H., Takedaent K., (2004). A text-mining system for knowledge discovery from biomedical documents, *IBM Systems Journal, Unstructured Information Management,* Volume: 43, No: 3.

Whiteley D., (2000). *e-commerce Srategy, Technologies and Applications,* McGraw-Hill (UK).

Yang J., Choi J., (2003). Agents for Intelligent Information Extraction By Using Domain Knowledge and Token-based Morphological Patterns, *Lecture Notes in Artificial Intelligence,* Vol. 2891, pp. 74-85.

Yang J., Oh H., Doh K., Choi J., (2002). A Knowledge-Based Information Extraction System for Semi Structured Labeled Documents, *Lecture Notes In Computer Science;* Vol. 2412, p.p.105 - 110 ISBN:3-540-44025-9 , Springer Verlag, London.

Zhou Y., Cho H., (2001). *Classification Algorithms on Text Documents,* http://www.cs.utexas.edu/users/hyukcho/classificationAlgorithm.html.

Zhou, G., & Zhang, M., (2006). Extracting relation information from text documents by exploring various types of knowledge, *Information Processing and Management,* Elsevier.

APPENDICES

Collecting Web Page Addresses into a Text File

```
function retrive_links(root,query,n)
fid=fopen([root 'links\' query '.txt'],'w+');
extension=[{'.pdf'} {'.doc'} {'.ppt'} {'.xls'} {'.jpg'} {'.jpeg'} {'.gif'} {'.pict'} {'.bmp'}
{'.tif'} {'.png'} {'.ico'}];
if n<=100
   N=fix(str2num(n)/10)
   num=10;
else
   N=fix(str2num(n)/100)
   num=100;
end
for p=0:N
url=['http://www.google.com/search?num=' num2str(num) '&q='];
q='+site:.com';
if p>0
   url=[url query q '&lr=lang_tr&start=' num2str(p*num) '&sa=N&filter=0'];
else
   url=[url query q '&lr=lang_tr'];
end
recentFile=urlread(url);
str='<a class=l href=';
hits = findstr(recentFile,str);
links=[];
new_link='';
for i=1:length(hits)
   new_link='';
   k=1;
```

```
    s='';
    while strcmp(s,'')==0
        s=recentFile(hits(i)+length(str)+k);
        new_link(k)=s;
        k=k+1;
    end
  fit=0;
  for k=1:length(extension)
    x=findstr(new_link,char(extension(k)));
    if length(x)>=1
        fit=1;
        break;
    else
        continue;
    end
  end
  if fit==0
        new_link=new_link(1:end-1)
        new_link=[num2str(p*num+i) ' ' new_link];
        fprintf(fid,'%s\n',new_link);
  end
end
if (num-length(hits))==0
    continue;
else
    break;
end
end % for p
fclose(fid);
```

Downloading Web Pages into Central Machine

```
retrieve_links(root,query,num);
fid=fopen([root 'links\' query '.txt'],'r+');
    while 1
        tline = fgetl(fid);
        if ~ischar(tline), break, end
        save_html(root,tline,query);
    end
fclose(fid);
t2=clock;
disp(t2-t1);
disp('All html files have been saved successfully...');

function save_html(root,url,keyword)
encoding='ISO-8859-9';
slCharacterEncoding(encoding);
path=[root 'products\' keyword '\'];
[filename url]=strtok(url);
[F,STATUS] = urlwrite(url,[path filename '.html']);
```

Extracting Keywords from Web Pages

```
function KeywordFinder1(root,query)
t1=clock;
D=dir([root 'products\' query '\*.html']);
path=[root 'products\' query '\'];
    for f=1:length(D)
        file=D(f).name;
        filepath=[path file];
       fid=fopen(filepath,'r');
```

```
    keywordfile=[root 'keywords\' query '\' file];
   tag='keywords';
    keywords=[];
    keywordlist=[];
while 1
      tline = fgetl(fid);
      tline=lower(tline);
      if ~ischar(tline), break, end
      %disp(tline)
      x1=findstr(tline,tag);  % Show the starting position of the tag in a line
      if size(x1)==0
         continue
      else
         x2=findstr(tline,'>');
         if size(x2)==0
            x3=findstr(tline,'content');
            if length(x3)>=1
               for i=x3(1)+9:size(tline)
                 keywords=[keywords tline(i)];
               end
            end
         else
           x3=findstr(tline,'content');
           if length(x3)>=1
             for i=x3(1)+9:x2(1)-2
                keywords=[keywords tline(i)];
             end
             index1=1;
             index2=1;
             for i=1:length(keywords)
                if keywords(i)==','
```

```
                    index2=i;
                    if keywords(index1)==' '
                       kline=keywords(index1+1:index2-1);
                    else
                       kline=keywords(index1:index2-1);
                    end
                    fid2=fopen(keywordfile,'a');
                    fprintf(fid2,'%s\n',kline);
                    fclose(fid2);
                    index1=index2+1;
                  end % if
                end %for
              end %if
            end %else
           break;
         end;
   end
   fclose(fid);
end
```

Keyword File Matrix Construction

```
function KeywordFileMatrix2(root,query)
D=dir([root 'keywords\' query '\*.html']);
keywordlist=[];
indexlist=[];
k=0;
for i=1:length(D)
   file=D(i).name;
   fid=fopen([root 'keywords\' query '\' file],'r');
   while 1
    tline = fgetl(fid);
```

```
      if ~ischar(tline), break, end
      k=k+1;
      keywordlist{k}=tline;
      indexlist(k)=i;
    end
    fclose(fid);
end
[X,I]=sort(keywordlist);
fid2=fopen([root 'reports\' query '\KeywordFileMatrix.txt'],'w+');
for i=1:length(X)
    str='';
    for t=1:length(D)
       str(t)='0';
    end
    for j=1:length(X)
        if strcmp(X(j),X(i))
            fileindex=indexlist(I(j));
            str(fileindex)='1';
        end
    end
    keyword=[];
    keyword=char(X(i));
    fprintf(fid2,'%s %s\n',keyword,str);
end

function CleanDuplicates3(root,query)
t1=clock;
fid=fopen([root 'reports\' query '\KeywordFileMatrix.txt']);
fid1 = fopen([root '\reports\' query '\KeywordFileMatrixClean.txt'],'w+');
once='xyzxyz';
 index=1;
kelimeler=[];
```

```
while 1    % birinci dosyadaki bütün kelimeler  kelimeler cell deðiþkenine atnýyor
     kelime= fgetl(fid);
   if ~ischar(kelime), break, end
   kelimeler{index,1}=kelime;
   index=index+1;
end

for i=1:length(kelimeler)-1    %kelimeler teker teker alýnýyor ve aþaðýdaki kelimelerle karþýlaþtýrýlýyor
  albirincikelime=kelimeler{i,1}; % ve aynýsý yoksa yazdýrýlýyor varsa yazdýrýlmýyor.
  yazdir=true;
  for j=i+1:length(kelimeler)
     alikincikelime=kelimeler{j,1};
     if strcmp(albirincikelime,alikincikelime) == 1
        yazdir=false;
        break;
     end
  end
  if yazdir == true
    fprintf(fid1,'%s\n',albirincikelime);
  end
end
```

Computing Keyword Frequency

```
function Frequency4(root,query)
t1=clock;
D=dir([root 'keywords\' query '\*.html']);
fid=fopen([root 'reports\' query '\KeywordFileMatrixClean.txt']); %
numwords=0;
```

```
keywords=[];
totals=[];
while 1
   tline=fgetl(fid);
   if ~ischar(tline), break, end
   numwords=numwords+1;
   str='';
   for i=1:length(tline)-length(D)-1
      str=[str tline(i)];
   end
   totalsite=0;
   for i=length(tline)-length(D)+1:length(tline)
     totalsite=totalsite+str2num(tline(i));
   end
  keywords=[keywords '_' str];
  totals(numwords)=totalsite;
end
fclose(fid);
[totalmax,Imax]=max(totals);
[totalmin,Imax]=min(totals);
avg=mean(totals);
index=[];
k=0;
for i=1:length(keywords)
   if keywords(i)=='_';
      k=k+1;
      index(k)=i;
   end
end
fid2=fopen([root 'reports\' query '\DeterminedKeywords.txt'],'w+');
for i=1:numwords-1
   if totals(i)>threshold
```

```
        word=[];
        for j=index(i)+1:index(i+1)-1
            word=[word keywords(j)];
        end
        fprintf(fid2,'%s %d\n',word,totals(i));
    end
end
fclose(fid2);
```

Detagging Web Pages

```
function Detag(root,query)
 t1=clock;
encoding='ISO-8859-9';
slCharacterEncoding(encoding);
root='C:\ferkan\'; query='nokia 1100';
 D = dir([root 'products\' query '\*.html']);
 for f=1:length(D)
url = ['file:///' root 'products\' query '\' D(f).name];
recentFile=urlread(url);
 recentFile= lower(recentFile);
```

Removing Stopwords

```
clear all;
clc
root='C:\ferkan\corpora\';
keyword='nokia 1100';
file=[root keyword '.txt']
alfabe='abcçdefgðhýijklmnoöpqrsþtuüvwxyz';
fid2=fopen(['C:\ferkan\corpora\' keyword '\CleanCorpora.txt'],'w')
```

```
load stopword_list;
url = ['file:///' root 'nokia 1100.txt']
recentFile=urlread(url);
 [token, rem] = strtok(recentFile);
while strcmp(rem,'')==0
    index=0;
   for i=1:length(alfabe)
       if token(1)==alfabe(i)
           index=i;
           break;
       end
   end
   if index==0
       fprintf(fid2,'%s ',token);
   else
       index2=0;
       for j=1:size(stopword_list,2)
          if isempty(stopword_list{index,j})==1
              break
          end
          if strcmp(stopword_list{index,j},token)==1
              index2=j;
              break
          end
         end
       if index2==0
        fprintf(fid2,'%s ',token);
       end
   end
   [token,rem]=strtok(rem);
end
fclose(fid2);
```

Converting Text into Lower Case

```
function textduzelt(path,file)
encoding='ISO-8859-1';
slCharacterEncoding(encoding)
file=[path '\' file]
file2=file(1:length(file)-4);
file2=[file2 '_2.txt'];
fid1 = fopen(file2,'wt+','native');
recentFile=urlread(['file:///' file]);
recentFile=lower(recentFile);
fprintf(fid1,'%s',recentFile);
fclose(fid1);
```

Creating input file

```
function CreateCorpus(root,query)
t1=clock;
root='C:\ferkan\';
query='nokia 1100';
D=dir([root 'detagged\' query '\*.txt']);
path=[root 'detagged\' query];
for i=1:length(D)
  CreateDocumentFile(root,query,D(i).name);
end
function CreateDocumentFile(root,query,file)
encoding='ISO-8859-1';
slCharacterEncoding(encoding);
path=[root 'detagged\' query ];
path2=[root 'corpus\' query];
```

```
f1=fopen([path '\' file],'r');
file2=file;
file2=[path2 '\' file2];
file2=file2(1:end-4);
file2=[file2 '_2.txt'];
f2 = fopen(file2,'w+');
while 1
  tline = fgetl(f1);
  if ~ischar(tline), break, end
  for i=1:length(tline)
     if isstrprop(tline(i),'punct')
        tline(i)=' ';
     end
  end
  [token, rem] = strtok(tline);
  fprintf(f2,'%s ',token);
  while strcmp(rem,'')==0
     [token,rem]=strtok(rem);
     fprintf(f2,'%s ',token);
  end
end
fclose(f1);fclose(f2);
f2=fopen(file2,'r');
f3=fopen([root 'corpus\' query '\' file],'w+');
while 1
  tline = fgetl(f2);
  if ~ischar(tline), break, end
  [token, rem] = strtok(tline);
    fprintf(f3,'%s ',token);
  while strcmp(rem,'')==0
     [token,rem]=strtok(rem);
     fprintf(f3,'%s ',token);
```

```
  end
end
fclose(f2);fclose(f3);
delete([path2 '\*_2.txt']);
```

Extracting Features and Values

```
root='C:\ferkan\reports\';
keyword=' ';
path=[root keyword '\'];
for i=1:length(monogram)
clear kelime
 for k=1:length(monogram{i,1})-1
  kelime(k)=monogram{i,1}(k);
 end
 for j=1:length(digram)
      if strcmp(digram{j,1},kelime)==1
         index=j;
         break
      end
 end
 for j=index:length(digram)
      if strcmp(digram(j,1),kelime)==1
         digram{j,2}
         digram{j,3}
         break
      else
         break
      end
 end
  break
end.
```