Technical University of Crete

Department of Electronic and Computer Engineering

Dissertation Thesis

Relevance Feedback Methods for Web Image Retrieval

KONTIS KLAYDIOS

Guidance Committee:

Euripides Petrakis	Associate Professor (Supervisor)
Kondogiannis Kostandinos	Associate Professor
Manolis Koubarakis	Associate Professor

Chania, June 2004

Abstract

Image retrieval is an open problem which hasn't been analyzed enough in the existing literature. In this thesis, we implement a prototype system for Web based image retrieval. The system is based on description of images by lexical chains extracted from text related to images in a Web page. Special attention is given to relevance feedback techniques that aim at the adjustment of our system to real user preferences. Some of the most important relevance feedback techniques and approaches are implemented in this dissertation thesis. The above relevance feedback techniques based on image text description are expanded to support retrieval by combining textual and visual features (extracted by applying image content analysis techniques).

All methods are implemented and compared with precision and recall criteria. The experimental results prove that retrieval methods that make use of both text and visual features achieve overall better results than methods based only on image's text description.

Table of Content

Introduction....................11
Goals of Thesis....................12
CHAPTER I - Image Retrieval ideas and Previous Work....................13
1.1 Web-based image retrieval....................13
1.1.1 Text-based retrieval....................15
1.1.2 Content-based image retrieval (CBIR)....................15
1.1.3 Annotated image collections....................16
1.1.4 Discussion....................16
1.1.5 Relevance feedback....................17
1.2 Image Retrieval systems....................18
1.2.1 GOOGLE Image Search....................18
1.2.2 Ditto....................19
1.2.3 Picsearch....................20
1.3 Prototype IR systems on Web....................21
1.3.1 Diogenes Search Agent....................22
1.3.2 PicASHOW (Search by Hyperlinks)....................23
1.3.3 WebSeer....................24
1.3.4 PicturePiper....................25
1.4 Analysis....................26
CHAPTER II - Methodology....................27
2.1 Image Search – Definition....................27
2.2 Image Representation – Criteria....................28
2.3 Text based image representation....................30
2.4 Selected image representation....................33
2.5 Exactness – Noise Removal....................36
2.6 Attribute Importance....................37
2.7 Similarity Computation....................38
2.7 Similarity between image and query....................40
CHAPTER III - Relevance Feedback
3.1 Semantic Accumulation....................43
3.2 Semantic Integration Differentiation....................44
3.3 MARS Relevance Feedback (Rui – Huang)....................46
3.4 FALCON – Related Relevance Feedback....................48
3.5 Pseudo-Relevance Feedback....................49
3.5.1 K-Means Clustering....................50
CHAPTER IV - System Description and Implementation....................51
4.1 Crawling....................51
4.3 Extraction – Indexing....................54
4.4 Database....................55
4.5 Retrieval....................56
4.5.1 Retrieval – Chainnet....................59

4.5.2 Retrieval – Modified Chainnet 60
4.6 Relevance Feedback 60
4.6.1 Semantic Accumulation 61
4.6.2 Semantic accumulation and integration 62
4.6.3 Falcon 63
4.6.4 Rui – Text 64
4.6.5 K-Means - Pseudofeedback 65
4.6.6 Rui – Text and Visual 66
CHAPTER V - Experiments and Evaluation 68
6.1 Evaluation method-approach 68
6.2 Experiments 70
6.2.1 Queries 70
6.2.2 Weight comparison 72
6.2.3 Semantic Relevance Feedback 73
6.2.4 Other relevance feedback methods 74
6.2.5 Relevance feedback with visual features 75
6.3 Ranking quality 77
6.4 Conclusions 78

Introduction

Image Retrieval is becoming a domain of increasing and crucial importance in the new information based society, as a part of Information Retrieval (IR) field. IR is described as accurate and speedy access to a vast amount of information which can be in the form of text documents, image collections, video or other multimedia objects. IR has been a very productive field for many researchers in the past years, most brilliants being Cleverdon [1], Sparck Jones [2], [3], Lancaster [4], Salton [5], [6], [7], [8] and others. Information Retrieval does not inform (i.e. change the knowledge of) the user on the subject of his inquiry. It merely informs on the existence (or non-existence) and whereabouts of documents relating to his request, usually sorting them in some relevance order. Even though the principles of all IR systems are based on the same theory there are certain characterists that apply only to specific data structures or objects. Text retrieval with Boolean or vector space systems (the most representative being the SMART systems [9], [10]) gave birth to the need of indexing, updating, searching and managing large text collections properly. Nowdays, image, video and other multimedia besides text retrieval systems, each requiring sophisticated and data-dependent representations and techniques, are being discussed and implemented. Among the various data types, images are of prime importance. Not only is the most widely used media type besides text, but it is also one of the most widely used bases for representing and retrieving video and other multimedia information. New, more efficient methods (in both time and process speed), simple (from the user point of view), incorporating user preferences and feedback is the main concern in the wide retrieval research area.

Image retrieval has been addressed in various ways [11], [12], [13], [14], [15], since with the increase of Internet bandwidth and CPU speed the use of images in the World Wide Web has become prevalent. Images are used to enhance the document, capture the attention of users and reduce the textual content of a page. Large and distributed collections of scientific, artistic, technical and commercial images are becoming a common ground, thus requiring sophisticated and precise methods for users to perform similarity and semantic based queries. The WWW is host to millions of images on every conceivable topic thus making images an indispensable component of web pages. Finding effective methods and more efficient representations to retrieve and rank images from this rich source has been the center of many scientific efforts.

Goals of Thesis

As described above the image retrieval is a vast but partially explored field. A fairly new idea (lexical chain approach [16]) is studied, analyzed and implemented. Also, the reasons that lead us to select and make use of the specific text representation for the images and the difficulties encountered during implementation are noted in details. Image collection gathering process (i.e. web crawling), image database scheme, indexing techniques and retrieval issues are also discussed. A better more user centered approach than the original weighted lexical chains model is proposed, analyzed and compared. Several up-to-date relevance feedback techniques are described in detail and fully implemented and their results are compared.

Furthermore, the system is expanded to combine textual with visual features focusing at a particular and rather narrow yet very interesting from the industry's point of view, image retrieval problem: logo / trademark retrieval. The difficulties that arise, weight assignment or image feature selection are described and analyzed since they may have an impact in the overall performance. Nevertheless, the combination of textual and visual features has been shown to achieve higher precision then text or image features alone. Several other approaches are also tested and discussed.

CHAPTER I

IMAGE RETRIEVAL IDEAS – PREVIOUS WORK

1.1 Web-based image retrieval

Web-based image retrieval is the process of obtaining relevant images and multimedia content from the Word Wide Web. Applications of Web-based image retrieval among other are:

- Navigation of image collections.
- Publishing and advertising.
- Medicine and health related application domain.
- Architectural and engineering design.
- Crime prevention and legal issues.

Images on the WWW are described by text coexisting with images in the HTML documents. However, an effective system should integrate text and image based retrieval techniques. Emphasis is on automatic image indexing and content-based retrieval, although it is not clear how the retrieval functionality found in these systems correlates with image information needs of real users. In addition, the use of image retrieval systems varies in different fields since users have their own specific information-seeking behavior, and need certain features designed for their tasks. Nevertheless, much work has already been done in the image retrieval field, but the majority of the systems developed is either too specific or too general.

Research in the image retrieval problem, web-based retrieval being a part of it, has led to a plethora of image retrieval system such as general web search engines with image retrieval capabilities (Google Image Search [17], Altavista MultiMedia search [18] etc.), academic image systems (PicAShow [19] , WebSEEK [20] etc.) and commercial search engines dedicated to image retrieval (Scour [21], Ditto [22]).

Methods or systems referred above differ in search strategies and content representations however; they all (more or less) adopt the following architecture:

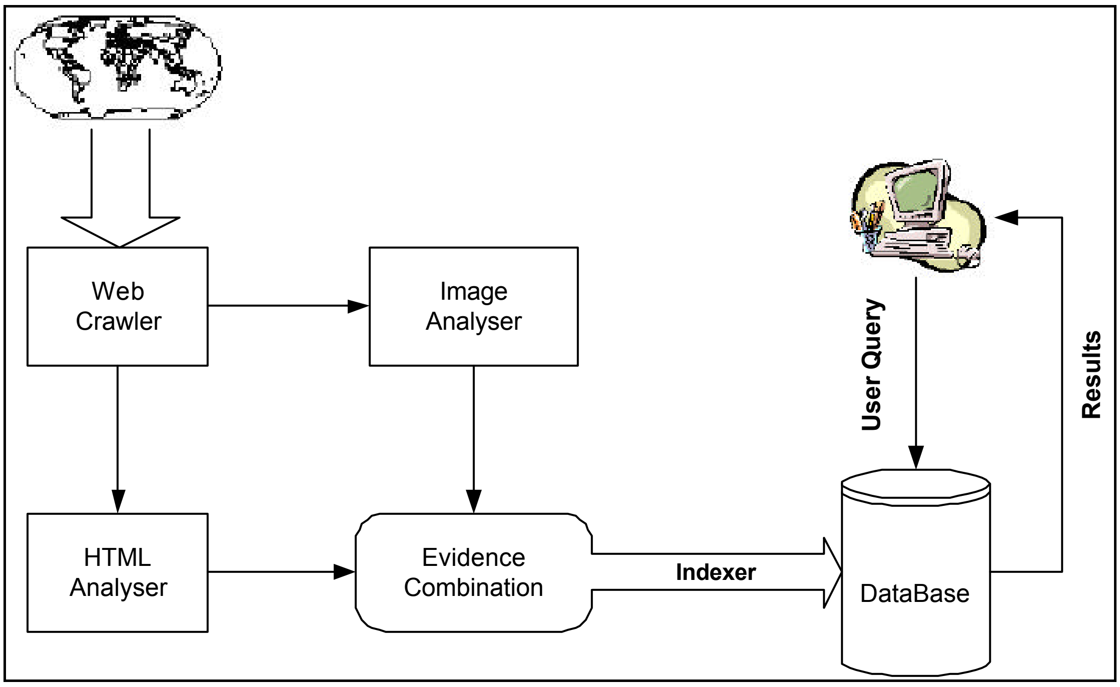


Figure1: Architecture of Web image retrieval system

A web crawler is used to collect images and their respective HTML documents from the World Wide Web and pass them to the text and image analyzer modules. The extracted data from the above modules go through an indexing process and are usually stored in database. The user submits its queries to the database which in turn replies by returning a set of results. The way the document's are represented, the features that are used to calculate the similarity between the user query and the web pages stored in the database and the representation of results depend entirely on the implemented system.

More specifically speaking, there are three main approaches to WWW image search and retrieval:

- *Text-based retrieval.*
- *Content-based image retrieval (CBIR).*
- *Annotated image collections.*

1.1.1 Text-based retrieval

This approach annotates images with text derived from the HTML documents that contain (display) them. This is based on the observation that an image in a Web page is semantically related to its surrounding text [16]. These can include the image caption, the image filename, the neighbor text round the image and or several other attributes. The extracted text is then almost always indexed and stored, represented in a specific schema that depends on the implementation of the system. The main idea behind text-based retrieval of images is that words or terms appearing at different locations of an HTML document have different levels of importance (relevance) to the images. Relevance feedback is often offered as the last part of such retrieval systems for the purpose of improving query results.

1.1.2 Content-based image retrieval (CBIR)

Image analysis techniques are also to extract a variety of visual features from images. These include histograms, color, texture measurements, image dimensions, shape, orientation, moment invariant features etc. The extracted features are usually indexed and stored in system's database. The user interacts with the CBIR system via a visual interface by issuing keyword queries, queries by image examples or queries combining keywords and image examples. The objective is to find and retrieve images from the database that satisfy the user's criteria of similarity with the query.

Usually the user can iteratively refine the search through the selection of more relevant images from the returned set, thus improving the precision with each itineration (relevance feedback).

1.1.3 Annotated image collections

There are several companies (e.g. Getty Images [23] and Corbis [24]), that specialize in providing visual content to a diverse range of image consumers. The images are indexed and retrieved by keywords and or query by example, which are manually assigned to each image or derived from proprietary techniques and algorithms. The image collections get updated periodically. While end users may use these services, they are especially geared toward companies and professionals who require high volumes of diverse images.

1.1.4 Discussion

The major fallback of the methods described above is their inability to make full use of both text and image data by integrating text-based and content-based retrieval techniques. Assuming that the most relevant pieces of information are extracted from the HTML document there is still the problem of irrelevant or subjective descriptions (plainly noise) that can lead to very poor retrieval performance. Also, the semantic meaning of the query must be considered as a key issue. On the other hand, content based retrieval schemes tend to categorize image in topics of interest trying to minimize text description to only one word. Content-based image retrieval depends on very well defined image features that may give good results considering the fact that the user is familiar with such low-end semantic information. In the real word this assumption is rarely true. Also, the human perception subjectivity i.e. different persons under different circumstances may perceive the same visual content differently stand way further from the computer centric approaches stated above.

Lately, the academic research is focused on combining text-based and CBIR image retrieval by selecting and using the most representative features of both approaches.Weight are assigned to each feature according to their importance, in order to achieve high result precision. The best example of this approach is the *Diogenes* [25], [26] system but still these efforts are far from being perfect since while HTML features derives from hyperlink structure might, loosely speaking, be considered as static data, visual features vary in regard to the category (taxonomy) to which the image belongs (cartoon, face, logo, landscape etc.). Also, efforts are made to go beyond the limited computer-centric approach in order to capture user preferences and information needs by offering various relevance feedback techniques. Other approaches of Image Retrieval on the Web include ImageRover [27], Weebseer [28] and other.

1.1.5 Relevance feedback

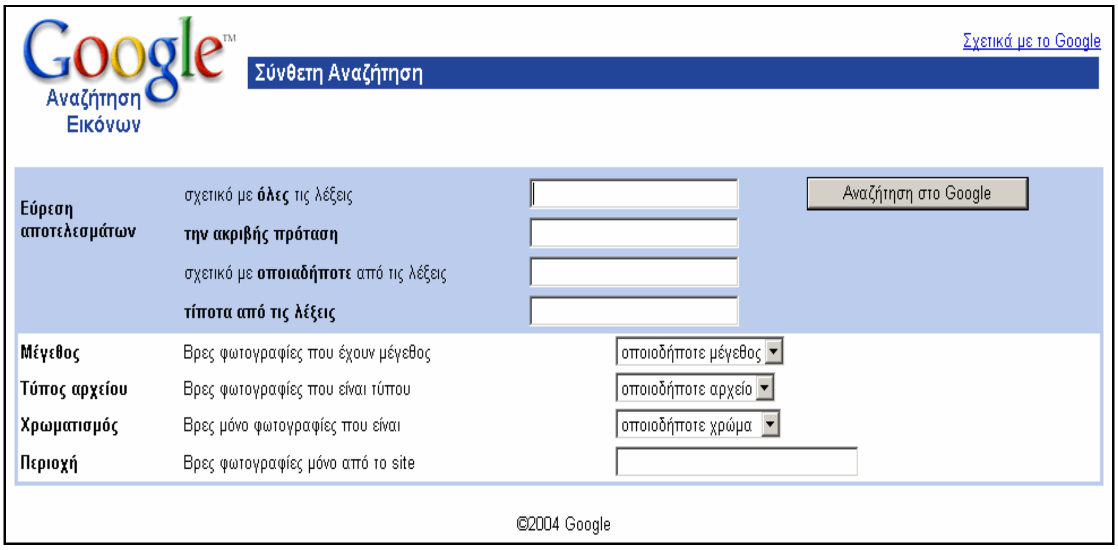
Relevance feedback is a powerful technique used in traditional text based information retrieval systems. The idea is to adapt the system to the specific user preferences making more important weights or features that reflect the actual user needs in order to achieve higher precis ion. Therefore we can define relevance feedback as the process by which human and computer interact in order to automatically adjust an existing query to the real user preferences. The idea is to adjust the query selection criteria to better approximate real user needs using the results retrieved by previous (or the original query). This is achieved though a series of query reformations and retrievals (called feedback) until the query results converge to what the user is actually looking for. Several issues must be dealt with here in order to achieve good result; the biggest one being the fixed, a priori determination of the attribute's weights and the necessity not to overburden the user by requesting direct weight specification before the adjustment procedure. Therefore, it is commonly suggested that the user should be asked only to mark the results (or a part of them) as relevant or irrelevant to the query while the weights are dynamically updated. Besides weight adjustment (e.g. Rui in the Mars system [29]), other relevance feedback techniques exists such as query point movement (MindReader [45]), calculation of a new (dis)similarity function (e.g. Falcon [31]) etc. A subcategory of the above, are the pseudorelevant techniques in which the system redefines the results without any user information applying weight adjustment using mean value and deviation of all features involved in the retrieval process.

1.2 Image Retrieval systems

In the following paragraphs we present some of the most representative image retrieval systems. Most of the online Image Retrieval systems lack documentation due to proprietary algorithms and techniques while the other category provides full documentation but no real time demonstration. Therefore we try to describe the most representative ones trying to note their architecture and techniques, the way they are implemented, their advantages and fallbacks.

1.2.1 GOOGLE Image Search

Probably the most known and used general-purpose image search engine It applies almost the same search techniques that are used by other search engines (Altavista, Lycos MMedia etc.) to search for multimedia content but it’s success is most due to the link analysis algorithm implemented within Google. It takes advantage of the very large, highly efficient backend database that Google search engine uses (crawling process occurs often) and of the proprietary PageRank [32], [33] ranking algorithm in order to achieve very fast and fairly accurate image retrieval. It relies heavily on indexed text-based searching techniques, appointing high priority weights to the surrounding image text, especially the image name and the caption of the image. It also uses the link analysis technique described in the PageRank algorithm to determinate the degree of relevance between the images, their WebPages and the user’s query image.


Google
Αναζήτηση Εικόνων
Σχετικά με το Google
Σύνθετη Αναζήτηση

Εύρεση αποτελεσμάτων	σχετικό με **όλες** τις λέξεις	
	την ακριβής πρόταση	
	σχετικό με **οποιαδήποτε** από τις λέξεις	
	τίποτα από τις λέξεις	
Μέγεθος	Βρες φωτογραφίες που έχουν μέγεθος	οποιοδήποτε μέγεθος
Τύπος αρχείου	Βρες φωτογραφίες που είναι τύπου	οποιοδήποτε αρχείο
Χρωματισμός	Βρες μόνο φωτογραφίες που είναι	οποιοδήποτε χρώμα
Περιοχή	Βρες φωτογραφίες μόνο από το site	

Αναζήτηση στο Google

©2004 Google

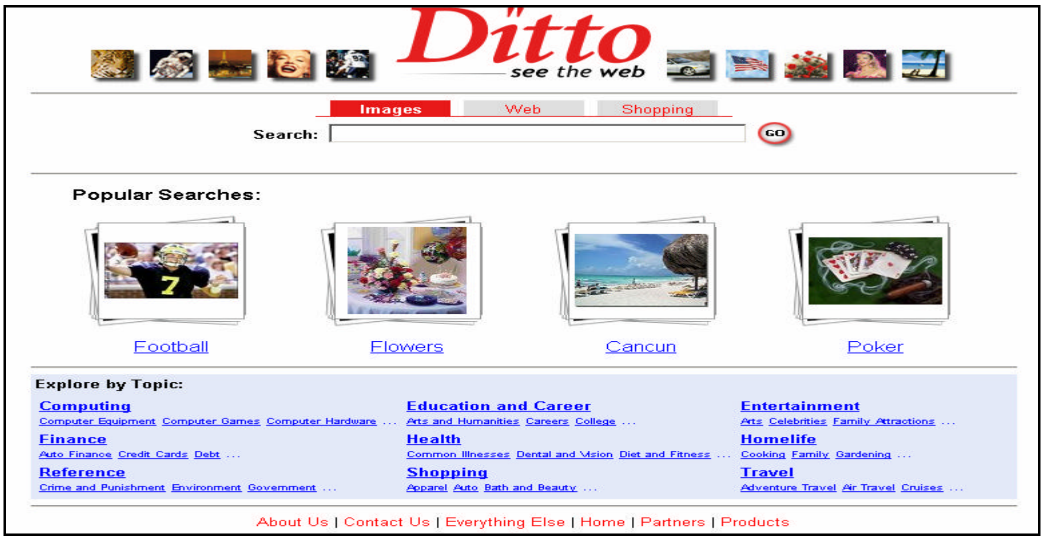
Figure 2: Google image search interface

Google provides a simple, nevertheless highly functional, search interface as seen on the figure 2. Also, images can be viewed even if the page they were embedded in is off the Web.

Even though Google Image search is one of the most used and somehow complete image retrieval system, the major disadvantage of this approach is the fact that the search depends heavily on textual analysis (word occurrence for example), such as returning low precision result for general-term queries. Also, despite the color checking routine (the user can specify if the images to be search are black&white, grayscale or colored), it lacks other visual content features making the system very sensitive to noise (non-relevant text) i.e. irrelevant results even for strictly defined queries. Google image search is available online at *http://images.google.com*

1.2.2 Ditto

The goal of Ditto image search is to deliver relevant thumbnail images and the relevant web sites underlying these images. Ditto claims it employs a text plus visual search method, with a 10 scale ranking scheme. Nevertheless, Ditto's technology can be described as a three step process. *Indexing:* identify websites containing media via automated crawling, selecting, ranking, weighting, filtering, rating and indexing pictures, illustrations, clipart, photographs, drawings and other image-related material. *Relevance:* a proprietary filtering process that combines sophisticated automated filtering with human editors. *Verification:* ongoing maintenance by continually reviewing current images and links to ensure that database contains accurate and up-to-date results. The relevance process is not to be confused with relevance feedback techniques or implementation.



Ditto
see the web
Images
Web
Shopping
Search:
GO
Popular Searches:
Football
Flowers
Cancun
Poker
Explore by Topic:
Computing
Computer Equipment Computer Games Computer Hardware ...
Education and Career
Arts and Humanities Careers College ...
Entertainment
Arts Celebrities Family Attractions ...
Finance
Auto Finance Credit Cards Debt ...
Health
Common Illnesses Dental and Vision Diet and Fitness ...
Homelife
Cooking Family Gardening ...
Reference
Crime and Punishment Environment Government ...
Shopping
Apparel Auto Bath and Beauty ...
Travel
Adventure Travel Air Travel Cruises ...
About Us | Contact Us | Everything Else | Home | Partners | Products

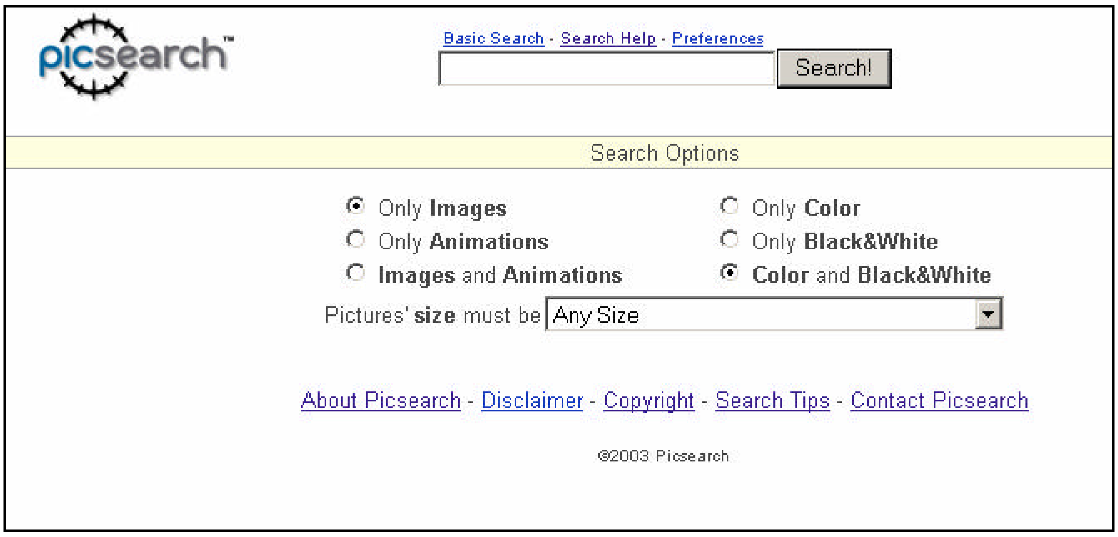
Figure 3: Ditto image search

Also, along with the images, it presents some Web sites that are relevant to the users's query. Ditto provides image search based on text description from the user. The returned results have low precision especially for one-term queries.

Ditto image search is available online at *http://www.ditto.com*.

1.2.3 Picsearch

Picsearch provides a specialized search engine for pictures only. It offers both text description and by example queries. Instead of using a traditional text search engine, Picsearch uses a crawler agent to collect and index images in their local database. Picsearch claims it has a *relevancy* unrivalled on the web due to it's patent-pending indexing algorithms. Its indexed pictures pass through advanced filters to eliminate offensive material. Also, Picsearch provides a very *user friendly* interface, designed to be simple, fast and accurate. An advanced search option is offered too: users can specify the kind of pictures (for example animations or images), the colour (black and white or colored), and the desired size (measured in pixel) of the pictures they are looking for.


picsearch™
Basic Search - Search Help - Preferences
Search!
Search Options
About Picsearch - Disclaimer - Copyright - Search Tips - Contact Picsearch
©2003 Picsearch

[✓ Only Images] Only Animations Images and Animations

Only Color Only Black&White [✓ Color and Black&White]

Pictures' size must be Any Size

Figure 4: PicSearch advanced image search

Since Picsearch algorithms are proprietary no detailed description is available. The results returned from queries by example have a very satisfying precision while those from text queries tend to have rather low accuracy.

Picsearch is available online at *http://www.picsearch.com* .

Other online image retrieval systems exist as well, some of those being **Altavista MultiMedia** [18] and **Scour** [21], nevertheless there is no key difference with the online systems described above since the index and retrieval philosophy are quite similar.

1.3 Prototype IR systems on Web

The second category are the retrieval systems that haven't gone commercial yet, developed by independent research groups in academic or research labs.Thy are fully documented and explained in details yet they are not available online for further tests or acquaintance. Therefore among several of them, we try to present the most representatives of each approach.

1.3.1 Diogenes Search Agent

Developed by Dr. Aslandogan at Uni. Texas Austin [25], [26], Diogenes is a web-based, automated image search agent designed specifically for retrieving person images. Diogenes works with the web pages that contain a facial image accompanied by a body of text. It uses a crawler to collect images offline (search engines, Altavista specifically, are used to retrieve the initial target set of pages) and build an indexed database. Diogenes uses both textual and visual features obtained from the face detection and face recognition module. If a face is found, image is submitted to the text extraction and face recognition modules. The face recognition is based on the eigen-face method that uses a set of known facial images for training, each associated with a person name. The text-HTML analysis module computes the frequency of occurrence and the location of the image within the page in order to determine the degree of association between a person name and an image. The most interesting factors include whether the person's name occurs in proximity of the image, whether they are enclosed in the same tag, whether the person's name is part of the image URL path.

The combination of the outputs of the above two modules is based on the Dempster-Shafer evidence combination theory i.e. a generalization of the Bayesian probability theory to account for uncertainty. The uncertainty degree of the face recognition and of text/html modules obtained locally are applied to the Dempster-Shafer rule for combination giving a rank score for each image. Since the outputs are generated automatically, Diogenes is able to determine the correct weights for emphasizing one feature or another without input from the user. In addition, the use of the face detection techniques that eliminate non-person images give more accurate results compared to other wide range image search systems. Also, the Dempster-Shafer theory provides fairly high precision results in the early stages of a query.

The big fallback is the limited use of Diogenes, since it can only be used for retrieving person images making it a system not suitable for wide range image retrieval queries. On-demand searching is offered too, but it is presumed to be rather slow since all the module output must be computed online, thus delaying the entire process.

1.3.2 PicASHOW (Search by Hyperlinks)

PicaShow is a fully automated WWW image retrieval system developed by Lemplel and Soffer at Haifa Technion University [19]. It is based on Kleinenberg's link analysis algorithms and Kleinenberg's co-citation [34]. PicASHOW further contend that the standard reasoning behind the co-citation measure applies to images just as it does to HTML pages: images which are co-contained in pages are likely to be related to the same topic and images which are contained in pages that are co-cited by a certain page are likely related to the same topic. Therefore a link from page *p* to page *q* can be viewed as an endorsement of *q* by *p,* and as some form of positive judgment of *q*'s content by *p*. The image collections is gathered by submitting queries to traditional text-based search engines then adding pages that are pointed by or point to the resultant set. PicASHOW represents each set as a quadruple *IC=(P, I, L, E)* where P is the set of WWW pages, I is the set of images contained in P,L the set of links that exists between the pages of P and E is the relation page y contains image x. Image ranking derives from calculating and comparing the adjacency matrixes of page-to-page relation *L* and of the page-to-image relation *E* based on co-citation influented methods. Identification of replicated images and noise filtering techniques (noise being loosely defined as banners, logos, sponsor links, inner links etc.) are applied in order to improve the precision and speed up the entire process.

The advantage of PicASHOW is that high precision is achieved even when image filenames are meaningless compared to the query. Also the fact that it requires no image or text analysis whatsoever, nor creation of taxonomies for preclassification of the images makes PicASHOW quite easy to incorporate in existing search engines with reasonable overhead in terms of both computation and storage with no change to user query format. It performs very well on wide-topic queries but on the other hand PicASHOW performs rather poorly on specific queries returning results that may be relevant to the wide topic not the specific aspect of it. By not performing any image or text analysis, PicASHOW fails to handle queries involving image qualifiers such as color, size etc. or to capture the semantic meaning of a query by applying textual analysis of the page the image is embedded in.

1.3.3 WebSeer

WebSeer [28] was an image retrieval system developed at the university of Chicago, Illinois (currently not available online) that made use of both visual and textual features. Like text search engines, all analysis of the image and surrounding text was done off-line during the creation of the database.

Multithreaded crawlers were used in order to download the HTML documents and the embedded images. Also, multithreaded text and image processing modules were used to extract and index the desiderable characteristics. The extracted textual features were the document's title, the image name, the alternate text and the image caption with different importance weights assigned to each one of them (for example title given lower weight then alternate text). When a user performs a search the summed weights of the matching words are used as one criterion for sorting the resulting images. Also, WebSeer uses information derived from analyzing the image content to complement the textual information associated with an image and information derived from the image header. This additional information was used to create a context in which image analysis algorithms can effectively operate. Image analysis algorithms were then used to classify the image within a taxonomy of types (photograph, portrait, computer-generated graphic etc.) and to extract useful semantic information such as the scale of a portrait (close-up, half-body shot, full-body shot, etc.). A face recognition module which searches for upright faces oriented towards the camera was used in parts of the image in order to determine if and how many faces were represented in the image. In addition, duplicate images were detected using an image checksum and other information. WeebSeer uses multidecision trees in order to classify the image into certain categories.

The searching process can be seen as a sequence of Boolean conditions on the user-defined desiderable visual characteristic that are applied to the results obtain from the sum of weights of the textual features. As far as we know no feedback techniques were implemented at all. In addition, as described in the respective paper, the user interface was sort of difficult for the middle user to handle, compared to other similar systems like the ones described above.

1.3.4 PicturePiper

PicturePiper [35] is a image search tool based on a proposed novel architecture that implements on-the-fly service addressing and re-configuration in order to support interactive applications. As such, PicturePiper provides a mechanism that allows user access to images on the WWW related to a topic of interest. It consist of these modules: Web search (query submission to search engines), Image Finder & Downloader (image and neighbor text are extracted and locally stored), Feature Extractor (computes vectors for the clustering features : color histogram, complexity and associated text) and Centroid creator and Multi dimensional scaling responsible for the clustering process.

The user interactively browses a stream of incoming images by repeatedly clustering them and selecting the clusters that look most interesting. Candidate images are collected by submitting the user query to a web search engine and by filtering the returned results using the Multi Modal Scather-Gather technique. In this technique, the system first separates the collection of images into a fixes number of clusters, each of which contain images that are similar in some way i.e. similar color feature, image complexity : cartoon to photographs and textual features as vector of word count in the surrounding text. User feedback is needed so that relevant images selected from the user contribute in the cluster's redefinition process. Also, there are no restrictions as to how many times the user can select his/her results. In addition, new images are added to the already downloaded ones, even after the user has begun the Scather-Gather process to narrow the results. These incoming new images are compared to the cluster centroids for each of the previous iterations. Also, the user can locally store images that he likes/ considers relevant and user interface is fairly simple to use.

PicturePiper clustering may prove superior for situations in which the user wants to learn what is available and then pick up the images that best fir his needs. But PicturePiper will give poor results in queries that aim at finding a specific image that the user has in mind.

1.4 Analysis

As seen, there are several approaches to image retrieval. In briefing, Google and Ditto use only text description (keywords) to image retrieval. Picsearch uses text or predefined image retrieval (by selecting from a set of examples) while all the above systems do not provide any relevance feedback at all. Diogenes makes good use of text and image visual features however, its application domain is rather narrow since it is implemented only for person-face retrieval. Picashow applies image retrieval by hyperlink analysis while WeebSeek relies on visual feature and relevance feedback from the user. The PicturePiper approach relies on visual features and repeated relevance feedback from the user in order to achieve good results.

CHAPTER II

METHODOLOGY

Traditional text retrieval with Boolean or vector space models when applied to the image retrieval problem provide rather low precision and fail to capture any semantic meaning around the image. Moreover, these methods fail to take advantage of the several hyperstructure attributes that refer (especially) to images. Weight assignment to various HTML text structures that describe the images is a better approach since it deals properly with the importance issues of the mentioned structures but also fails to deal with semantic capture and calculation.

This chapter describes a different approach that identifies and captures the image semantics within a HTML document by introducing a new image and query representation that use certain image related hyperlink structures. We will try to present some paradigms tha t we deem will make things simpler in theoretical basis.

2.1 Image Search – Definition

First let's try to explain how we comprehend the phrase "image search". By image search we mean a process or procedure, being part of a whole system that takes user queries and replies with specific images that are semantically and visually similar to the requested query. Usually these images are retrieved either from the World Wide Web (online or on-demand query) either from an internal database (offline query) derived from crawling the WWW. The format of user queries may be:

- **Text**: few words describing the desired images.
- **Image**: an image representing the category of interest.
- **Text and Image**: some describing words and a similar image.

Each of the above represents a stand-alone implementation of both text and visual features that depend heavily on the format the query has. For example in an image query some visual features are calculated and images with similar values are returned. Therefore the key issue in designing and implementing an image retrieval system a representation for a WWW image, the capture and representation of the query semantics and a similarity measure/function between an image and a query definition based in the above representation.

2.2 Image Representation – Criteria

Independently from the query format, some desired conditions have to be fulfilled in order for the defined representations to achieve quick, high and approvable results. We could say that the same conditions applying to the well-know text based search methods are acceptable in image search too. Nevertheless, considering the peculiarity, abstract and multiple use of images in nowdays Web pages we have identified several desiderable properties of query/image representation such as:

- **Automatic extraction**
- **Exactness**
- **Space efficiency**
- **Computationally inexpensive**
- **Insensitive to noise**
- **User friendly**

Automatic extraction

The best approach is that the representation must be extracted automatically from the HTML documents or the available images. The human factor must be removed during the extraction process since it has become obsolete (quite cheap, high speed computer devices) and has high cost (both in labor and maintenance terms). Also, perception varies from person to person i.e. different persons give different annotations for the same image thus making the system very fragile and reducing precision.

Exactness

The representation, beyond the simple approach, must be able to capture the essential query or image semantic meanings. Other techniques, rather then the usual text or content based mechanisms, must be deployed in order to achieve high precision, possibly in the early stages of the returned results.

Space efficiency

The above representations must be space efficient, meaning that they must not consume too much storage. Besides the cost of large storage devices (even though storage cost is decreasing rapidly nowdays), repeated indexing and update maintenance, the goal is to make the internal database reduce much of the necessary I/O, thus reducing the cost and improving the output's speed.

Computationally inexpensive

The chosen representation of the image set and of the user query must be fast to extract, store, interpret and process in computer terms. Also, it must be fast in calc ulating the similarity between the representations, thus imposing the need for simple, yet effective computer structures and languages.

Insensitive to noise

The selected representations must not be drastically affected by any sort of noise that is entered during the process of locating and extracting the desired features. Checking must be done in order to avoid same images and to strip off the text representation of common words, irrelevant HTML clauses etc that might have a negative impact on overall output precision.

User friendly

The system (especially the user's query representation) must be fairly simple to operate by any user, not overburdening with criteria that can be avoided, thus shielding from the details of the implementation and approaching the already widely used and best known text search engine systems.

2.3 Text based image representation

Given the desired properties of an image retrieval system, the next issue is the presentation of our features and of the reasons that led us in selecting them. As foretold in the previous chapter our observation is based on the fact that an image in a WWW page is *semantically* related to its surrounding text (the so called *functional* images such as next, previous, home, under construction etc. being an exception). The surrounding text is usually used to provide description and illustrate the particular semantic of the image content, i.e. what is the image about, the objects in the image, the date and place and what is happening in the image. The plethora of HTML components in an image tag seems to be a problem since a lot of information, not always relevant, sometimes unnecessary is used. Nevertheless, we can aspect that some of these components include and represent the most significant semantic information about the image. The most representative being:

- **Image filename**
- **HTML document's title**
- **Alternate description of the image.**
- **Image caption**

Many questions may arise regarding the selection of the specific attributes, but after having carried out several experiments the relation of an image and its corresponding HTML document, based on our conclusions; we think that the four descriptions above are the most semantically related to the embedded image. Other parts might have been chosen as well with **metadata** or the **whole text** being the most important candidates. The **metadata** do indeed provide some information about the whole document consequently about the embedded images too, yet the information is most of the times too much and too unrelated. We have also excluded the capture and processing of the **whole HTML document** since the goal of extracting the semantic meaning of the surrounding text is lost. For example the text at the beginning of a page probably has nothing to offer regarding an image positioned at the end of the same page. Also, the tradeoff between information gained and storage space and computational power is rather poor, making whole document capture not an efficient solution.

We note that the four attributes described above can be easily extracted from every document based on existing hypertext structures. In the paragraph below we define each of the selected attributes described above.

- **Image filename**
  Image file name, simply image title, is a single word or a sequence of ASCII characters that in the best case indicate the main event, place or object that the image is concerned with. By "*best case*" we mean that often images are given file names that are unrelated, some times even confusing with the image itself. The image filename is can be extracted from the ***<img>*** (image tag) or ***<a>*** (image hyperlink tag) in a HTML document.

- **Alternate text**
  Its origin is from the early stages of Internet, when certain browsers could not display images therefore, a text description was often provided as well. We define alternate text as a word or a sentence that usually represents a text abstract of the image semantics. Alternate text, when available, is extracted from the ***alt*** attribute of the <*img*> tag.

- **Page title**
  It is usually a short text or a few sentence description about the content of the specific Web page. Since most of the images are used for adding visual content to Web pages, the page title

provides a certain amount of information about every image that the page contains. Almost every HTML document has a title, so it can be extracted from the ***<title>*** tag.

- **Image caption**

  While the three previous attributes are well defined in hypertext structure, caption definition is an issue that requires heuristic techniques in order to identify and extract. It is the part that provides the most important semantic information yet the most abstract one. Generally speaking, we define image **caption as the surrounding text in or close to the image tag**. It can range from a few words to few paragraphs that contain many sentences of text.

Since image caption is not strictly defined in existing hyperstructures it might be confusing let's try to expand the meaning by giving some examples on how we perceive this attribute. Let assume that in both examples the target image is "*logo.gif*".

... </td><td>Our company's logo is registered since 1990. It represents the dynamic of our firm in computer software <img src="logo.gif" alt="software logo" ><br>Copyright ... </td><td> ...

...<p>Our company logo is registered since 1990.</p>
<a href="http:/.../logo.gif"></a>
<p> It represents the dynamic of our firm in computer software. Our logo is copyrighted ... </p>...

In the first example we consider the caption of the image to be the text inside the table data tags. Since the image is included inside the <*td*> tag the text included there can give us a semantic description about the image itself. In the second example the image is within a hyperlink tag. We capture the caption as the text that surrounds the image i.e. the text included in the paragraphs right above and below of the image position assuming that since they are neighboring paragraphs the text they contain provides some semantic information regarding the image.

A special case is images incorporated to HTML *table* structures. We consider that close text description here is only the text terms that are in the same *data cell* with the image. In doing so we guarantee the immediate relation of the extracted text with the image and avoid the large and probably too irrelevant text contained in the whole HTML table. Considering the vast and sometimes irregular use of HTML we consider this to be the closest approach to our needs.

2.4 Selected image representation

The next step is to find a more organized structure that can represent the image attributes selected and their semantic meaning more adequately. We adopt the model of [16] referred to as "Lexical Chains". A *lexical chain is defined a sequence of words semantically related to the image.* A lexical chain might contain a few words to some sentences. Each image attribute is represented by a lexical chain, thus having a *page title* lexical chain (*PLC*), an image title lexical chain (*TLC*), a alternate text description lexical chain (*ALC*) and finally a caption lexical chain (*CLC*) that include the whole extracted image caption. Normally, only the CLC may exceed the limit of one sentence, since the image caption in some cases may include a few paragraphs. All the other lexical chains are usually confined to a few words or a single sentence.

Yet this representation on its own may not be enough to fully catch the semantic meaning [16]. Therefore, two other lexical chains that are constructed from the caption LC, since the caption it the only attribute that may include multiple sentences. The first one, called **sentence lexical chain** (SLC) represents a *single sentence in the image caption*.

The second one, called **reconstructed sentence lexical chain** (*RSLC*) is a *new sentence made up of two related sentences. By related sentences, we mean that they share at least one common word*. Once a common word is found, the sentences are split in two, the first half of the first sentence and the second half of the second sentence forming a reconstructed sentence LC. The remaining halves form the second RSLC. Let's try to explain the above with an example on how the SLC and RSLC are constructed from a CLC. Let us assume that the extracted caption is the one described below:

CLC : " *Greek prime minister simitis visited belgium.*
Government officials welcomed simitis"

We see that there are two sentences in the above CLC so the produced SLC are the following two:

SLC 1 : "*Greek prime minister simitis visited belgium*"
SLC 2 : "*Government officials welcomed simitis*"

We also observe that the two SLC share one common word, that being the word "*simitis*". The reconstructed sentence lexical chains are :

RSLC 1 : "*Greek prime minister welcomed simitis*"
RSLC 2 : "*simitis visited belgium government officials*"

The way that RSLC are created does not depend on the number of common words: for more then one the procedure is just the same. Nor it depends on the total number of words a SLC has: defining the half of each sentence is an implementation issue.

Why introduce the latest two lexical chains? As stated at [16], by introducing the options of SLC (sentence lexical chain) and RSLC (reconstructed sentence) we achieve overall better semantic capture of the image's surrounding text (image caption in our representation). So we have the representation consisting of six lexical chains, each one of them capturing a portion of the semantic structure of the image. **TLC** (*title*) captures the main theme of the image, **PLC** (*page title*) shows part of its content, **ALC** (*alternate text*) provides a short text description about the image, **CLC** (*caption*) keeps the whole overall image semantics, **SLC** (*sentence*) captures the semantics of a single sentence in the image caption and **RSLC** (*reconstructed sentence*) captures the semantic meaning of related sentences.

For the user query we can use the same representation since it is generally a word or a sentence of keywords that describes the desired image. Therefore *we can present the user query as a lexical chain of words*, cleared off of stop words and stemmed. Certainly, we represent it as a **Query Lexical Chain** (*QLC*). The image and the user query representation are illustrated below:

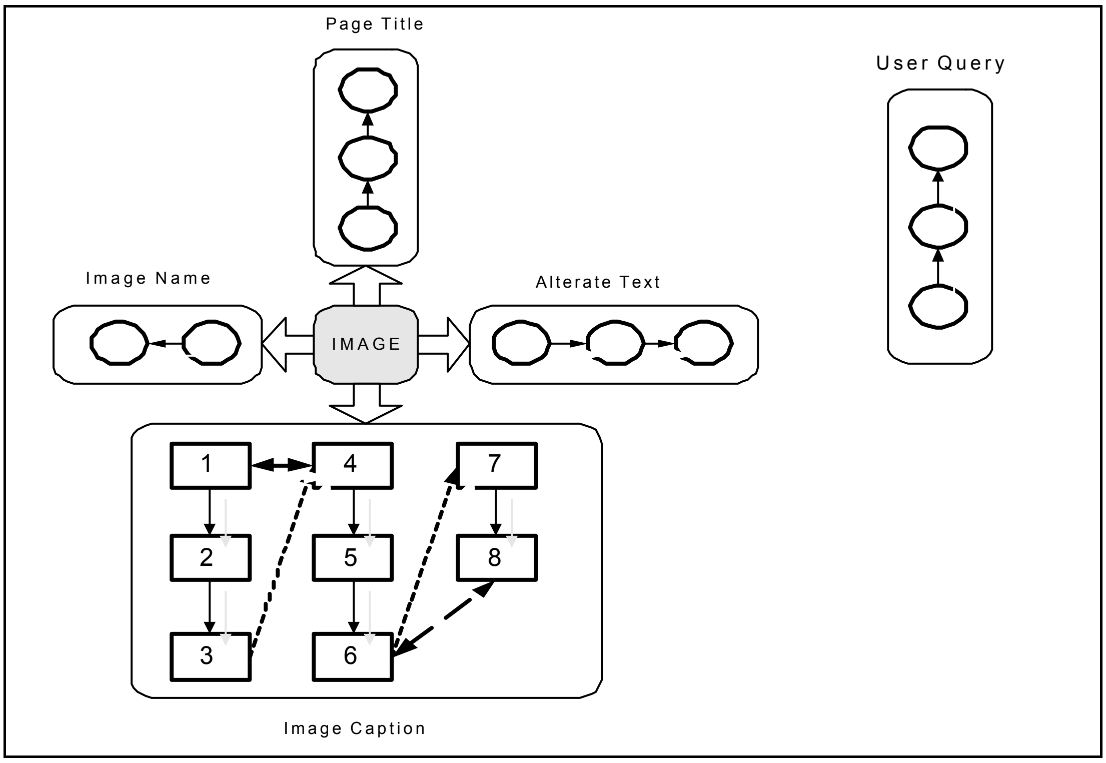


Figure 5: Lexical chain representation of image text description

The circles represent keywords whether the rectangles within the caption square represent SLC. An RSLC is represented as two SLC connected in both directions while all the SLC represent the whole caption of the image.

2.5 Exactness – Noise Removal

Questions may arise at this point regarding the exactness of the above representation. We all notice that some words do not offer any real semantic information. Also, other words, in linguistic terms, refer to a common root therefore often provide the same semantic meaning. For example:

CONNECT
CONNECTED
CONNECTING
CONNECTION
CONNECTIONS

Obviously in same cases words might seem unreasonable to truncate words to a common root (for example "relate" to something and "relativity" of physics theory) but frequently, the performance of an IR system will be improved if term groups such as this are conflated into a single term. Also the *space efficiency* criteria and noise insensivity are not upholded in this common root case. That is why we require that each extracted Lexical Chain is strip off of stop words and stemmed before processing further in the system.

Stop words are the common clauses of English that contain no real semantic information plus they add a lot of noise into the representation. Examples are the terms*: I, somehow, whatever, thus, will* etc. As such, these can be safety removed improving the whole precision while keeping information intact. In our system we chose the Salton's SMART [9] block list which contains more then 4000 suffices and common words of contemporary English.

Stemming on the other hand is a more complex problem that has been approached with a variety of methods including suffix removal, character truncation, word segmentation and linguistic morphology. In our system we selected the Porter stemming algorithm [36], [37], that implements suffix stripping, since it works quite well, is quite fast and already implemented in some of the most important computer languages. The results achieved are quite satisfying. The version adopted in our system is the POSIX C implementation [38]. Let us consider the case below:

ALT : "*This image shows the connection of our national computer brand with other institutions*"

At first step stop words are removed. Here being the words "this", "the", "of", "our", "with", "other". The derived lexical chain:

ALT : "image shows connection national computer brand institutions"

The final lexical chain after stemming is performed will be the following :

ALT : "*imag show connect nation comput brand institut*"

We see that we have significally reduced storage requirements and at the same time kept the most important semantic information of the specific Lexical Chain while shielding our representation from a significant amount of noise (stop and unstemmed words).

2.6 Attribute Importance

Nevertheless, the representation in it's current form is not expected to give good results since it lacks the ability of capturing the relative importance of the various lexical chains. For example, page title and alternate text play different role in the representing of the image semantics. Therefore they must also have a different role in the similarity calculation. That is the reason that we divided the image caption in three lexical chains. We want to differentiate the importance of each sentence due to their position within the image caption.

That leads us to the conclusion that the three LC in the image caption are not of equal importance. If the same word in a query appears in a SLC, CLC and RSLC, the SLC possesses the most important semantic information, followed by RSLC and finally by CLC because SLC is more semantically structured then RSLC, which is more structured then the loose CLC. Therefore their importance order if the following: SLC > RSLC > CLC. For example, if a query matches only a specific LC in each image, then the image that matches a SLC is probably the most relevant, followed by the image that matches a RSLC and finally by the image that matches only a CLC.

A weight model approach must be adopted in order to capture the relative importance of all lexical chains produced. In this way we assure the property of exactness since higher weights are assigned to attributes that contain the most important semantic information. Below, we explain how similarity between two lexical chains is computed in our representation.

2.7 Similarity Computation

Here we try to determine a similarity measure between a query and an image representation. Since all semantic information is stored in the form of list, we propose a list space formula to calculate the similarity between two lexical chains as follows :

$$\texttt{Similarity} \equiv \frac{\sum_{i=0}^{list1.size} \sum_{j=0}^{list2.size} e_i * e_j}{\sqrt{list1.size} * \sqrt{list2.size}} * MatchScale$$

where e_i and e_j are matched words in list1 and list 2 respectively. Two words are matched if they are the same word. We can describe the formula as the number of matched words multiplied by MatchScale and divided by the production of the square roots of the respective list's (lexical chain) size. MatchScale is defined as the closeness of two lists from the view of match order. Let's explain what do we mean with an example. Assume the two lists are the following:

L1 ≡ ”Greek president received a beautiful gift from the visiting kids”
L2 ≡ ”The gifts given to kids from the president were lovely”

We note that there are three matching words for the above LCs. In the first one, the matching words are in order of “*president gift kids*” and in the second they are “*gift kids president*”. The orders of the matching words are not the same. We treat each string of common matched words as a child Lexical Chain of the original LCs. Obviously, the closer the matched orders of two children LCs are, the closer the semantics of the original LCs are. Therefore, we need another formula to calculate the closeness of the matched order of two LCs. From the formula for the angle between two nonzero vectors in 2d-space we define MatchScale as below:

$$Matchscale_{v1,v2} = \frac{v1 \bullet v2}{\|v1\| * \|v2\|}$$

Where **v1** and **v2** represent the child LCs of the original LCs. The element value is their position in the respective LCs. But the dot-product between two Lexical Chains is redefinded as the following:

$$v1 \bullet v2 \equiv \sum_{i=1}^{v1.size} v1_i * v2_j$$

Where $\mathbf{v2_j}$ is the matched word in **v2** for $\mathbf{v1_i}$ in **v1**. As mentioned above, two words are matched as long as they are the same. With MatchScale, we count on determining the similarity between two Lexical Chains since the higher the MatchScale score, the higher the similarity. However the two LCs might not be semantically related. For example let us consider the query “*greek flag*” and two images, **I1** with several concurrencies of “*greek*” in it’s CLC and **I2** about the greek flag but with only one occurrence of “*greek*” in it’s CLC. The first image would get a higher similarity then the second one even though the second is the most relevant one. Therefore another parameter is needed in order to ensure that our LCs are semantically related to each other. As stated in [16], the LCMatchLevel parameter is introduced; defined as the minimum match level threshold for a LC to keep its original semantics. We say that a LC is semantically related to a Query LC only if its LCMatchLevel is equal or greater then the QLC's match level threshold. The higher the match level threshold, the fewer but more relevant result are. The lower the match level threshold, more but probably irrelevant results are. Therefore we can say that the match level determines if the LC is semantically related to a QLC and the similarity formula presented above determines how well it is related to the QLC. Nevertheless, in our implementation results have shown that the MatchLevel criterion is a of disputable precision since it tend to cut both bad and certain good images with low score from the results but that will be explained in details below.

2.7 Similarity between image and query

From the discussion above we know that the most important attributes of an image are captured and represented in a number of Lexical Chains. Also, queries are represented as Lexical Chains. To calculate the overall similarity between an image and a query we use the following formula:

$$\begin{aligned} Similarity \equiv\ & S(TLC,QLC) + S(PLC,QLC) + S(ALC,QLC) \\ & + S(CLC) + \sum_{i=1}^{SLC.number} S(SLC_i,QLC) + \sum_{i=1}^{RSLC.number} S(RSLC_i,QLC) \end{aligned}$$

Also the overall image match level is calculated as:

ImageMatchLevel(? LC,QLC) = TLC.weight * LCMatchLevel (TLC,QLC) + ALC.weight * LCMatchLevel (ALC,QLC)+ PLC.weight * LCMatchLevel (PLC,QLC) + CLC.weight * LCMatchLevel (CLC,QLC)+ SLC.weight * LCMatchLevel (SLC,QLC)+ RSLC.weight * LCMatchLevel (RSLC,QLC)

The overall similarity of a user query and an image represented by the similarity sum of all the lexical chains that represent the image while the degree of relevance between them is computed from the ImageMatchLevel formula. A very sharp issue here is the weight assignment-value since they play a vital role in the ranking quality and overall results. In contrast with the original ChainNet model we believe they must not be assigned as in the original Chainnet model [16] based on (at best) discussable observations but rather user defined, adapted (on real-time) to the user preferences.

CHAPTER III

RELEVANCE FEEDBACK

As analyzed in details above the peculiarity of the image retrieval problem need other relevance feedback techniques rather then the classic ones. Nevertheless the same problems remain. The biggest one being the subjectivity of the human perception: the way people perceive and judge is based on subjectivity that varies from person to person. The other is the computer-centric issue, simply defined as the weight problem and the inability to combine high level concepts with user subjectivity.

Users can't be asked to define a priori a class of weights simply because that is beyond the understanding of most users and might have a disastrous impact on the quality of the results. Furthermore, too many details in the feedback process might disorganize and burden the user. A general approach is the one that wants the user to simply mark and distinguish the relevant from the irrelevant results and submit this information to the system to be used properly in the after steps. A key issue here is the information provided: if user marks few images the results might not be satisfying since not enough information is provided. On the other side, if the user marks too many images then the results might excel yet this depends totally on how the user is willing to go through several pages of results and evaluate them; a rather annoying thing for most normal users. Several pseudo relevance feedback techniques that does not require any feedback from the user but instead try to recalculate similarity according to shareholding or other criteria. Yet, their performance is expected to be below the user-engaged techniq ues. Nevertheless, most relevance feedback techniques depend on user fed back information on previously retrieved objects in order to combine and adjust an existing query to the user's preferences. Several techniques are described in details below.

3.1 Semantic Accumulation

Based in the lexical chains representation this method allows the user to pick the most relevant image from the results. Then it gathers and accumulates the semantic information (the lexical chains) of the selected image in order to construct a new, richer query. Besides new terms, noise will be added into the query as well. Therefore rather then the whole image a single, most related – highest similarity lexical chain is used. The most related lexical chain is calculated from the list's similarity formula:

$$\text{Similarity} \equiv \frac{\sum_{i=0}^{list1.size}\sum_{j=0}^{list2.size} e_i * e_j}{\sqrt{list1.size} * \sqrt{list2.size}} * MatchScale$$

The steps, described below, are illustrated in figure 1:

1. Search using user query.
2. User selects feedback image.
3. Extract most related LC from the selected image.
4. Merge query and lexical cha in to obtain new query.
5. Make search with new query (step 1)

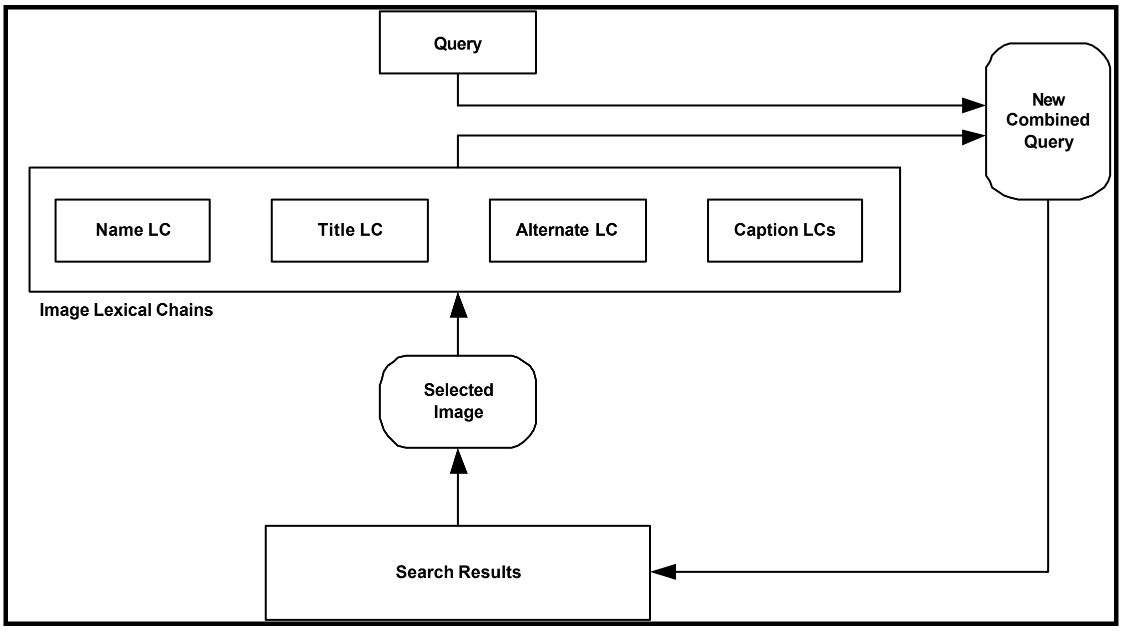


Figure 6: Relevance feedback by semantic accumulation

As the user continues to select most relevant images, the submitted queries accumulate and therefore carry richer and richer semantics that represents user's needs. We must note that this approach has certain drawbacks that at the same point are her spearhead. Query enrichment might insert new images to the results yet at the same time it might as well insert many unrelated images in regards to the first query. Thus we may say that this method practically would either narrow the result to a very good set either widen them to a large set of not necessarily relevant images. Also, a rare sub case is that if the extracted LC with highest score and the old query are the same there is practically no feedback at all since the search would return the same results as before.

3.2 Semantic Integration Differentiation

This approach is an improved version of the previous technique. Since selecting one image at a time is rather tedious and time consuming now the user can select both relevant and irrelevant images judged as such. The system integrates the relevant feedback images to construct a new query. After that, the system differentiates the irrelevant images from the returned results based on the feedback irrelevant selections. The system extracts the most semantically related LCs from each image marked as relevant and the previously submitted query in order to make a new, enriched query. Furthermore, the system extracts the least semantically relevant LC for the images marked as such in order to form a *negative query*. Results returned from the new query are matched against the negative one and dropped if more similar to the negative lexical chain. The steps, described below, are illustrated in figure 1:

1. User selects a number of relevant and irrelevant images.
2. Extract the most related LC from each relevant image.
3. Combine query and extracted LCs to obtain a new query.
4. Search using new query.
5. Extract the most un-related LCs from the irrelevant images and combine in a query-form lexical chain.
6. For each image, remove it from results if more similar to the LC obtained from the irrelevant images.
7. Go to step 1.

The main issue in the above technique is the fact that the query enrichment might have a negative effect, inserting too many new terms that might be irrelevant with the initial query. Also, if the combined negative LC that is too long i.e. contains too many words then probably the set of results will be very restrained both in precision and in number of elements. As in the accumulation feedback there is absolutely no weight rearrangement at all, instead relevance is achieved by expanding the results each time with related queries.

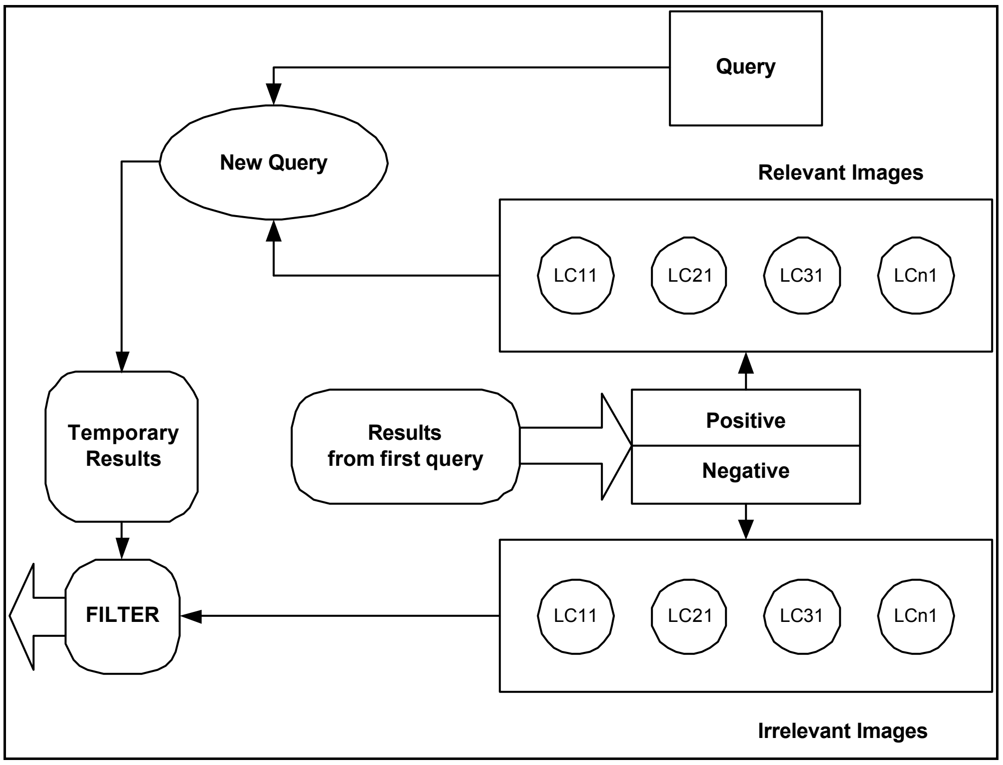


Figure 7: Semantic integration differentiation

3.3 MARS Relevance Feedback (Rui – Huang)

Used in the MARS retrieval system this method is considered to be a reference to all feedback related techniques. According to Rui, all images are treated as a multimedia objects with certain features while there may be more then one representations for one specific feature (color histogram and moments for example). The concept is the same : user is asked to judge some or all images with 5 level of relevance : highly relevant, relevant ,no opinion, irrelevant , highly irrelevant with respective scores of 3, 1 ,0 , -1, -3. Then, based on users preferences the system tries to find which features the user likes or more important to the user. The contribution of each representation to the overall score of the specific attribute is auto calculated based on mean and deviation values. For example if all relevant images have similar values for the color attribute then this attribute is a good indicator of the user's needs. If its values are very different then probably it is not a good indicator therefore its weight must be properly rearranged. Therefore inner weights (e.g. all text attributes or all visual feature attributes) are calculated as 1/deviation, or any proposed combination that might suit better, for each representation. Outer weights (e.g. text or visual features) are calculated directly from the user judgment as the normalized sum of the user scores (where relevant has a score of 1 and highly irrelevant one of –3) for the best NRT results of each attribute.

1. User marks some images.
2. Obtain X best results.
3. Calculate inner weights.
4. Calculate outer weights.
5. Recalculate similarity score for all images.
6. Go to step 2 or Exit.

Outer weight assignment is done as in the algorithm below:

$$RT = [RT_1 \ldots RT_i \ldots RT_{NRT}] \text{ set threshold results.}$$
$$RT^j = [RT_1 \ldots RT_i \ldots RT_{NRT}] \text{ set threshold for } j \text{ attribute.}$$

Sort results according to score for each attribute and select the *NRT* first. Therefore we have as many RTs as presentations (e.g. text or visual) that are considered in the total score calculation.

```
For all attributes
    IF  $RT^j{}_i$ e??a?st? RT
        $W_i^j = W_i^j + SCORE\ i$   IF marked from user.
        $W_i^j = W_i^j + 0$          Not marked from user.
    Set $W_i^j = 0$   if $W_i^j < 0$

    $W^T$ - Sum of all new weights.
    $W^T = \sum W_i^j$

    Normalize all new weights $W_i^j = \frac{W_i^j}{W^T}$
```

After the weight adjustment process image similarity is calc ulated by the formula below:

$$\text{Score}_i = \text{W}_{text}^{outter} * \text{Score}_{text} + \text{W}_{visual}^{outter} * \text{Score}_{visual}$$

$$\text{Score}_i = \sum_0^{nr.attrib}(\text{w}_j^{inner} * \text{Score}_j)$$

Where: *i* text or visual

j number of attributes for visual or text

Image's similarity scores are normalized before entering the feedback process i.e. all set in the [0, 1] space. Also all weights are analyzed such as to have a sum of 1. Therefore, this method does not try to find the ideal query instead it tries to calculate the ideal weights for each attribute close to the user preferences (reweighing approach).

Rui method is expected to have the best result since user preferences are quite detailed yet it depends heavily on the information provided i.e. better results will appear if the user judges more and more images. Also, since the MARS system was only content-based, some changes made to fit the method into our system are described in detail in the next chapter.

3.4 FALCON – Related Relevance Feedback

While the above feedback tries to calculate the ideal weights and the other methods (accumulation and integration) try to find the ideal query, FALCON depends on a dissimilarity function that recalculates the score for all images in the initial set, based on user judgment. The user is asked to mark at least one relevant image from the results which are considered as a set of "good points". A new, dissimilarity measure is calculated: the sum of the power of the distance between an image and the user provided images (good points) divided with the number of these good images. Therefore, similar to the query are those images that have a small distance from the set of good points. Distance of a candidate image is considered the difference between the image's score and the user-selected good image. A distance is calculated for all the images marked from the user (set of good points).The dissimilarity functions and the steps of the algorithm are described below.

$$D_G = \frac{1}{k} * \sum_{i}^{k} d(x_j, g_i)^a$$

Where x_j - Candidate image score

g_i - Score of user marked relevant image

$d(x_i, g_i)$ dissimilarity function implemented as $x_j - g_i$

k – Number of user provided relevant images.

a parameter set to –5

1. User chooses set of relevant images.
2. Construct or expand the set of good points.
3. For all images calculate distance from good points.
4. Calculate dissimilarity for all images based on function.
5. Display.
6. User adds other images to the good set or EXITS.
7. Go to step 2.

The sum of the a-th power of this distance divided by the number of the user selected images provides the dissimilarity score of the candidate image.

The set of good point is expected to change since user might add other relevant images. Therefore the dissimilarity score for each image changes as well. We must note here that changes of the value of a parameter are expected to change results as well. Also, we suspect that for the algorithm to work properly from the first stages the user needs to provide many relevant images such as to establish a clear set of good points. Nevertheless, FALCON is an interesting, new method when compared to the other proposed relevance feedback approaches.

3.5 Pseudo-Relevance Feedback

In all the above methods the information provided from the user is of crucial importance for the whole feedback process. Nevertheless, the normal user sometimes may find it rather disturbing to mark and evaluate even some of the results. Therefore some pseudo relevance feedback techniques are called to fill the gap. Pseudo meaning that if no information at all is available from the user then the systems itself is called to rearrange the results by some techniques: shareholding, clustering, automatic weight recalculation etc.

3.5.1 K-Means Clustering

The same principals of the well known clustering algorithm can be applied as a pseudo relevance feedback technique. Considering the fact that we expect the upper part of the results to be quite relevant and the lower part rather irrelevant we can apply the K-means algorithm in order to try and cluster the results into good, don't care and irrelevant to the query. The number of centers may vary but in our implementation we adopt a 3 center approach. We can define some initial center for good, don't care and irrelevant and run the algorithm several times. In order to improve result we have certain options like displaying only the good results, or the good and the don't care etc. The algorithm is as below:

1. Find center through thresholding or use given centers.
2. For each result include in the category whose center is closest and adjust the center properly.
3. Go through clustering again with derived center.
4. Display one or more categories.

The selection of the center here is of interest: if good threshold is too high, then some relevant images might get into the don't care category. If irrelevant threshold is too low then some probably relevant images might be considered as irrelevant. More then 3 centers can be offered too, like in the score model of the Rui approach, yet we consider the 3 center selection more close to our needs. Also, the repetition of the center-calculation part might be of some interest yet it's expected that after 2-3 steps the results will not change anymore.

CHAPTER IV

System Description - Implementation

In this chapter we will explain the structure of our system, the way the desired attributes are extracted, the interaction with the user, the relevance feedback process and certain implementation details. The core of our system is the image and html documents database. It contains more than 200.000 images and the respective web pages that incorporate them. The crawling process preserves the link structure of the downloaded pages. Image text description and visual feature extraction are processed prior to image storage and indexed properly. The user interacts with the system by submitting queries in text form i.e. keywords that best describe what he is looking for. The system returns a set of best-matched, sorted by similarity while providing the user with several feedback methods in order to narrow, specify and generally improve the results according to his preferences.

4.1 Crawling

Considering the huge size of the WWW and its millions of images it would be wise to have a common point of reference from which to start. We consider Google (images.google.com) as such since its benefits (huge database, speedy retrieval etc) are well known. We use Larbin [41], a simple yet very efficient web crawler (modified to meet our needs and specifications by Epimenides Voutsakis) to gather our data set. The initial data set is formed by submitting to Google certain keywords asking for web pages that contain images relevant to the query keywords, such as to form a large enough data set. Google's restricts query reply to a set of less then 1000 answers (when available) might be a problem. Therefore, the initial data set is further expanded with pages pointed to or pointing to web pages in the initial data set. The process is repeated twice so that the final data set contains a data set of interconnected pages; locally storing all data for further use. Each HTML document is analyzed as a whole and added to the overall hyperlink structure of the data set while all images that it contains are downloaded. Appropriate text descriptions and visual features are extracted from all these images and stored in the database along with the original pages.

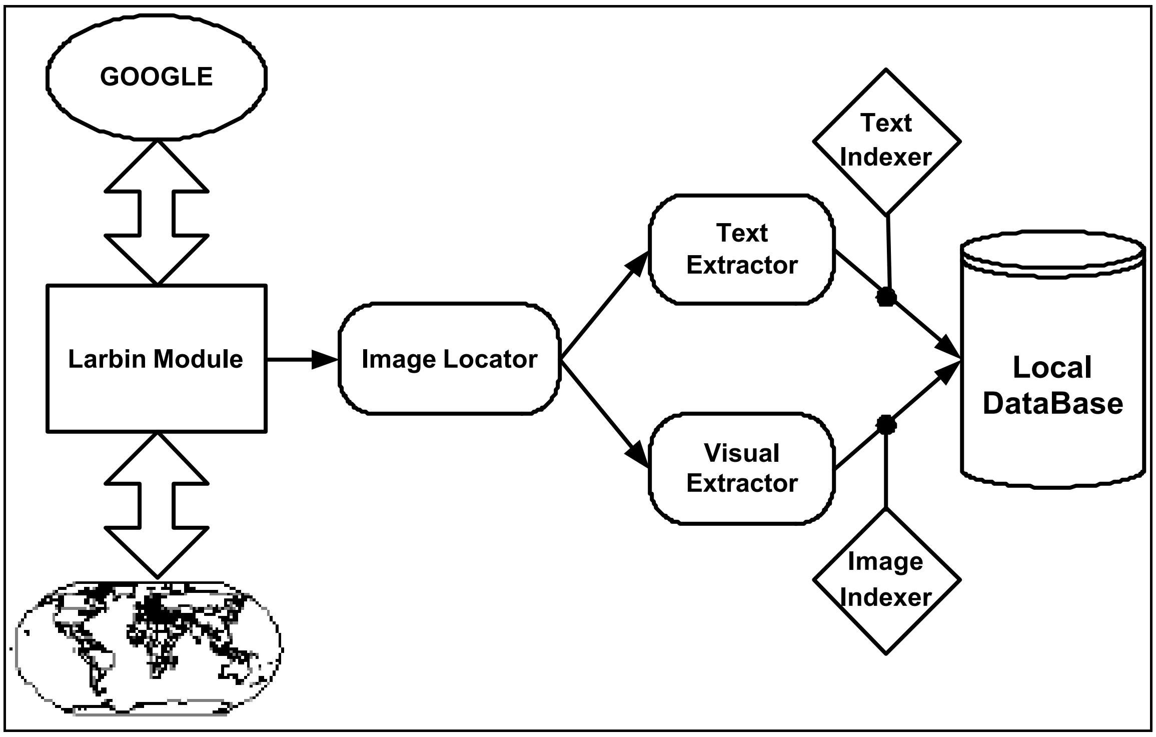


Figure 8: Web crawling and indexing in our system

As illustrated above, the initial crawling begins by interacting with Google, submitting queries and by expanding the results obtained from Google. Documents are passed through an image locator module and for each image visual and text features are extracted and stored in our database. More details on crawling and indexing can be found on the dissertation thesis by Epimenides Voutsakis at [42].

4.2 Visual features

While text features depend on the surrounding text of the image, visual features depend from the image raw data itself. The specification and definition of the appropriate image features is the key issue here. Also an automatic method of extraction, categorization and classification must be take place since human annotation suffers from the problems named before (subjectivity, speed, coherence etc). Since our approach concerns only the logo/trademark category we can narrow our criteria. We accept that logos and trademark are usually small graphics images, with no too many intensity levels and or colors, small in size and in storage space. More details on image features and image analysis can be found at [43]. In brief, a set of features are:

- Intensity histogram.
- Color spectrum for each of the H, S of the HIS color space.
- Frequency spectrum or variance of frequency as a function of the distance **r** from the origin. Only the content of low/high frequencies is of interest therefore is computed as the integral of the 2D frequency spectrum all over **?** (theta).
- Moment invariants of the whole image.
- Image size and dimensions.

Once defined the next issue is to find an appropriate function to determine if an image is a logo or trademark and to find a similarity function between two images. A first criterion for the logo detection is the image size: logos tend to be small. Also, detection can be based on thresholds defined on feature histograms such as to distinguish logo from non-logo images. Certainly there are other criteria as well but the main idea is that logos and trademarks are small images in size, graphic images rich in frequency content, with not many intensity levels of gray or color. Nevertheless this is not always the case. At [42], the method for logo and trademark detection computes the probability which corresponds to the likehood that an image is logo or trademark. Once this defined, it all comes again to a thresholding problem since with a high threshold on this probability-score we might exclude many good candidates while with a low one we might insert many no relevant images (noise in our case). The decisions regarding this cant be made manually. This score is computed by a machine learning algorithm: based on positive or negative examples of logo and non-logo images it determines the criteria for distinguishing between logos and others providing in the end a simple yes/no answer and a logo-probability score for each image.

A similarity function for the detection of similar images is the feature distance. This includes the distances between histograms that are computed using the normalized histogram intersection value while for the other features this is defined as the Euclidean distance. We use the visual features to compute a similarity score as the sum of all the above attributes:

$$\text{Score}_i = \sum_0^{nr.attrib}(w_j^{inner} * \text{Score}_j)$$

Where: *attributes* is the number of visual features (histogram intersection, geometrical distance, Otsu values for I, H, S, F), w_j^{inner} is the weight given to the specific attribute (initially set to 1) and Score $_j$ is the similarity score of the image.

Visual feature extraction and indexation is provided by Epimenides Voutsakis [42] while logo probability, the training algorithm (Wekka) and image similarity functions are implemented from Vassilis Hatzidiakos [43].

4.3 Extraction – Indexing

All stored web pages are passed to the extractor whose purpose is to extract the desired text and image features along with the logo-score (probability) for each image and pass them to the indexer. We must note here that regarding the image caption we chose to restrict the size of the caption, which can be rather big with certain unpleasant effects, to a maximum of 30 words left and other 30 right of the image's position. This way we ensure the most significant keyword; those very close to the image and avoid at the best some noise and unnecessary information. All text output are then cleared off noise: using SMART's block list [9], [10], with more then 5000 elements, such as to strip off common words (noise) and through a stemmer to deduce to root of words. We choose to apply the POSIX C [38] implementation of the Porter stemming algorithm [36], [37].All output is then indexed using inverted files (all text extracted to image in a special ***image-text*** index) and stored locally for use in the retrieval process. The extracted text descriptions of all images are indexed in a new inverted file that contains only the terms found in the page title, image name, image alternate description and image caption (if available). The visual features and other extracted information are indexed in a similar way, as stated in [42].

4.4 Database

The database is implemented using Berkley DB [44]. The indexes we use are inverted files. Here, each document is presented by a list of terms stored alphabetically in the index file. For each term, a list of documents containing the term is maintained. So instead of a full scan on the document, we can scan really quickly the inverted file. Concerning our system (more information is indexed too e.g. visual features) text features such as: image caption, page title, image name and alternate text are indexed in an ***image-text*** inverted file. When user submits a query this is the index that is searched first to locate which images contain the user terms and pass them along with the respective full text to the similarity calculation module. Term extraction (stemmed and stripped off common words) and indexing process is implemented thoroughly in Perl. We choose Perl in order to take advantage of its excellent string and pattern handling and of its data structures, especially hashes that prove of significant help in the similarity calculation part of the whole process.

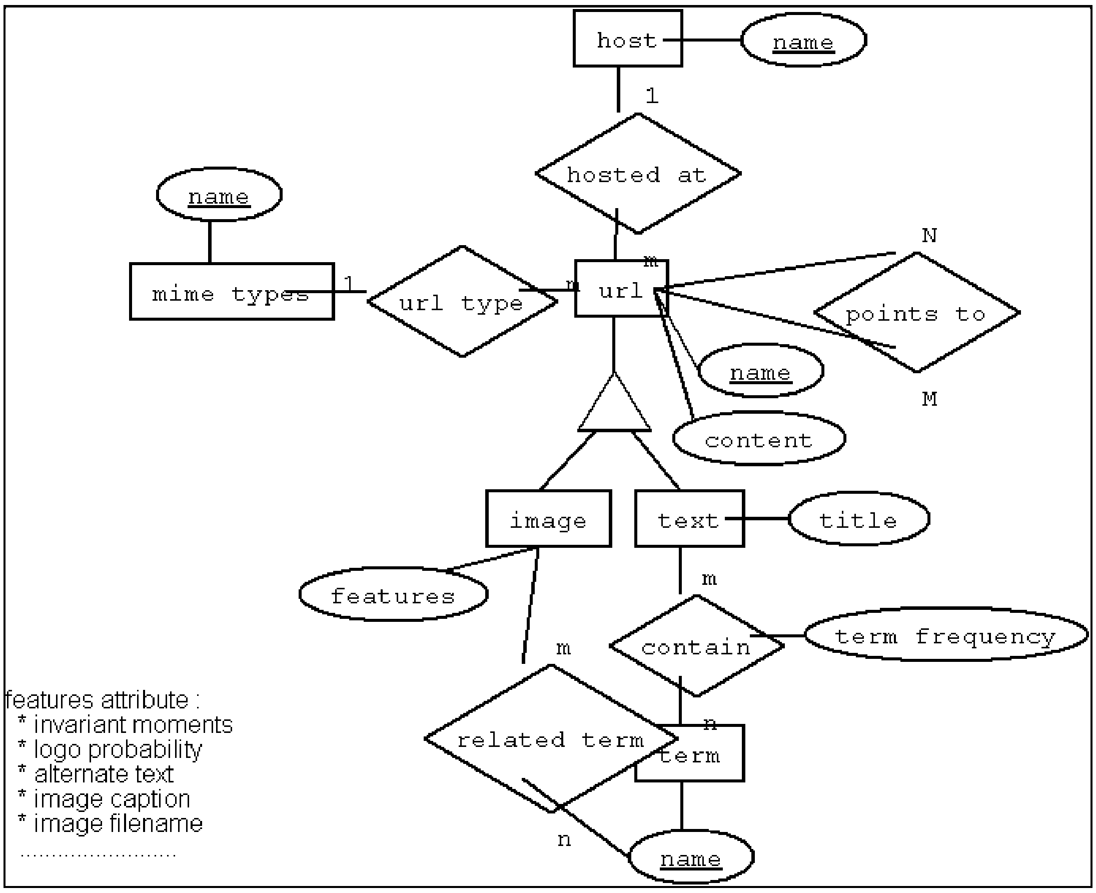


Figure 9: Model of the local of database

An ER model of our system's database is shown in the figure 9.

4.5 Retrieval

The user interacts with the system through a web-based form interface by submitting one or more keywords that represent the image he is looking for. The query keywords are stemmed and passed to the image's inverted file (the file that contains the index of terms found in the page title, image name, alternate description and image caption) to obtain an initial set of results that probably are relevant to the user's request. The next step is to call the proper function in order to determine the similarity of each image (i.e. its description) to the user's keywords and rank them in the proper order. The steps followed are as below:

1. The user submits one or more keywords.
2. Image inverted file is searched for the query keywords and returns image text containing **all** keywords.
3. Similarity calculation of image text description and user query based on lexical chain's model. (section 2.7, page 28)
4. Remove images with low MatchLevel (not close enough to query semantics. (page 30)
5. Return ranked results.
6. Proceed with feedback or start another query.

The interface with which users interact is below:

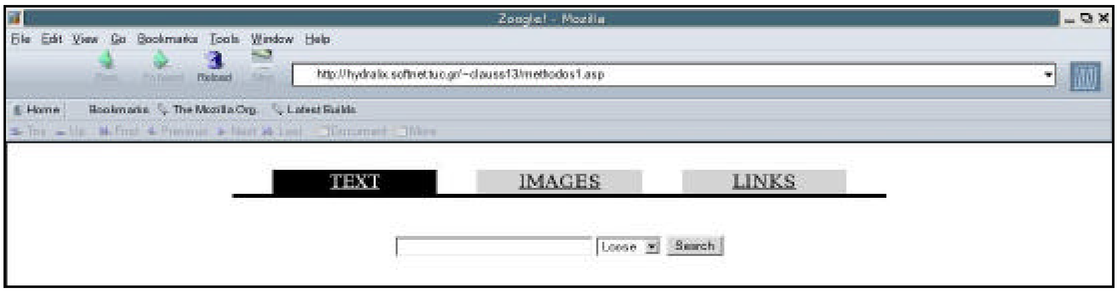
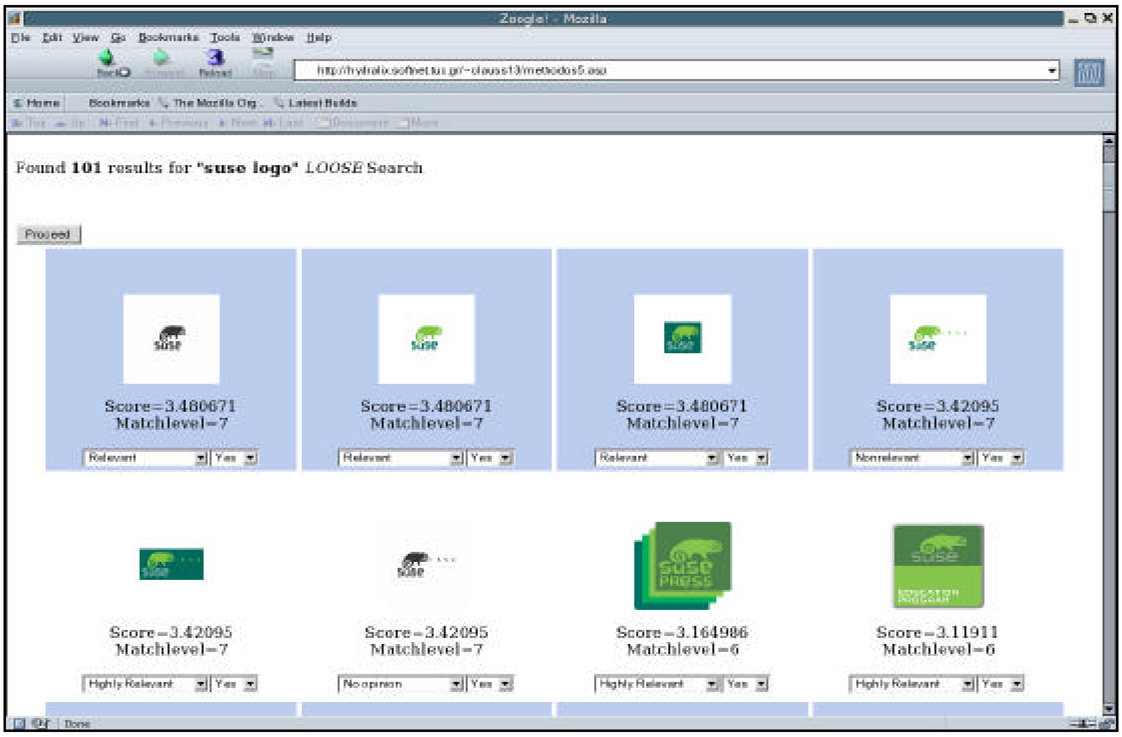

Zoogle! - Mozilla
File Edit View Go Bookmarks Tools Window Help
Reload
http://hydralix.softnet.tuc.gr/~clauss13/methodos1.asp
Home
Bookmarks
The Mozilla Org.
Latest Builds
TEXT
IMAGES
LINKS
Loose
Search

Figure 10: System's initial interface

The user enters query keywords and defines the MatchLevel (page 30) of the returned results. MatchLevel is a rather intriguing parameter. As explained in page 30, MatchLevel is the number of distinct query terms that a lexical chain should contain in order to be considered relevant to the query lexical chain. While it definitely removes some images not relevant to the query, we believe it will also remove some probably relevant images as well. That why we offer a two level approach to this feature to be selected from the user itself. Images with a **loose** MatchLevel value of less then the size of the query are considered as irrelevant and therefore not included in the final results. By query size we mean the number of the keywords that make the query. If a **strict** MatchLevel value is selected from user than images with a lower value than 1, 6 * query size are considered irrelevant. The default value of the MatchLevel parameter is set to **loose**.

If the user submits more then one keywords the inverted file will return only the documents that contain all keywords. This way we keep a strict track of the user keywords and eliminate a lot of (probably) irrelevant images. For example, for the keyword "Linux" the inverted file would return an estimated total of 8000 images, for the word "logo" an estimated total of more then 12000 images. If we exclude all image text descriptions that do not have both terms (i.e. user query is “Linux logos”) and keep those that have booth, we get a total of 3000 images. As can be quickly noticed, we ensure the preservance of the initial user description-semantics and in the same time achieve a significant decrease of the total response time of our system since fewer similarity calculations are made.

The next step is similarity calculation for each image using the lexical chains approach. The lexical chains model is implemented in C (chosen primarily for its speed and flexibility). Also, since each lexical chain can be seen as lists of words, we find C data structures perfect for our implementation. We note here that all weights at this phase of the process are set to 1 except for the original Chainnet method where weights are set as described on [16].


Zoogle! - Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
http://hydralix.softnet.tuc.gr/~clauss13/methodos5.asp
Home Bookmarks The Mozilla Org Latest Builds
Proceed
Done

Found **101** results for **"suse logo"** *LOOSE* Search

Score=3.480671
Matchlevel=7
Relevant | Yes

Score=3.480671
Matchlevel=7
Relevant | Yes

Score=3.480671
Matchlevel=7
Relevant | Yes

Score=3.42095
Matchlevel=7
Nonrelevant | Yes

Score=3.42095
Matchlevel=7
Highly Relevant | Yes

Score=3.42095
Matchlevel=7
No opinion | Yes

Score=3.164986
Matchlevel=6
Highly Relevant | Yes

Score=3.11911
Matchlevel=6
Highly Relevant | Yes

Figure 11: Returned results

As seen on the above picture, the results (after the removal of images with lower Matchscale then the one required) are shown to the user along with the URL of the document (not present in the picture for better display) that contains them, the similarity score and the MatchLvel value. Finally, the user can go through one of the many feedback processes by providing needed information for the method chosen, as described in details in the former chapter, or submit a new query.

4.5.1 Retrieval – Chainnet

The first method implemented is the original Chainnet model described at [16]. The user query is represented as a list while the various text attributes form the respective lexical chains (implemented as list too). The similarity score and the Matchscale between the query lexical chain and an attribute are calculated by the formula:

$$\text{Similarity} \equiv \frac{\sum_{i=0}^{list1.size} \sum_{j=0}^{list2.size} e_i * e_j}{\sqrt{list1.size} * \sqrt{list2.size}} * MatchScale$$

$$Matchscale_{v1,v2} = \frac{v1 \bullet v2}{\|v1\| * \|v2\|}$$

$$v1 \bullet v2 \equiv \sum_{i=1}^{v1.size} v1_i * v2_j \quad \text{Where v1, v2 new lists of common words}$$

The similarity score for each image is the sum of the similarity scores of each attribute multiplied with their respective weights. On this method the weights are set as described in [16] i.e. to 0.6 for the page title (PLC) , 0.8 for the image name (ILC), 0.6 for the alternate description (ALC), 0.2 for the whole image caption (CLC) , 1 for the each of its sentences (SLC) and 0.5 for its reconstructed sentences (RSLC). Same weights are used to calculate the image MatchLevel (distinct number of words between attribute and user query). Images with lower MatchLevel than the one specified are removed as non-relevant. All results are presented in decreasing similarity score to the user. This method is available online at http://hydralix.softnet.tuc.gr/~clauss13/methodos2.asp

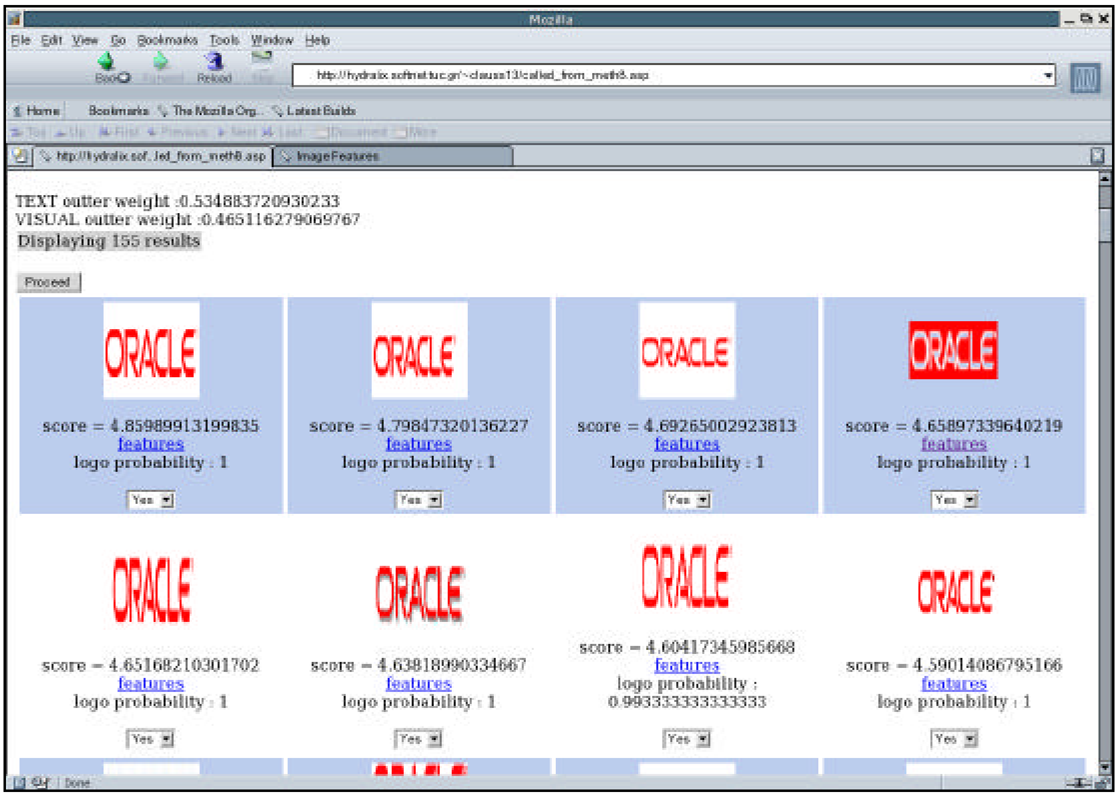
4.5.2 Retrieval – Modified Chainnet

We believe that weight assignment a priori is not the best possible approach. We believe that all weights should be the same and changed (adjusted) by the user by interacting with the system in a relevance feedback process. Therefore we set all weights to 1 for all attributes and expect (encourage) the user to proceed one of the many relevance feedback methods. The same formulas as above are used to calculate similarity score, MatchScale and MatchLevel. Except the original Chainnet approach, implemented strictly as stated on [16] this is the base approach for all our system. This method is available online at:
http://hydralix.softnet.tuc.gr/~clauss13/methodos1.asp

4.6 Relevance Feedback

We consider relevance feedback a very important part of the overall system therefore the proper attention is given to it. The picture below illustrates the general way the results are displayed after each step of the feedback process. Certain characteristics of the process (weights, example images etc) are displayed along with the total number of results. Similarity score, MatchLevel and logo probability (when available) are displayed too. Also if visual features are used, they can be viewed (image histograms, moments, size, aspect ratio etc) by the user for each image.

The user must provide both positive and negative results (when needed) in order for the implemented methods to give good results. Still, the user can provide only good data examples-points (for example Falcon) to the relevance feedback process. On the other hand, he cannot provide only negative data to the system since that kind of feedback is beyond the scope of this thesis. In this case, when user cannot find at least one or some good results among the initial set, we expect him to start and submit a new, better defined query.



TEXT outter weight :0.534883720930233
VISUAL outter weight :0.465116279069767
Displaying 155 results

score = 4.85989913199835 features logo probability : 1 Yes	score = 4.79847320136227 features logo probability : 1 Yes	score = 4.69265002923813 features logo probability : 1 Yes	score = 4.65897339640219 features logo probability : 1 Yes
score = 4.65168210301702 features logo probability : 1 Yes	score = 4.63818990334667 features logo probability : 1 Yes	score = 4.60417345985668 features logo probability : 0.993333333333333 Yes	score = 4.59014086795166 features logo probability : 1 Yes

Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
http://hydralix.softnet.tuc.gr/~clauss13/called_from_meth8.asp
Home Bookmarks The Mozilla Org... Latest Builds
http://hydralix.sof...led_from_meth8.asp
ImageFeatures
Proceed
Done

Figure 12: Rui Visual and Text interface

All relevance feedback methods are implemented as stated and explained in theory. Nevertheless, certain details must be mentioned in order to explain our implementation in order to explain the results in the next chapter.

4.6.1 Semantic Accumulation

This method, proposed as an extension of the original Chainnet approach by the same authors [16], provides relevance feedback by submitting new queries and not by reweighting the already returned results. The user is allowed to select only one relevant image (if more are selected the rest are rejected). The user selected image is passed to a C executable that will return as output the lexical chain with the highest score. We believe that query enrichment with words from the user-selected images is the heart of this feedback method. We understand and therefore implement it as the enrichment of the query with new words from the lexical chain with highest score of the selected image: meaning that we add to the query's keywords other keywords. Nevertheless there is always the possibility that the LC with the highest score is the

same as the query (in this case no feedback is achieved after all). The new query is processed similarly to the original query through the inverted file and will return results that will undergo the same procedure for similarity calculation as stated above. This method is available online at:
http://hydralix.softnet.tuc.gr/~clauss13/methodos3.asp

4.6.2 Semantic accumulation and integration

Semantic accumulation and integration is an improved version of the above method. Instead of only one relevant (positive) image the user can select more than one positive or negative examples from the answers. Positive images are passed to a C executable to extract the LCs with the highest similarity score in order to combine a new query. From each positive example the LC with the highest similarity score is extracted and merged with the old query to form a new, enriched query. Negative (irrelevant) images are passed to the same executable that will extract the LC with the lowest similarity score. All the extracted lowest score LC are then merged in single lexical chain. The new query is processed through the inverted file and will undergo the same similarity calculations as previously mentioned. The same process will be repeated with the negative LC serving as query lexical chain. Images whose similarity to the new query is bigger then similarity to the negative LC are presented to the user as final results. Images that are more similar to the negative LC than to the new query are considered irrelevant and therefore removed. The similarity is calculated from formula below where instead of the query list the negative lexical chain is given as input.

$$\text{Similarity} \equiv \frac{\sum_{i=0}^{list1.size}\sum_{j=0}^{list2.size} e_i * e_j}{\sqrt{list1.size} * \sqrt{list2.size}} * MatchScale$$

If the negative LC contains too many terms then its probable that the similarity score between image text attributes (lexical chains of caption, title etc) and negative LC is bigger than the similarity score between image text attributes and the new, enriched query. This will lead to zero results since according to the method all these results will be considered as irrelevant. This method is available online at: http://hydralix.softnet.tuc.gr/~clauss13/methodos4.asp

4.6.3 Falcon

Falcon method is implemented as stated in [31]. User must provide a set of (only) relevant images that make up a set of "good points". Based on the user-provided images and their scores a new dissimilarity score is calculated for each image. The dissimilarity is calculated by the formula:

$$D_G = \frac{1}{k} * \sum_{i}^{k} d(x_j, g_i)^a$$

Where $\boldsymbol{g}_i$ user selected images (good points)

x_j - Candidate image score

g_i - Score of user marked relevant image

$\boldsymbol{d(x_i, g_i)}$ dissimilarity function implemented as $x_i - g_i$

$\boldsymbol{k}$ – Number of user provided relevant images.

$\boldsymbol{a}$ parameter set to –5

Therefore the dissimilarity score is the sum of the a-th power distance of each image from the good points divided by the total number of good points (user selected images). By distance of each image we mean the sum of the difference between the image's score and of the "good points" (user marked as relevant images).

The results from the modified Chainnet approach are the base for the Falcon method. User must select a set of only relevant images that will be the set of good points. The **a** parameter set to –5 as suggested on [31] even though other values are mentioned as well and the selection of this value is expected to have a major impact on the after-feedback results.

We suspect that this method needs more information from the user than the ones described above (at least an average of more than 10-15 images must be marked while for other methods this amount of relevance is desired but not necessary) since an accurate definition of good point is crucial to a successful relevance feedback.

Depending on the user provided data and also on the value of the **a** parameter we aspect this method the be the most controversial one.
This method is available at: http://hydralix.softnet.tuc.gr/~clauss13/methodos7.asp

4.6.4 Rui – Text

Rui based method for text features only is implemented as explained in theory. This spearhead of this method is the rearrangement of attribute weights based on user information. Users are expected to provide detailed information ranking as many images as possible: highly relevant, relevant, no opinion (don't care), non-relevant and highly non-relevant. The approach consists of various representations (text, visual etc) each with various attributes (e.g. for text caption, alternate description, image name and page title). According to [29], [39] the weights for the attributes that make up each representation (inner weights) are calculated based on mean and deviance values while weights of the representations (text, visual etc) are calculated based on user preferences. In this approach, since we have only one representation (text) no double weights (inner and outer) are necessary. The attributes (caption, name etc) weights are calculated on user feedback as follows:

1. Sort results according to attribute score (caption, name etc)
2. Consider combined score results as RT and each attribute-sorted (e.g. Results sorted by alternate text score) as RT^{j} where *j* caption lexical chain, page title LC, image name LC etc.
3. Threshold results to the first *NRT*.
4. For all images in attribute-sorted score define new weight based on user evaluation as :

   IF $\mathrm{RT}^{j}{}_{i}$ e??a? st? RT

   $\mathrm{W}_i^j = \mathrm{W}_i^j + \mathrm{SCORE}\,i$ IF marked from user.

   $\mathrm{W}_i^j = \mathrm{W}_i^j + 0$ Not marked from user.

   Set $\mathrm{W}_i^j = 0$ if $\mathrm{W}_i^j < 0$

   Where $RT^{j}{}_{i}$ an image in attribute-sorted results.

W_i^j weight for the specific attribute j.

SCORE i User score (if available) for the specific image.

New weights are normalized to Normalize all new weights $W_i^j = \frac{W_i^j}{W^T}$ where

$W^T = \sum W_i^j$ is the sum of all weights.

This is an altered version of the [29], [39] approach based only on text features. An alterative approach to the above weight assignement algorithm would be weight calculation based on mean value and deviance of each attribute's scores. However, we choose to implemet the first approach (the algorithm used for the calculation of outer weights when more then one represetation is present). Similarity score is calculated with the new weights and the result are shown to the user. Some step approach is offered yet after the second one, sometimes even after the first no major changes are observed; therefore the results are quite the same. In this method users are expected to provide detailed information: the more images they evaluate/mark the better the results will be. This method is available at: http://hydralix.softnet.tuc.gr/~clauss13/methodos5.asp

4.6.5 K-Means - Pseudofeedback

As mentioned on chapter 3 users provided information is crucial for the relevant feedback process. Nevertheless sometimes users consider rather boring to mark results as needed by the various relevance feedback methods. Therefore an automatic feedback method, that requires no user interaction at all is provided (pseudo feedback) along with the other feedback methods.

The base results are the modified Chainnet results. All scores are normalized in the [0...1] distance. Based on K-means theory we set 3 centers. We consider the 3 centers approach more suitable to our implementation since there are not too many results for most of the queries and therefore the selection of 5 centers would not be the best choice. If the number of results was large then the 5-center approach would achieve better results. The centers are set to 0.75 for the relevant, 0.5 for the don't care and 0.25 for the irrelevant images where the space [0...1] represents all possible scores for each image. After conducting several experiments with the number of times the algorithm must run, we decided that 2 or 3 times are the better solution. At this number of repetitions maximum clustering is achieved and centers tend to have insignificant changes. We consider the results that belong to the high center (as found after the clustering steps) as the most relevant ones while those to the medium center don't care and finally the results around the "bad" (low) center as the most irrelevant ones. This method's results are (theoretically) not expected to surpass the user-based relevant methods nevertheless is offered as an alternative approach. This method is available at: http://hydralix.softnet.tuc.gr/~clauss13/methodos11.asp

4.6.6 Rui – Text and Visual

Rui based method [29], [39] for text and visual features are implemented exactly as explained in theory. The results on which feedback is processed are those returned from the modified Chainnet method even though it can be seen as a keyword and query search. This method deals primarily with weight rearrangement. As in the Rui text features method, users are expected to mark the results according to the 5-scale score model. Visual feature are extracted only from images marked "highly relevant" from the user; a visual score is calculated for all images based on those. In absence of a "highly relevant" image, the same procedure is applied to images marked as "relevant". If no "highly relevant" or "relevant" images are marked then user is urged to submit a new query since this kind of feedback is beyond the scope of this implementation.

Since there are two representations in this approach we have two kinds of weights: outer (text and visual) and inner (attributes belonging to text and visual). Therefore the similarity score is calculated as the sum of the above:

$$\text{Score}_i = \text{W}_{text}^{outter} * \text{Score}_{text} + \text{W}_{visual}^{outter} * \text{Score}_{visual}$$

$$\text{Score}_i = \sum_0^{nr.attrib} (\text{w}_j^{inner} * \text{Score}_j)$$

Where: *i* text or visual

j number of attributes for visual or text

Inner weights are rearranged based on mean value and deviance of each specific attribute score based on the idea of [39] that the smaller the deviance from the mean value the more uniform this attribute is and therefore more importance (higher weight) should be given to it. Therefore the weights of the various LC that make up the text score or of the various visual features that make up the visual score are calculated automatically based on the principle stated above. Outer weight i.e. importance given to text and visual representations are assigned as follows:

1. Sort results according to text score.
2. Sort results according to visual score.
3. Threshold sorted results to *NRT*.
4. Calculate outer weights as

   IF RT^j_i e??a? st? RT

   $W^j_i = W^j_i + \text{SCORE}\, i$ IF marked from user.

   $W^j_i = W^j_i + 0$ Not marked from user.

5. Normalize outer text and visual weights to sum of 1 $W^j_i = \dfrac{W^j_i}{\sum_{i=0}^{nr-repres} W^j_i}$

Similarity score is calculated for each image with weights adjusted and based on the similarity score formula stated above. Some step approach is offered yet after the second one, sometimes even after the first, no major changes are observed therefore the results are quite the same. Users are expected to provide detailed information: the more images they evaluate/mark the better the results will be. This method is available online at: http://hydralix.softnet.tuc.gr/~clauss13/methodos8.asp

CHAPTER V

Experiments – Evaluation

6.1 Evaluation method-approach

The evaluation is based on human relevance judgments by an independent human referee. For each method, the referee inspected the answers of each query and, for each answer, judged if it is similar to the query or not. The same procedure was required for the relevance feedback methods; here the referee was expected to provide information necessary to the process based on his preferences. This is a highly subjective process. Two or more methods may retrieve the same answer for the same query, but the same answer may not be recognized as similar when it is retrieved by different methods. To be fair, the evaluations must be consistent. To achieve consistency, a query and a retrieved image are considered as similar if they are taken as similar by at least one method. All methods were double checked against inconsistency mistakes and contradictory answers were re-evaluated properly such as to assure exactness and consistency in all queries and methods.

To evaluate the effectiveness of each candidate method, the following quantities are computed:

- **Precision** that is, the percentage of qualifying (relevant) images retrieved with respect to the total number of retrieved images.
- **Recall** that is, the percentage of qualifying images retrieved with respect to the total number of images in the database.
- **Ranking quality** which computes the differences between the rankings of the results obtained by a method and by a human referee.

Certain observations occur here. Regarding **recall**, it is practically impossible to compare every query with each database image. To compute recall, for each query, the answers obtained by all candidate methods are merged and this set is considered to contain the total number of correct answers. This is a sampling method, known as "pooling method", which has been used extensively in the text retrieval literature [46]. This method does not allow for absolute judgments such as "method A misses 10% of the total relevant answers in the database". It provides however, a fair basis for comparisons between methods allowing judgments such as "method A returns 5% fewer correct answers than method B".

Regarding **ranking quality**, the higher the values the better the ranking quality of a method, that is, the method retrieves the qualifying (similar) entries before the non-qualifying ones. Computed as:

1. The answers of the candidate method are evaluated (i.e., each answer is judged as qualifying or not qualifying).
2. Each answer is assigned a "rank" which equals its order within the answer set (i.e., the first answer has rank 1; the second answer has rank 2 and so on).
3. These answers are taken in pairs such that (a) Only pairs with one qualifying and one non-qualifying answer are taken and (b) In each pair, the qualifying entry is first and the non-qualifying entry is second.
4. The relative ranks of the answers in each pair are examined and R_{norm} is computed as follows:

   $$R_{norm} = \frac{1}{2}\left(1 + \frac{S^{+} - S^{-}}{S^{+}_{\max}}\right) \text{ if } S^{+}_{\max} > 0 \;||\; 1 \text{ otherwise.}$$

   Where S^{+} is the number of correctly ranked pairs (i.e., the qualifying entry has higher rank), S^{-} s the number of the erroneously ranked pairs (i.e., the method assigned higher rank to the non-qualifying entry) and $S^{+}_{\max}$ is the total number of ranked pairs.

A precision-recall diagram is presented for each experiment. The horizontal axis in such a diagram corresponds to the measured recall while, the vertical axis corresponds to precision. Each method is represented by a curve. Each query retrieves the best 30 answers (best matches) and each point in a curve is the average over 20 queries. Precision and recall values are computed after each answer (from 1 to 30) and therefore, each curve contains exactly 30 points. The top-left point of a precision/recall curve corresponds to the precision/recall values for the best answer or best match (which has rank 1) while, the bottom right point corresponds to the precision/recall values for the entire answer set.

A method is better than another if it achieves better precision and recall. It is possible for two (or more) precision-recall curves to cross-over. This means that one of the methods performs better for small answers (containing less answers than the number of points up to the cross-section) while, the other performs better for larger answer sets. The method achieving higher precision and recall for large answer sets is considered to be the best method (the typical users retrieve more than 10 or 20 images on the average).

Precision and recall are considered to be the standard criteria for evaluating the accuracy of retrieval methods and they have been used extensively within the context of the text retrieval research [46]. Ranking quality is rarely used. It characterizes the ability of a method to retrieve the correct answers (the answers that the users consider similar to the queries) before the incorrect ones, a desirable feature which, however, cannot show which method is more accurate (i.e., as it is shown in the experiments, the method with the highest ranking quality is not always the most accurate one). In this work, ranking quality is used to distinguish the more effective method among methods with similar precision and recall.

6.2 Experiments

Our system's database consists of nearly 200.000 images and their respective documents, collected as described in the previous chapter. Text and visual features extraction is implemented in C/C++ while the indexing process in Perl. The retrieval part is implemented thoroughly in ASP-Perl. The retrieval process is fairly quick depending on the number of the returned results that need to be analyzed.

6.2.1 Queries

For the evaluation –testing purposes we decided to submit to the system (in its various approaches) the same queries that we submitted to Google in order to create our database. The base approach is by submitting a list of queries-keywords (text description for the desired image). However the relevance feedback process that incorporates both text and visual features can be considered as a special case of text and query-by-example retrieval. We use 20 queries while evaluating the first 30 answers of each query. Queries that return less then 30 answers are automatically filled with negative results i.e. **if** the result was present it would be non-relevant. The list of used queries is as below:

Query	**Query Description**
Q1	Mozilla
Q2	Java
Q3	Music
Q4	Suse logo
Q5	RedHat logo
Q6	IBM logo
Q7	Debian logo
Q8	Audi logo
Q9	Oracle logo
Q10	Juventus logo
Q11	AMD logo
Q12	Intel logo
Q13	VW logo
Q14	Ebay logo
Q15	Hilton logo
Q16	Linux logo
Q17	Gnu logo
Q18	Suzuki logo
Q19	Sun Microsystems logo
Q20	Microsoft windows XP logo

As described in the previous chapter, we limited the number of results to be analyzed to only those image's text descriptions that contain all the query terms (keywords). In this way we keep the initial query semantic meaning, avoid a certain amount of noise and reduce the overall system's response time.

6.2.2 Weight comparison

In this experiment, we compare the two *Chainnet* approaches: the first one with fixed weights as described in [16] and the *modified Chainnet* with all weights set to 1.

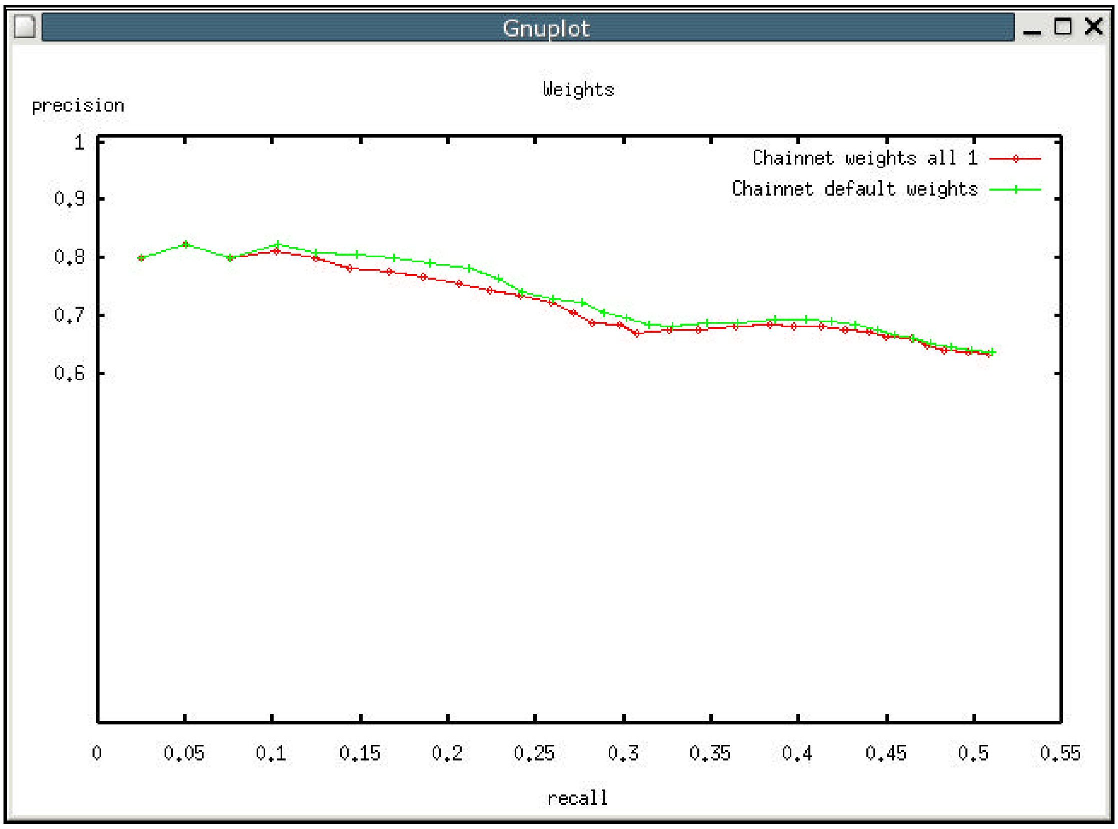


Figure 13: Weights of text attributes

With the green color the original approach while with red the modified (all weights set to 1). As can be plainly observed, the differences between the two approaches are minimal. Therefore it might be wise to say that fixed, a priori weights, does not improve the precision or the recall of the Chainnet approach. All attributes weights can be set to 1 initially (without any loss of precision, recall or overall accuracy-performance) based on the fact that they can be properly rearranged to the real user preferences in the relevance feedback process. However, assuming there will be no relevance process, the improvement of the fixed weights is rather minimal.

6.2.3 Semantic Relevance Feedback

In this experiment we try to compare the performance of the two relevance feedback methods as stated on [16] with the C*hainnet* approaches.

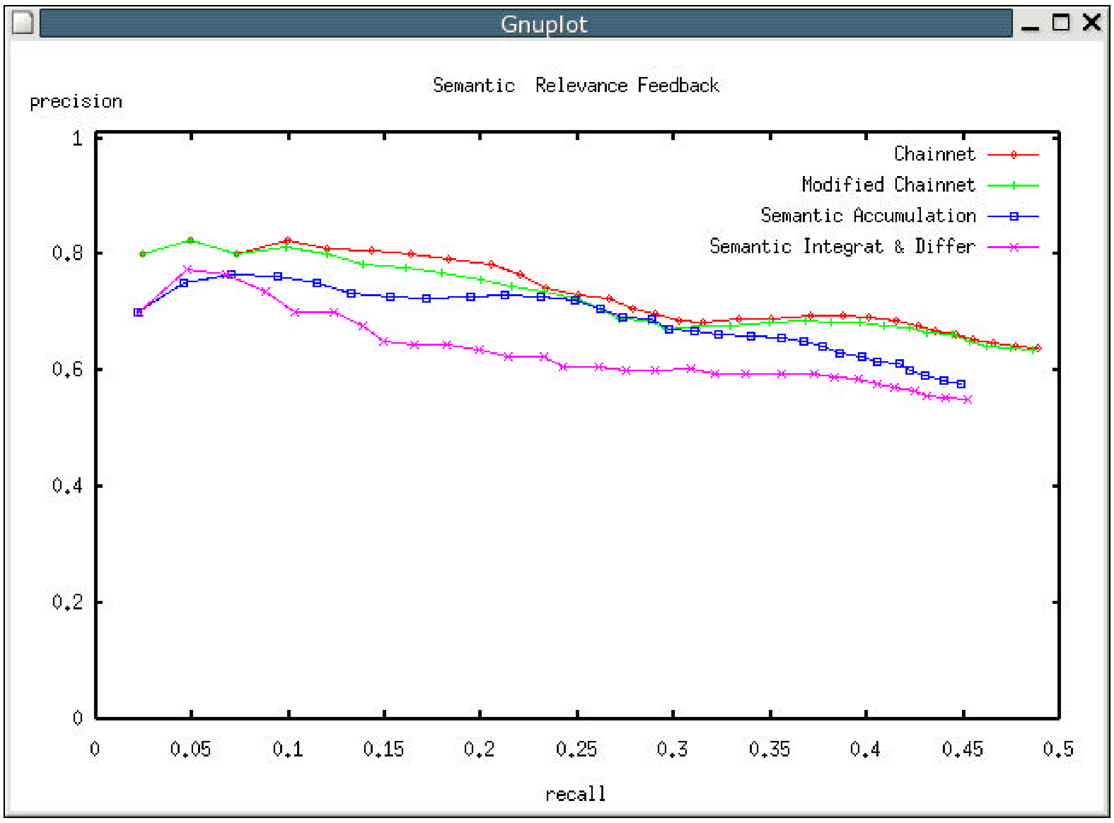


Figure 14: Semantic relevance feedback

With green and red we see the original and the modified Chainnet, with *blue* the *semantic accumulation* method and with *purple* the *semantic integration and differentiation* method. As seen on the diagrams the performance of the above relevance feedback methods is rather low.

The *semantic accumulation* method is fairly close to the Chainnet approach yet it fails to achieve any better results. Our explanation is that query enrichment has an opposite effect : instead of narrowing and defining better terms query criteria it tends to widen up the result, inserting new terms (sometimes beyond the initial query specifications) that have a negative impact on the overall performance.

The *semantic integration and differentiation* has the lowest precision and recall. This can be explained considering the insertion of new terms during the query enrichment process (sometimes introducing unnecessary noise) and the possible removal of some correct (relevant) answers during the differentiation stage of the process.

Also it might be worth mentioning here that, especially, in the *semantic accumulation* method, if the lexical chains of the user selected query are the same as the old query, there is basically no feedback at all. In the *semantic integration and differentiation* method this would lead to a possible elimination of even good results from the initial set, worsening the overall precision as seen on the plot.

6.2.4 Other relevance feedback methods

In this experiment we compare the Chainnet approach to the other relevant methods that are mainly concerned with weight adjustment based on user preferences and/or redefinition of similarity criteria. In this stage of the experiment all 30 images were marked for relevance feedback (even though unnecessary) in order to have equal user intervention-interaction for all methods.

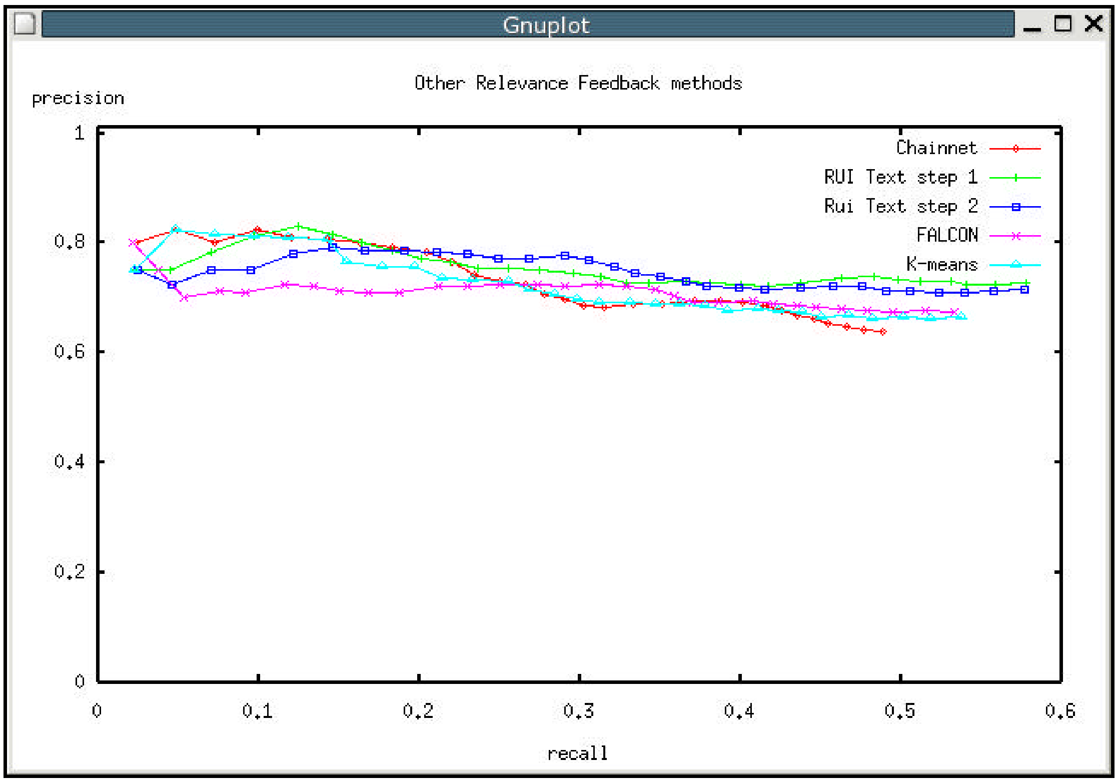


Figure 15: Other relevance feedback

The methods compared are *original Chainnet* approach (red), Rui –text only in it's first step (green), Rui in its second step (blue), Falcon in its first step (purple) and K-means after 2 repetitions. For the K-means method no user feedback is necessary. As seen on the plot the Rui in its both steps is clearly better then all other approaches; due to the highly detailed user information it incorporated. Falcon is somewhat lower then the original approach probably due to the fact that either user provided information was too little either some of the user examples provided were not consistent as "good points". However, Falcon achieves greater recall then the original method. K-means is quite the same as the original method with slight improvement in the overall recall achieved. K-Means does not require any information from the user and probably that is the weakness of it: it cannot compare with user-based methods such as e.g. Rui's approach. Nevertheless, even though the results or pseudo relevance processes were expected to be low in comparison with other methods, k-means achieves somehow good overall statistics.

6.2.5 Relevance feedback with visual features

In this experiment we compare the original Chainnet method, the best method of only text relevance feedback (Rui –text only in this case) and Rui's feedback that incorporated both text and visual features. As in the previous experiment all 30 images were marked from the referee for equality reasons. In all previous work and literature has been shown that systems that make use of both visual and text features achieve higher precision and recall then systems using only one of the above.

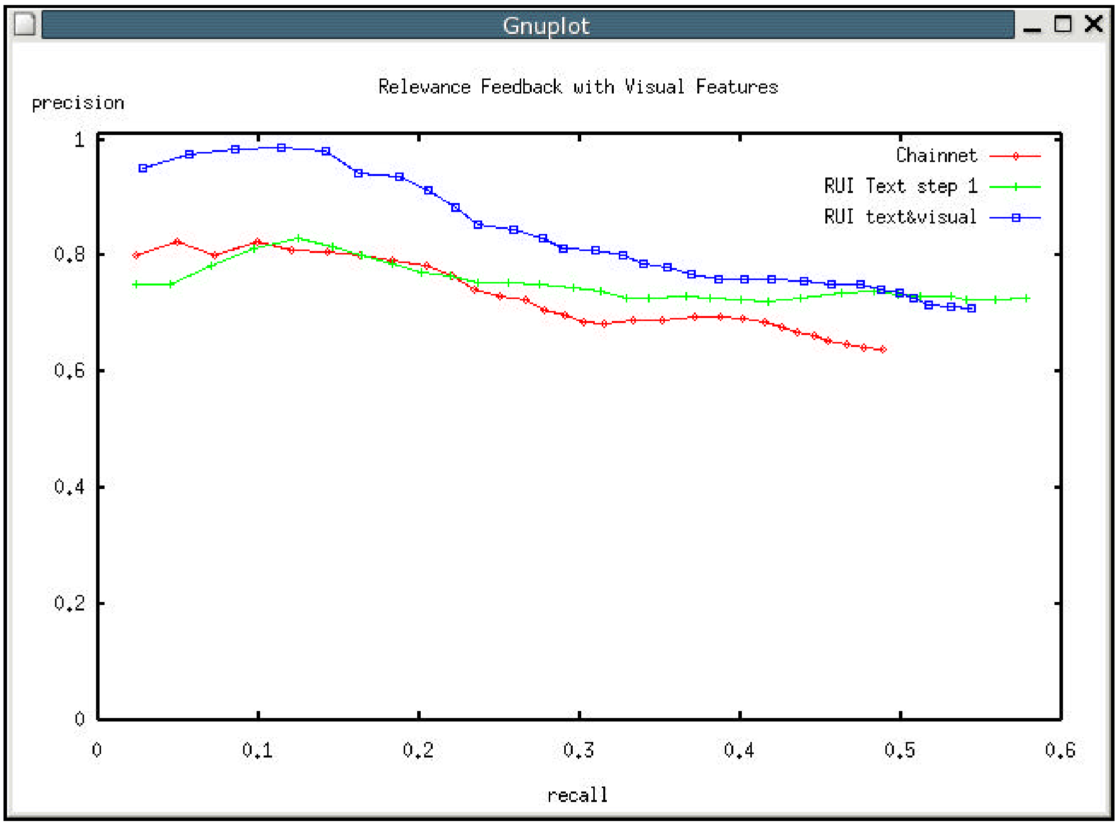


Figure 16: Visual and text feedback

As seen in the above plot the method that uses both visual and text features achieves very high precision and recall compared to the other methods. That can be explained to the fact Rui's approach makes the best use of visual and text features. Considering the results achieved from the text-only version of Rui, we must conclude that visual features make up big part of its success. Also the adjustment of weights based on user information is surely to have very good results if user information is accurate and detailed and the algorithm is properly implemented. Nevertheless, this kind of feedback has been shown to achieve high precision and recall even without detailed user information.

Also its worth mentioning here that this kind of feedback will have best results for images with strict definition of visual content meaning that it will give best results if all relevant images marked are more of less the same (logo queries being such), nevertheless, it will provide good results even for images with different visual features (non logo queries are such)

6.3 Ranking quality

As mentioned on the previous sector, the ranking quality characterizes the ability of a method to retrieve the correct answers before the incorrect ones, a desiderable feature which however cannot show which method is more accurate (i.e., as it is shown in the experiments, the method with the highest ranking quality is not always the most accurate one.

Method Name	**Ranking quality**
Original Chainnet	0.591143
Modified Chainnet	0.608792
Semantic Accumulation	0.706883
Semantic Integration & Differentiation	0.723659
Rui (text only) step 1	0.521293
Rui (text only) step 2	0.509414
Falcon	0.701741
Rui (Text and Visual)	0.736551
K-Means	0.597237

As we see, the feedback approach that incorporated text and visual features has the highest ranking quality. Nevertheless, while Rui's feedback approach has better precision – recall that the two semantic feedbacks it has a lower ranking quality value. This can be partly explained to the fact that probably new images enter in Rui's feedback approach. Also, the ranking quality value might work better with smaller elements and not with such a large set of results or database.

6.4 Conclusions

In this thesis we implemented a fairly new idea of image retrieval based on text description and visual features. We see that image retrieval by text description (considering that extraction and user keyword selection are done properly) achieves high precision and fairly high recall. The size of or database and the plethora of images in it guarantee that the results are very good and the retrieval speed of this method is rather low making it quite interesting for further research and implementations.

The weight problem and the key issues that arise from it are dealt with a priori, all-as-one, initialization that (as shown) provides the same results as the fixed weight approach. Nevertheless, the users are urged to proceed to relevance feedback since by doing so they will improve the system's judgment criteria and improve the overall results.

The various implemented relevant methods that imply weight adjustment achieve high precision recall compared to the "query redefinition" methods (semantic accumulation and semantic integration and differentiation) for reasons explained and discussed above. Observation here is the fact that the weight problem is solved in the best way by incorporating user preferences and adjusting the results properly based on them.

At last, as mentioned in theory it is shown that the incorporation of text, visual features and user preferences is surely to provide very high precision and recall in the first steps of the relevance feedback process. Rui's algorithm, as discussed in theory, provides a very efficient and user friendly way to improve the results.

References

[1] Cleverdon, C.W., "Progress in documentation. Evaluation of information retrieval systems", Journal of Documentation, 26, 55-67, (1970).

[2] Sparck Jones, K., Automatic Keyword Classification for Information Retrieval, Butterworths, London (1971).

[3] Sparck Jones, K., "Some thoughts on classification for retrieval", Journal of Documentation, 26, 89-101 (1970).

[4] Lancaster, F.W., "Information Retrieval Systems: Characteristics, Testing and Eva luation", Wiley, NewYork (1968).

[5] Salton, G., Automatic Information Organization and Retrieval, McGraw-Hill, (1968).

[6] Salton, G., Yang, C.S. and Yu, C.T., "A theory of term importance in automatic text analysis", Journal of the American Society fo r Information Science, 26, 33-44 (1975).

[7] Salton G., McGill M., "Introduction to modern Information Retrieval" McGraw-Hill, 1983.

[8] G. Salton, Automatic Text Processing, Addison Wesley, 1989.

[9] http://pi0959.kub.nl:2080/Paai/Onderw/Smart/smart.html

[10] Salton G, "The SMART retrieval system – Experiments in automatic document processing" pages 313-323. Prentice Hall, 1971.

[11] Y. Rui, T.S.Huang, S. Mehrotra, Content based image retrieval with relevance feedback.

[12] W. Niblak et al., "The QBIC project: Querying images by content using color, texture, and shape", in Proc. of SPIE, vol. 1908, 173-182, 1993.

[13] M. Flickner et al., "Query by Image and Video Content: The QBIC System", IEEE Computer, 28, 9, 23-31, 1995.

[14] R. Bach et al., "The Virage Image Search Engine: An open framework for image management", in Proc. of SPIE, vol. 2670, 76-87, 1996.

[15] J.R. Smith, S.F. Chang, "VisualSEEK: a fully automated content-based image query system", Proc. of ACM Multimedia'96, 1996.

[16] Heng T.Sh., Beng C.O., Kian-Lee T., "Giving meanings to WWW images".

[17] Google Inc, http://images.google.com/

[18] AltaVista Company, http://www.altavista.com/

[19] Lempel R., Soffer A., “PicASHOW: Pictorial authority search by hyperlinks on the Web”, ACM transactions on Information Retrieval, 2002.

[21] Scour Inc., http://www.scour.com/

[22] Ditto, Visual search engine, http://www.ditto.com/

[23] Getty Images Inc., http://www.getty-images.com/

[24] Corbis Inc., http://www.corbis.com/

[25] Y. AlpAslandogan, Clement T. Yu,"Multiple Evidence Combination in Image Retrieval: Diogenes Searches for People on the Web. ACM SIGIR 2000.

[26] Y. AlpAslandogan, Clement T. Yu, “Diogenes: A Web search agent for person images”, ACM Multimedia, 2000.

[27] Sclaroff S., Taycher L., La Cascia M., “ImageRover: A content based image browser for the WW Web” IEEE Workshop Content based Access of image and video libraries, 1997.

[28] Swain M., Frankel C., Athitsos B., “WeebSeer: An image search engine for the WW Web”, University of Chicago, 1996.

[29] Y. Rui, T.S.Huang, S. Mehrotra, “Content based image retrieval with relevance feedback in MARS”, Proc. of IEEE ICIP'97, 1997.

[31] Wu L., Faloutsos C., Sycara K., Payne T., “Falcon: Feedback adaptive loop for content-based retrieval”, VLDB conference Cairo, 2000.

[32] Brin S., Page L., “The anatomy of a large scale hypertextual Web search engine” New York, 1998, pages 107-117.

[33] Page L. et al “The PageRank citation ranking: bringing order to the Web”, Stanford digital library, 1998.

[34] Kleinberg J., “Authoritative sources in a hyperlink enviroment”, ACM press, 1998, pages 668-677.

[35] Fass A., Adar E., Bier E., “PicturePiper: using a reconfigurable pipeline to find images on the Web”, ACM Press, 2000.

[36] Hull D., Grefenstette G., “A detailed analysis of English stemming algorithms”, Xerox research center, Palo Alto, 1996.

[37] http://www.tartarus.org/~martin/PorterStemmer/index.html

[38] http://www.pasc.org

[39] Y. Rui, T.S.Huang, S. Mehrotra, Ortega M., "Relevance feedback: a power tool for interactive content-based image retrieval", Uni. Of Illinois, Chicago.

[40] Petrakis, G.M. E., "Design and Evaluation of Spatial Similarity Approaches for Image Retrieval", Image and Vision Computing, Num.1, Volume 20, pp. 59-76, January 2002

[41] http://larbin.sourceforge.net/index-eng.html

[42] Epimenides Voutsakis, Dissertation thesis, "Image retrieval on the World Wide Web"

[43] Vassilis Hatzidiakos, Dissertation thesis, "Automatic logo and trademark extraction from large websites".

[44] http://www.sleepycat.com/

[45] Yoshiharu I., Ravishanhar S., Faloutsos C., "MindReader: Quering databases through multiple examples"