Contextualized access to distributed and heterogeneous multimedia data sources

Christian Vilsmaier

N° d'ordre 2014ISAL0094 Année 2014

Institut National des Sciences Appliquées de Lyon Université de Passau

École doctorale InfoMaths : Informatique et Mathématiques (EDA 512)

CONTEXTUALIZED ACCESS TO DISTRIBUTED AND HETEROGENEOUS MULTIMEDIA DATA SOURCES

Thesis

Presented and publicly defended in the 26th of September 2014

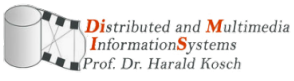
To obtain

The degree of Doctor in Computer Science

by
Christian Vilsmaier

Committee members

Rapporteur	Günther SPECHT	Professor (University of Innsbruck)
	Vincent CHARVILLAT	Professor (Institut National Polytechnique de Toulouse)
Examiner	Oliver AMFT	Professor (University of Passau)
	Elöd EGYED-ZSIGMOND	Maître de Conférence (LIRIS-INSA Lyon)
	Jacques SAVOY	Professor (Université de Neuchâtel)
	Catherine BERRUT	Professor (Université Joseph Fourier Grenoble)
Supervisors:	Lionel BRUNIE	Professor (LIRIS-INSA Lyon)
	Harald KOSCH	Professor (University of Passau)
	Mario DÖLLER	Professor (University of Applied Sciences Kufstein)



Laboratoire d'InfoRmatique en Images et Systèmes d'information - UMR 5205

Lehrstuhl für Verteilte Informationssysteme

Abstract

Making multimedia data available online becomes less expensive and more convenient on a daily basis. This development promotes web phenomenons such as Facebook, Twitter, and Flickr. These phenomena and their increased acceptance in society in turn leads to a multiplication of the amount of available images online. This vast amount of, frequently public and therefore searchable, images already exceeds the zettabyte bound. Executing a similarity search on the magnitude of images that are publicly available in the Internet, and receiving a top quality result is a challenge that the scientific community has recently attempted to rise to. One approach to cope with this problem assumes the use of distributed heterogeneous Content Based Image Retrieval system (CBIRs). Following from this anticipation, the problems that emerge from a distributed query scenario must be dealt with. For example the involved CBIRs' usage of distinct metadata formats for describing their content, as well as their unequal technical and structural information. An addition issue is the individual metrics that are used by the CBIRs to calculate the similarity between pictures, as well as their specific way of being combined. Overall, receiving good results in this environment is a very labor intensive task which has been scientifically but not yet comprehensively explored.

The problem primarily addressed in this work is the collection of pictures from CBIRs, that are similar to a given picture, as a response to a distributed multimedia query. The main contribution of this thesis is the construction of a network of Content Based Image Retrieval systems that are able to extract and exploit the information about an input image's semantic concept. This so called semantic CBIRn is mainly composed of CBIRs that are configured by the semantic CBIRn itself. Complementarily, there is a possibility that allows the integration of specialized external sources. The semantic CBIRn is able to collect and merge results of all of these attached CBIRs. In order to be able to integrate external sources that are willing to join the network, but are not willing to disclose their configuration, an algorithm was developed that approximates these configurations. By categorizing existing - as well as external - CBIRs and analyzing incoming queries, image queries are exclusively forwarded to the most suitable CBIRs. In this way, images that are not of any use for the user can be omitted beforehand. The hereafter returned images are rendered comparable in order to be able to merge them to one single result list of images, that are similar to the input image. The feasibility of the approach and the hereby obtained improvement of the search process is demonstrated by a prototypical implementation and its evaluation using classified images of *ImageNet*. Using this prototypical implementation an augmentation of the number of returned images that are of the same semantic concept as the input images is achieved by a factor of 4.75 with respect to a predefined non-semantic CBIRn.

Keywords: Multimedia Databases, Content Based Image Retrieval, CBIRs, Data Fusion, Data Merging, Image Retrieval

Zusammenfassung

Multimedia Daten im Internet bereitzustellen wird von Tag zu Tag kostengünstiger und bequemer. Diese Entwicklung fördert Netz-Phänomene wie Facebook, Twitter und Flickr. Dies wiederum führt zu einer Multiplikation der im Internet verfügbaren Bilder. Diese Menge von häufig öffentlichen und darum suchbaren Bildern übersteigt schon heute die Zettabyte-Grenze. Eine Ähnlichkeits-Suche auf diese Masse von Bildern, die öffentlich im Internet verfügbar sind, auszuführen und ein hochwertiges Resultat zu erzielen ist eine Herausforderung der sich die wissenschaftliche Gemeinschaft heutzutage stellt. Um beispielsweise dieses Problem mit Inhaltsbasierten Bilder Suchsystemen (CBIRs) zu lösen müssen Probleme, die durch ein verteiltes Anfrageszenario entstehen, behandelt werden. Damit sind u.a. die Verwendung von unterschiedlichen Ausprägungen von Metadaten Formaten involvierter CBIRs um ihre Bilder zu beschreiben, so wie ihre ungleiche technischen und strukturellen Informationen gemeint. Oder die individuellen Metriken, die von den CBIRs verwendet werden um die Ähnlichkeit der Bilder zu berechnen, wie auch ihre spezielle Art diese zu kombinieren. Im Großen und Ganzen besteht eine sehr arbeitsintensive Aufgabe darin, unter diesen Umständen gute Resultate zu erzielen. Diese Aufgabe wurde von wissenschaftlicher Seite noch nicht umfassend erforscht.

Das Problem das hauptsächlich in dieser Arbeit thematisiert wird, ist das Sammeln von Bildern die einem vorgegebenen Bild ähnlich sind, als Antwort auf eine verteilte Multimedia Anfrage. Der Hauptbeitrag dieser Dissertation ist der Aufbau eines Netzwerks von CBIRs, das die semantischen Konzepte der Anfragebilder nutzen kann, genannt semantisches CBIRn. Das semantische CBIRn ist in der Lage, Ergebnisse von CBIRs zu sammeln und zusammenzufügen, die vom semantischen CBIRn selbst angelegt wurden, und CBIRs, die von spezialisierten externen Quellen integriert wurden. Um in der Lage zu sein, externe Quellen zu integrieren die dem Netzwerk beitreten möchten, jedoch nicht ihre Konfigurationen preis geben möchten, wurde ein Algorithmus entwickelt der diese Konfiguration berechnet. Durch die Kategorisierung der angelegten - wie auch der externen - CBIRs und durch die Analyse einkommender Anfragen, werden inhaltsbasierte Suchanfragen ausschließlich zu den geeignetsten CBIRs weitergeleitet. Die anschließend zurückgegebenen Bilder werden vergleichbar gemacht um imstande zu sein, sie zu fusionieren und eine Ergebnisliste zurückzugeben, die dem Eingabebild ähnlich sind. Die Machbarkeit des Ansatzes und die gewonnenen Verbesserungen des Suchprozesses wird durch eine prototypische Umsetzung und einer Auswertung von Tests mit klassifizierten Bilder von *ImageNet* durchgeführt. Hierbei wurde im Vergleich mit einer vordefinierten nicht-semantischen CBIRn eine Steigerung der Anzahl der rückgegebenen Bilder die dem selben semantischen Konzepts wie das das Eingabebild entsprechen, um den Faktor 4.75, erreicht.

Keywords: Multimedia Datenbanken, Inhaltsbasierte Bilder Suche, Inhaltsbasierte Bilder Suchsysteme, Daten Fusion, Daten Verschmelzung, Bilder Suche

Résumé

Rendre les données multimédias disponibles en ligne devient moins cher et plus pratique sur une base quotidienne, par exemple par les utilisateurs eux-mêmes. Des phénomènes du Web comme Facebook, Twitter et Flickr bénéficient de cette évolution. Ces phénomènes et leur acceptation accrue conduisent à une multiplication du nombre d'images disponibles en ligne. La taille cumulée de ces images souvent publiques et donc consultables, est de l'ordre de plusieurs zettaoctets. L'exécution d'une requête de similarité sur de tels volumes est un défi que la communauté scientifique commence à cibler. Une approche envisagée pour faire face à ce problème propose d'utiliser un système distribué et hétérogène de recherche d'images basé sur leur contenu (CBIRs). De nombreux problèmes émergent d'un tel scénario. Un exemple est l'utilisation de formats de métadonnées distincts pour décrire le contenu des images; un autre exemple est l'information technique et structurelle inégale. Les métriques individuelles qui sont utilisées par les CBIRs pour calculer la similarité entre les images constituent un autre exemple. Le calcul de bons résultats dans ce contexte s'avère ainsi une tâche très laborieuse qui n'est pas encore scientifiquement résolue.

Le problème principalement abordé dans cette thèse est la recherche de photos de CBIRs similaires à une image donnée comme réponse à une requête multimédia distribuée. La contribution principale de cette thèse est la construction d'un réseau de CBIRs sensible à la sémantique des contenus (CBIRn). Ce CBIRn sémantique est capable de collecter et fusionner les résultats issus de sources externes spécialisées. Afin d'être en mesure d'intégrer de telles sources extérieures, prêtes à rejoindre le réseau, mais pas à divulguer leur configuration, un algorithme a été développé capable d'estimer la configuration d'un CBIRS. En classant les CBIRs et en analysant les requêtes entrantes, les requêtes d'image sont exclusivement transmises aux CBIRs les plus appropriés. De cette façon, les images sans intérêt pour l'utilisateur peuvent être omises à l'avance. Les images retournées cells sont considérées comme similaires par rapport à l'image donnée pour la requête. La faisabilité de l'approche et l'amélioration obtenue par le processus de recherche sont démontrées par un développement prototypique et son évaluation utilisant des images d'*ImageNet*. Le nombre d'images pertinentes renvoyées par l'approche de cette thèse en réponse à une requête image est supérieur d'un facteur 4.75 par rapport au résultat obtenu par un réseau de CBIRs predéfini.

Keywords: Bases de données multimédias, Recherche d'Images Contexte Sensitive, Fusion de données, Unification de données, Recherche d'Images

Contents

List of Figures

List of Tables

1. Introduction

1.1. Motivation

These days, if a user is involved in a process that requires images and s/he has no adequate images to hand - e.g. a student that needs a superior image of a chromosome to decorate the cover page of an assignment or an editor-in-chief that is not provided adequate images to illustrate their articles perfectly - what would they do? They would most likely take advantage of a search engine that operates on keywords in order to acquire the image. Which means for the student s/he would launch Google's[1], Bing's[2] or Baidu's[3] keyword based image search and browse for a license free replacement for the chromosome image that s/he wanted to use. The editor-in-chief would log in to Fotolia[4] and scan for fitting images. Even though both are in possession of images that nearly meet their requirements, both of them prefer to start a new keyword-based search instead of proceeding with a search using a content-based method. What are the reasons for this?

The results of current Content Based Image Retrieval systems (CBIRs) are quite often unsatisfying [56, 19]. The functionality that such CBIRs should fulfil is defined as follows: The CBIRs are given an input image, compare this to images that they are linked to or have stored and return a, so called result set, presenting the most similar images. Results are ofen unsatisfactory due to the gap between the CBIRs' extracted image semantics and the user search intention. This problem is known in the literature as the problem of the "semantic gap" [127] and has been investigated intensively [30, 33, 50, 148, 159, 37, 51].

Keyword driven search engines derive the content of a multimedia file by analyzing the file's name, metadata as well as its surrounding text [60]. This was a good strategy and worked well in Web 1.0. Newspapers accompanied their articles with topic related pictures, commercial sites were putting short explanatory texts next to their merchandise photos, and people that had websites used images to inform the reader about the multimedia content they were seeing. Web 2.0 breaks with this tradition.

By combining users from different cultures and social classes, it becomes impossible to win information from the environment a piece of content is generated from. An example to mention in this context is the varying naming behavior of the users in Flickr. While user A names a photo "Natives are having a traditional procession", user B assigns the name "My family thanking for the good harvest" to a very similar photo he uploaded independently from user A. In addition, user C thinks an appropriate name for a similar picture would be "Amazon holiday Picture 128/150". As shown in the precedent example, the naming process of a picture is highly relative to the user's cultural and social origin as well as to the circumstances in which the photo was taken. For the same reasons it is often difficult to assign significant keywords to the photos.

[1] http://www.google.com
[2] http://www.bing.com
[3] http://www.baidu.com
[4] http://www.fotolia.com

Nevertheless, to find these Web 2.0 photographs, newer search engines are implementing Content Based Image Retrieval (CBIR). The authors of [64, 139, 125] outlined research attempts as well as recent commercial efforts concerning CBIRs. In contrast to current keyword driven search engines, they deliver, independent from the textual context, visually similar pictures to the search picture [64]. These works however still suffer from the semantic gap. Indeed when a search engine tries to specify a universal solution that allows users to search all image domains simultaneously, the results are not satisfying. However by dividing the whole image domain into multiple narrow and broad search domains [126] and by specializing CBIRs to search these narrow domains [68, 102] the problem of a universal search strategy is bypassed and very good results are obtained.

Therefore, to bring heterogeneous specialized CBIRs to work together, this thesis proposes the concept of a semantic *Content Based Image Retrieval network* (CBIRn). This semantic CBIRn will not solve the problem of the semantic gap, but will allow the user to search the mass of images that are available online more efficiently. This is possible thanks to the integration and generation of CBIRs that are each specialized on one specific semantic concept. The limitation to one specific concept each is necessary due to the application of state of the art concept detection algorithms [21, 42, 144]. These concept detection algorithms are specialized on single concepts and are applied just in a concept detection step and in a subsequent step to forward the queries to the appropriate CBIRs.

To render an integration of external CBIRs possible, their configuration must be analyzed [143] and stored for a use in the query process. An integration step is necessary as the configuration of a CBIRs - normally consisting of a selection of features, metrics, and weights - might differ for each CBIRs in oder to match the challenge of ranking domain specific images. Among other things, thanks to the additional information obtained from the integration step, it is subsequently possible to forward the queries inside the *semantic CBIRn* in a way that privileges the CBIRs whose semantic concept is more relevant for the search [142]. Hence discrepancies of the returned results list, that are caused by the different configurations of the involved CBIRs, are straightened out and their returned results are rendered comparable [141].

It is the motivation of this thesis to lay the foundations for a *Content Based Image Retrieval network* (*CBIRn*) that enables a standard Internet user to search for similar images, based on an input image's semantic concepts.

1.2. Research Questions and Main Contributions

Nowadays CBIR research appears to concentrate on three scientific areas: new features, CBIRn, and the semantic gap. A lot of effort has been invested to make features perform better in terms of storage usage, extraction speed, and comparison performance [75]. For decades now many efforts have been undertaken to narrow down the semantic gap with some success [40]. The approach proposed in this thesis takes advantage of all of these research fields. Recently CBIRn strategies have been enhanced by developing new protocols, distributing content in a more advantageous way [122], and applying different design paradigms e.g. relevance feedback [158]. In order to realize the approach explained in the precedent section, the following research questions will be addressed in detail:

- **How can a semantic CBIRn be modeled and implemented?** - The poor performance of modern CBIRs and CBIRn is caused to a large extent by the semantic gap. In order to diminish this semantic gap the approach of this thesis specifies the fundamentals of a

semantic CBIRn that is capable of detecting the input image's semantic concept and taking advantage of this knowledge. This objective is achieved by the following measures: the idea of the standard CBIRn is rethought and virtually sub-divided into sub-networks that, for the previously discussed reasons, are each specialized on one single semantic concept. By building up this so called *semantic CBIRn*, queries for input images that the network receives can be allocated to the suitable predefined semantic concepts or CBIR subnetwork in order to improve the results that are returned to the user.

- **How can the configuration of a CBIRs, which should be incorporated in a *semantic CBIRn*, be detected?** - In order to make the construction of a *semantic CBIRn* possible, two divergent situations have to be taken into consideration. On the one hand CBIRs can be configured by the *semantic CBIRn* itself, and therefore the configuration - the used features, metrics, and weights - of the CBIRs are known to the CBIRn. On the other hand CBIRs can be instantiated by an anonymous instance. In this case the configuration of the CBIRs has to be reverse engineered. Therefore, a process is developed in order to cope with the task of the identification of the configuration of an external CBIRs. This process needs the CBIRs to enroll themselves in the *semantic CBIRn* in order to be considered as a source for a multimedia query. In order to get hold of the configurations of external CBIRs that request to join the *semantic CBIRn* - e.g. image databases of art museums or real estate agencies - each external CBIRs is asked for a small selection of images. By analyzing the returned results, an approximation of the configuration of the external CBIRs is reverse engineered and the external CBIRs is integrated in the *semantic CBIRn*.

- **How can inquiries in a *semantic CBIRn* be exclusively forwarded to relevant CBIRs?** - There is a variety of CBIRs available these days, but none of the CBIRs is considered globally the very best. A CBIRs' performance depends highly on the domain of application - e.g. face identification systems have a good performance when it comes to identifying faces, fingerprint comparison systems have a good performance when comparing fingerprints, but both of them exhibit poor performance when it comes to distinguishing between two different kinds of flowers. In order to obtain the best tailored performance possible from the *semantic CBIRn*, an analysis of the input images combined with a selective usage of CBIRs is applied. The selection of CBIRs can be achieved by using the analyzed information about the primary semantic concept of the query image. Secondary semantic concepts will be considered in future work. This way the CBIR systems that are most suitable to provide a top quality response to the current query are chosen. The systems that appear not to promise good performance for the current query are not considered as relevant data sources. This way the query is forwarded solely to CBIRs that are capable of handling the input image.

- **How can the results of similarity searches of individual CBIRs be merged in a *semantic CBIRn*?** - By using a multiplicity of CBIRs, the number of individual results increases. Unfortunately, the scores of the results are in general incomparable. In order to deal with this problem, an approach to compute an integrated results list is developed. In this approach a normalization of the similarity scores and subsequently a reordering of the resulting items takes place. It is seen as important to keep the computation expenses at a low level and obtain a useful result, as the developed algorithm is intended to render very big amounts of data content based searchable. In a first step the algorithm normalizes the returned heterogeneous scores. Subsequently a strategy is applied to calculate a new score using the scores of items that show up in multiple result lists and by performing a

regression analysis. Accordingly, using a global search measure, a filter is used to carry out a re-ranking on the global result set.

1.3. Outline

The remainder of this thesis is organized as follows. Chapter 2 focuses on a state of the art review of the topic. Chapters 3 to 5, describes the approach that was developed in this thesis. As this thesis' approach consists of three sub-approaches, each of them is discussed in separate chapters. Eventually chapter 6 concludes this thesis. A more detailed overview of the content of each chapter follows:

Chapter 2 - The fundamentals of information, web, image, and content based image retrieval is introduced in this chapter. Individual techniques and their success in certain retrieval areas are presented in order to sensibilize the many different ways that image retrieval can be performed. The mode of operation of Content Based Image Retrieval systems (CBIRs) is explained in detail in order to make later modifications more comprehensible. The definition of the semantic gap and problems that derive from it, as well as solutions to deal with it, are explored. The chapter is concluded by using examples of usage of CBIR in current applications.

Several systems implementing content-based image retrieval is introduced in this chapter. Their different modes of operation are outlined in order to explain the reasons for their differing performances. Furthermore the issue of the subjectivity of image tests, as well as examples of test sets, are highlighted. Fundamentals of Distributed Image Retrieval are introduced in this chapter. Later on the standards that enable the implementation of a CBIRn, its history, and its mode of operation are discussed in this chapter. Eventually recent progresses in concept detection and ranked based late fusions conclude the chapter.

Chapter 3 - Chapter three spotlights the ***Me***rging of ***R***eturned ***R***esult ***Se***ts (***MeRRSe***). After a brief introduction of the chapter, the methodology of the developed approach - that merges the result sets of distributed image queries - is discussed. The detailed approach outline is followed by an evaluation section which concludes this chapter.

Chapter 4 - The beginning of the fourth chapter introduces the problem of the unknown configuration of an external CBIRs when it intends to join a network. In the accompanying methodology the proposed method ***Ge***t the ***C***BIRs ***Co***nfiguration (***GeCCo***) is detailed. This methodology is then elaborated in the approach. A detailed evaluation of the approach concludes chapter four.

Chapter 5 - The fifth chapter addresses the issue of the construction of a *semantic CBIRn*. At its conception, it discusses why the necessity of such a network is given. The methodology of the construction of such a network and the ***Que***ry ***Di***stribution approach (***QueDi***) is spotlighted in the following section. The subsequent approach spotlights the theoretical details of the network. In the following section the ***QueDi*** is evaluated. The chapter is concluded by an evaluation of this thesis' approach, including an explanation of the necessary adaptions to combine ***MeRRSe***, ***GeCCo***, and ***QueDi***.

Chapter 6 - In this chapter a summary of this thesis' realizations is given. In the first half of this chapter, the solution for the research questions that were established in this chapter

are discussed. The second half of this chapter concentrates on the possible future work that might follow and herewith concludes this thesis.

2. State of the Art

In the beginning of this thesis a basic understanding of the emerging topics is established. This will be achieved by giving an overview of the field of information retrieval while concentrating mostly on the Content Based Image Retrieval variant. This overview starts with a short introduction of information retrieval in section 2.1, followed by an introduction of the techniques and strategies that are used in web retrieval in section 2.2. Section 2.3 and 2.4 contribute to this chapter by discussing the state of the art in Web Image Retrieval and Content based image retrieval. Eventually section 2.5 completes the overview of this thesis' relevant technologies by a discussion of Distributed Retrieval in CBIR.

2.1. Fundamentals of Information Retrieval

The following section will deliver an insight into the historical development of the most important terms around and the standard topology of a system that is specified for information retrieval. The section will be concluded by a discussion of the various fields of application for information retrieval.

2.1.1. History

Even in the earlier ages of mankind, approximately 5000 years ago, people were just as eager to note down information. Those days the sumerians surprisingly already had a primitive version of a data storage system. They chose specific places to store clay tablets with cuneiform inscription. Already, the Sumerians were conscious of the fact that proper organization and access to the clay tablets was critical for a productive use of the written information. Therefore they developed special classifications to identify every tablet and its contents [124].

More recentlt, in the early 19^{th} and 20^{th} century, the Jacquard Loom - the electromechanical data tabulator and the statistical machine - was invented. However, the idea of using computers to search for relevant pieces of information was not popularized until the article As We May Think by Vannevar Bush in 1945 [17]. Eventually, in 1950, Calvin N. Moore wrote in the first Proceedings of the International Congress of Mathematicians the following lines [91]:

> *The problem of directing a user to stored information, some of which may be unknown to him, is the problem of information retrieval[...] The technical goal is finding, in minimum time, those messages of interest to the receiver, where the receiver has available a selective device with a finite digital scanning rate.*

By doing so, without knowing, he gave birth to a new field in computer science. In the following years important steps towards the computerization of information retreival were conducted. In 1951, Philip R. Bagley approached James Perry, an MIT faculty member, with the idea of carrying out some improvements to the Vannevar Bush's Rapid Selector [17] as his Master's degree thesis topic. Perry knew that the world's largest and most powerful computer at that time (the IBM Whirlwind) was available on the MIT campus for their use, and that

Bagley was experienced in programming that machine. Bagley took the suggestion, executed that work, wrote his very thoughtful and analytic thesis on that topic in 1951, but did not implement any such system [5]. Bagley's conclusion in 1951 was that computerized searching of large bibliographic files was infeasible at that time because of the current technical limitations.

Subsequently in 1955 Allen Kent published a first article on, amongst other things, precision and recall measures [59]. Eventually, in 1958 the first international conference The International Conference on Scientific Information in Washington DC included consideration of Information Retrieval systems as a solution to the identified problems [1]. Thus, in the late 1950s and the 1960s, the first important books and articles appeared treating the subject of Information Retrieval [76, 83, 106, 58, 8, 71, 115, 120, 119], as well as the first Information Retrieval systems (e.g. SMART [114], MEDLAR [109], and Intrex [98]) were specified. Further developments in the 1970s and 1980s brought up web, image, and content based image retrieval. These methods will be discussed in the succeeding chapters.

2.1.2. Definitions

This section should give a general view of the research field of information retrieval. In order to introduce the nuances of information rietrieval this subsection defines its most important terminology:

Precision in information retrieval is defined as the fraction of the number of the intersection of relevant and retrieved documents over the number of retrieved documents.

$$precision = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{retrieved\ documents\}|} \tag{2.1}$$

When all retrieved documents are relevant this would result in a precision of 1. In contrast when no relevant document is retrieved this would result in a precision of 0. Equation 2.1 was mentioned for the first time in a scientific article in [59].

Recall in information retrieval is defined as the fraction of the number of the intersection of relevant and retrieved documents over the number of relevant documents.

$$recall = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{relevant\ documents\}|} \tag{2.2}$$

When all relevant documents are retrieved this would result in a precision of 1. In contrast when no relevant document are retrieved this would result in a precision of 0. Equation 2.2 was mentioned for the first time in a scientific article in [59].

2.1.3. Theory of Information Retrieval Models

In information retrieval, documents are typically transformed into a suitable representation, as well as differing strategies are used to effectively retrieve relevant documents. Each retrieval strategy incorporates a specific model for its document representation purposes. The following information about the types of model in information retrieval, including figure 2.1 that illustrates the relationship of some common models is based on [66]. Figure 2.1 categorizes the models according to two dimensions: the mathematical basis and the properties of the model.

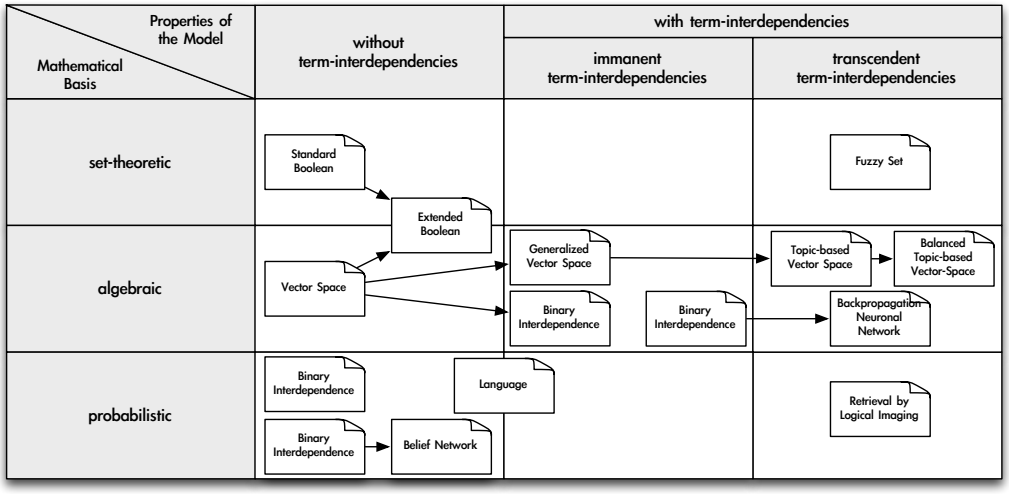


Fig. 2.1.: Categorization of IR-models (adapted from [66])

The vertical dimension

The y-axis of figure 2.1 defines the mathematical dimension. The models in this dimension can be classified into Set-theoretic, Algebraic, Probabilistic, and Feature-based models. The latter models were not taken into account in figure 2.1 as feature functions are arbitrary functions of document and query, and as such can incorporate almost any other retrieval model as just yet another feature.

Set-theoretic models represent documents as sets of words or phrases. Similarities are usually derived from set-theoretic operations on those sets. An example for those operations would be intersections combined with boolean operators. A further example for those operations would be the union. A common model for set-theoretic models may be mentioned in the Standard Boolean Retrieval model [67], the Extended Boolean Retrieval model [117], and Fuzzy Retrieval Model [117]. Hereby the Standard Boolean Retrieval model doesn't consider term weights in queries whereas the Extended Boolean Retrieval model makes use of partial matching and term weights. Further, the Fuzzy Retrieval Model brings together the Extended Boolean model and the Fuzzy set theory.

Algebraic models represent documents and queries usually as vectors, matrices, or tuples. The similarity of the query vector and document vector is represented as a scalar value. This value therefore defines the distance of two vectors and allows the comparison of vectors. Common models are the Vector Space model [118], the Generalized Vector Space Model [152], the Extended Boolean Model [117], and Latent Semantic Indexing [23]. Hereby the Generalized Vector Space Model introduces term to term correlations, which render unnecessary the pairwise orthogonality assumption of the Vector Space model. Those two models use metrics to determine the similarity of the query input to the stored documents, in contrast to the Extended Boolean Model that uses boolean comparisons. Whereas Latent Semantic Indexing is an indexing and retrieval method that uses a mathematical technique called singular value decomposition, to identify patterns in the relationships between the terms and concepts contained in an unstructured collection of text.

Probabilistic models treat the process of document retrieval as a probabilistic inference. Similarities are computed as probabilities that a document is relevant for a given query. Probabilistic theorems like the Bayes' theorem are often used in these models. Common models are the Binary Independence Model [151], the Probabilistic Relevance Model [108], the Uncertain Inference [137], certain Language Models, and the Latent Dirichlet Allocation [12]. The Binary Independence Model is a probabilistic information retrieval technique that makes assumptions in order to make the estimation of document similarity probability feasible. The Probabilistic Relevance Model similarly makes an estimation of the probability of finding if a document is relevant to an input query. This model is in contrast to the Binary Independence Model, which assumes that the probability of relevance depends on the query and document representations. Furthermore, it assumes that there is a portion of all documents that is preferred by the user as the answer set for the input query. In contrast, in Uncertain Inference a user's query can be interpreted as a set of assertions about the desired document. It is the system's task to infer, given a particular document, if the query assertions are true. If they are, the document is retrieved. Latent Dirichlet Allocation is a generative model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. For example, if observations are words collected into documents, it assumes that each document is a mixture of a small number of topics and that each word's creation is allocable to one of the document's topics.

Feature-based retrieval models like those discussed by the authors of [88] view documents as vectors of values of feature functions, or just features. Feature-based retrieval models hereby seek the best way to combine these features into a single relevance score, typically by learning to rank methods. As discussed in the explanation of the vertical dimension of figure 2.1, these feature functions are arbitrary functions of the document and the query, and as such can incorporate almost any other retrieval model as just yet another feature.

The horizontal dimension: properties of the models

The horizontal dimension in the figure describes the properties of the models. The models in this dimension of figure 2.1 can be classified into three different groups. In the first group there are the models without term interdependencies. In the second group there are solemnly models with immanent term interdependencies. In the last group there are models with transcendent term interdependencies.

Models without term interdependencies treat different terms or words as independent. A sentence is hereby merely seen as a set of words rather than as a concatenation of words that follows the complex grammar of a language. This fact is very likely to be represented in Vector Space Models [118] by the orthogonality assumption of term vectors. In the probabilistic Models on the other hand, they are represented by an assumption of independency for term variables.

Models with immanent term interdependencies allow a representation of interdependencies between terms. However the degree of the interdependency between two terms is defined by the model itself. The degree of the interdependency is usually directly or indirectly derived from the co-occurrence of those terms in the whole set of documents. This can for example be achieved by dimensional reduction.

Models with transcendent term interdependencies representation of interdependencies between terms, but they do not allege how the interdependency between two terms is defined. Models with transcendent term interdependencies use external sources to measure the degree of interdependency between two terms. This can for example be achieved by a human specialist in the desired field.

Taking into consideration the preceding presentation of dimensions, the CBIR technique that is used in this thesis belongs in the x-axis to the algebraic row, and in the y-axis to the column of immanent term interdependencies. More precisely, it belongs to the Generalized Vector Space Model. This is the case because of the applied weighted metrical feature distance calculation and the possible interdependencies between features.

Library

The SMART system [114, 116] was one of the first systems in information retrieval. It was manly used to look for dictionary entries. Nevertheless at that time in 1966, many important concepts like the vector space model, relevance feedback, and Rocchio Classification that are nowadays still attracting scientific interest were already defined.

Research

MEDLARS (Medical Literature Analysis and Retrieval System) [109] was a computerized biomedical bibliographic retrieval system. In 1964, it was launched by the US National Library of Medicine and was the first large scale, computer based, retrospective search service available to the general public. It therefore had an important role to demonstrate the advantages that come along with the new technology and to reduce the societies fears.

2.2. Fundamentals of Web Retrieval

Web Retrieval is the idea of information retrieval brought to the World Wide Web. Thus, web retrieval enables users to search for web pages, images, music, movies, and other types of files in the World Wide Web. For every search, the results are generally presented in a ranked list of results on the search engine's results pages. Unlike web directories, which were maintained exclusively by human editors, search engines maintain real-time information by running an algorithm on a web crawler. As the algorithm of this thesis is developed in order to provide this service for images in a content based method, this chapter introduces the historical and theoretical development of its' early proceeder and its most important fields of application.

The following section will therefore deliver insight into the historical development of the most important terms around and the standard topology of a system that is specified for web retrieval. The chapter will conclude with a demonstration of the various fields of application for web retrieval.

2.2.1. History

In the early years of the web, there was simply an HTML file hosted on the CERN webserver that referenced webservers that were online. A snapshot of this HTML file is still available[1].

[1] http://www.w3.org/History/19921103-hypertext/hypertext/DataSources/WWW/Servers.html

Eventually, more and more webservers were made available online and the centralized HTML file could not keep up with this development, even with the help the NCSA that lists new servers on its website[2].

Thankfully, early in 1987 Archie [36], the first tool capable of searching the web, emerged. The name Archie was derived from the word archive. Computer science students at McGill University in Montreal - namely Alan Emtage, Bill Heelan and J. Peter Deutsch - had the idea to download all the directory listings of the files located on public anonymous FTP sites. In this way they created a searchable database of file names. However, due to the limited amount of data, the sites were not indexed and manual search was necessary. Later, in 1991, Gopher [4] cleared the path for the search programs Veronica and Jughead. Indexes and titles were stored in Gopher index systems as they were catalogued by Archie. Veronica, which is the short form for Very Easy Rodent-Oriented Net-wide Index to Computerized Archives, made it possible to search for keywords of most Gopher menu titles in the Gopher listings. To retrieve the menu information from particular Gopher servers, Jughead, which is the short form of Jonzy's Universal Gopher Hierarchy Excavation And Display, was developed. In the summer of 1993, numerous specialized catalogues were maintained manually but no search engine yet existed for the web.

The web's first primitive search engine was released on September 22^{nd} 1993. It was basically a series of Perl scripts that would periodically mirror all the web pages and rewrite them into a standard format which formed the basis for the W3Catalog[2]. Therefore, the first web robot was built by Matther Gray in June 1993 [150]. It was a Perl-based World Wide Web wanderer and was used to create an index called 'Wandex'. Originally the motive to build the wanderer was to measure the size of the World Wide Web. It performed its task until late 1995.

Soon after the introduction of the first search engine, a second web search engine, -Aliweb, started its service in November 1993 [123]. In contrast to the W3Catalog, Aliweb had no robot attached. The system depended on notifications of the existence of each website from the administrators in a specific index format. The third web search engine, JumpStation was released in December 1993 [61]. It used a web robot to identify web pages and to create its index. Additionally it used a web form as an interface to its query program. It was thus the first web search engine that united the three essential features of a web search: crawling, indexing, and searching. Unfortunately its indexing and hence searching was limited to the titles and headings of web pages. This was the case because of the limited resources available on the platform on which the search engine was running.

A search engine called WebCrawler, started in 1994, was beneath the first full text crawler-based system [123]. In contrast to its predecessors, users could search for arbitrary words in the webpages. WebCrawler was additionally the first search engine that was broadly accepted by the public. Lycos was also launched in 1994. It became a major commercial endeavor. Soon new search engines appeared and tried to enter the recent market. Magellan, Excite, Infoseek, Inktomi, Northern Light, MSN Search (later Bing), and AltaVista were some of them. 'Yahoo!' was very popular in the beginning. Unfortunately its search algorithm did not work on full text copies of web pages but only on a web directory. In the late 1990s several companies entered the market. By now, some have specialized in enterprise-only editions, such as Northern Light, and many companies disappeared in the dot-com bubble in 2001. Around 2000, Google's search engine became publicly known. The company achieved better search results than its competitors by using the PageRank system. Google's search algorithm ranks web pages based on the

[2] http://home.mcom.com/home/whatsnew/whats_new_0294.html
[2] http://tinyurl.com/W3Catalog

number and PageRank of referencing web sites.

Nowadays there is still a selection of web search engines available. The major competitors are Yahoo!, AOL, ASK, Baidu, Bing, and Google. However, with a marketshare of 70.91 % the market is strongly dominated by Google.[3]

2.2.2. Theory of the Fundamentals of Web Image Retrieval

The usefulness of a search engine depends highly on the relevance of the result set it returns. As nowadays there are millions of web pages that include a particular word or phrase, some pages may be more relevant, popular, or authoritative than others. Therefore most search engines employ methods to rank the results to provide the best results first. How a search engine decides which documents are better matches, and what order the results should be shown in, varies widely from one engine to another. The methods also change over time as Internet usage changes and new techniques evolve.

There are two main types of search engine that have evolved. One is a system of predefined and hierarchically ordered keywords that humans have programmed extensively. The other is a system that generates an inverted index by analyzing texts it locates. The first method relies much more heavily on the computer itself to do the bulk of the work. As explained in chapter 2.2.1, current Web retrieval search engines examine the World Wide Web using web-crawlers. Figure 2.2 visualizes the schematic structure of such a web search engine. These previously mentioned web-crawlers, also referred to as spiders or crawlers, try to visit - starting from a seed of predefined web pages - all websites that are searchable in the World Wide Web.

In principle, every web page that is not explicitly blocked by its administrator is searchable. Blocking a web page can be accomplished by putting a file called robots.txt in the folder of the web page. This file is generally provided, and its proper usage well explained, by the search engine operators. While visiting every web page, referenced by any visited web pages, these crawlers are scanning the World Wide Web for documents. The documents found (e.g. web pages, videos, images etc.) are subsequently analyzed and indexed.

Indexing allows fast comparative querying for documents and therefore makes complete mirroring dispensable. However currently, search engines cache all or parts of the documents as well as their metadata to have them down pat whenever it is necessary. This cached document always holds the last known status, since it was cached while the document was indexed. The caching strategy is very useful when the content of the document has been updated or the search terms are no longer available in a web page, as it becomes possible to revert to the cached document.

This problem might be considered a mild form of linkrot, and the way the search engines handle it increases their usability by satisfying user expectations. When searching, e.g. for a web page, the user expects the search terms to be on the returned web page. Caching satisfies the principle of least astonishment, since in case of a dead or wrong link the cached page can be loaded. The major competitors in this domain cache the URL of a document and the HTML code or features. Both the HTML-code as well as the features are very compact representations of a document that allow comparative querying. For examople, for text documents the normalized set of words [147] that are suitable to describe the text as well as their frequency is a qualified feature. For images it is possible to use textual features as well as images that are usually surrounded by text (e.g. label) and an alternative tag frequently describes the image.

[3] http://www.netmarketshare.com/google-market-share

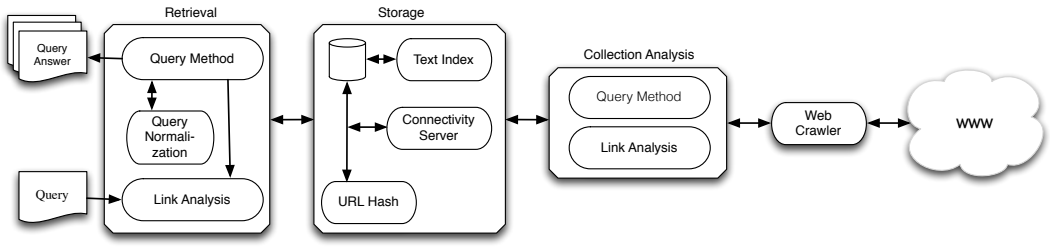


Fig. 2.2.: Schematic Visualization of a search engine – adapted froms [147]

Eventually, a vector is extracted from every document that contains the characteristics of the feature used to describe the document. From now on this vector will be referred to as the feature vector. Whenever a query is sent to a web search engine, a feature vector is extracted from the query document. Subsequently the similarity between this feature vector and the gathered vectors of indexed objects is calculated. Documents with a higher degree of similarity become part of the search result. Frequently used similarity functions are the euclidian distance (see formula 2.3) or the cosinus function (see formula 2.4) [3]. In both cases, d is the document feature vector that is currently used to compare, q is the query vector, and F is the set of features.

$$sim_1(d,q) = \sqrt{\sum_{j=1}^{|F|}(d_j - q_j)^2} \tag{2.3}$$

$$sim_2(d,q) = \frac{\sum_{j=1}^{|F|} d_j \cdot q_j}{\sqrt{\sum_{j=1}^{|F|} d_j^2 \cdot \sum_{j=1}^{|F|} q_j^2}} \tag{2.4}$$

It is notable that the euclidian distance returns smaller values for more similar vectors, whereas the cosinus function returns bigger values. Consequently the euclidian distance is basically not a similarity function, but a distance function [3]. [3] also offers a good overview about similarity functions and introduces a method for the automatic creation of the fitting parametrized function for the respective use case. For further studies of the architecture of web search engines' generation, [147] explains the architecture and the implementation of the search engine *IntelliSearch*.

2.3. Fundamentals of Web Image Retrieval

Due to the digitalization and miniaturization of cameras, as well as their integration into mobile phones, the amount of digitally available images has increased tremendously in the last decade. In order to make those images searchable, a tremendous effort has been made in research as well as in economy. E.g. in July 2001 Google introduced a search service that allowed users to search the Web solely for images[1].

The keywords for the image search are based on the filename of the image, with the link text pointing to the image, the alternative tag, and the text adjacent to the image. When searching

[1] http://www.nytimes.com/2001/07/12/technology/news-watch-a-quick-way-to-search-for-images-on-the-web.html

for an image, a thumbnail of each matching image is displayed. When the user clicks on a thumbnail, the image is displayed in a box over the website that it came from. The user can then close the box and browse the website, or view the full-sized image. As the algorithm of this thesis is developed in order to provide this service in a content based way, this chapter introduces the historical and theoretical development of its predecessor and its most important fields of application.

2.3.1. History

Thanks to the the digitalization and miniaturization of cameras, and their integration into mobile phones, as well as the evolution of the world wide web to Web 2.0 [97], the number of available images, especially photos, has increased considerably [22, 85]. Online sharing platforms like *Flickr*, *Picasa* and *PhotoBucket* as well as social networks like *Facebook* enable users to make their content available online. From the last public announcements of the companies, the following numbers can be cited: *Facebook* was said to host over 100 billion images by the end of 2011. *PhotoBucket* currently hosts more than 10 billion images. Finally *Flickr* announced in the end of 2012 that they hosted 8 billion images. Obviously an amount of images of this magnitude is not manageable without effective searching and indexing algorithms [85].

Texts have been manually arranged in libraries for hundreds of years and can be searched with textbased search engines. An automatic arrangement of images however is more difficult [127, 22, 84]. The exploration of image retreival began around 1990 [84, 127] and since 1997 [127] has become a dense field of research. The authors of [22] refer to the years between 1995 and 2005 as exhibiting an exponential growth of publications in the field of image retrieval. Due to the increasing number of publicly available images and new use cases that emerged, involving smartphones, the research interest continued to rise in subsequent years. A basic understanding of image retrieval can be learned from [84]. A comprehensive overview of state-of-the-art developments and prospective outlooks is offered by [127, 22].

2.3.2. Definitions

This section should give a general view of the research field of image retrieval. In order to introduce the nuances of the topic this subsection defines its most important terminology:

Metadata: Data that describes the background of a data is called metadata [22, 84]. Metadata is occasionally written into image files (e.g. the name of the owner, copyright & contact information). Examples of metadata for images are; size of the image, primary color, what camera created the file (for photos), along with time information such as when the image was made and descriptive information such as keywords about the photo. In the context of this thesis, even text surrounding the images, the alternative tag, and names of images are considered as metadata.

Result page: Result pages are the pages major search engines use to display their results. When using the textual image search from any major search engine, the engine presents the images it considers as results in a page. Usually this page is filled with images from left to right and up to down depending on the relevance. This page is referred to as theresult page.

Search suggestion: A Search suggestion is a term search engines use to complete the first letters of a text a user enters in the search field of a search engine while they are still typing.

When using the textual image search from any major search engine, the engine attempts to guess what the user is interested in. Therefore, usually nearly invisible, in the beginning of the result page, suggestions for alternative textual search terms are suggested. These terms are referred to as search suggestions.

2.3.3. Theory of Fundamentals of Web Image Retrieval

The objective of every image retrieval system is abstractly summarized by [127, 22] as the overcoming a) of the sensoric gap, which exists between an object of the real world and its photographic or digital representation, and b) of the semantic gap, which exists between the information which was gained through the digital processing of the image and the interpretation of the same image by an individual user.

In image retrieval many perspectives have to be considered in order to take into account these gaps. Is the user browsing, surfing, or searching? Is he aiming for an image within a broad or a narrow domain? Is he browsing freely, searching text based, or using an interactive or a combined approach? The multiplicity of possibilities depend on the applied retrieval strategy, the domain of the coveted image being broad or narrow, and the applied search type. Therefore each of these options will be discussed in the following subsection.

Retrieval Strategy

There are three strategies a user can apply to retrieve images from an image retrieval system. Namely they are called Browsing, Associative Search, and Search. They are described briefly below.

Browsing: Browsing does not consider a clearly defined task or queries. The retrieval process can stop at any moment with or without any result. Users that are using this retrieval strategy are most likely browsing as a leisure activity. An example for a person that uses this retrieval strategy is somebody that searches the Internet using a search engine for random terms that come into his mind. He might search for 'train' and obtain no interesting results. Subsequently he might search for 'apple', 'water', 'red', and 'turquoise'. In the 'turquoise'request one result might be returned that seizes the attention of the user, and he might enter the website and stops the browsing.

Associative Search: In the associate search the target of the search is vaguely defined and the approach is iterative. Due to the flexible nature of the approach, it leads to related topics that the user can access and start other iterations. User that are using the associative search are most likely trying to gain a visual overview of a certain domain. In fact, Google utilizes, and thus every user is performing, an Associative search. Google analyzes the search requests of every single user. If the user is logged in using his Google account this analysis can be done more precisely. By using the extracted information, it optimizes the results to a user.

The search focus of the user is refined and can be interpreted as a long term associative search. A short term associative search would be a user that tries to test his knowledge about mushrooms and uses Google Image Search. He types in 'mushroom' and receives 100 images. S/He recites the names of the first 35 mushrooms and chooses mushroom number 36. S/He taps on mushroom number 36 to get visually similar images. Using those images s/he recognizes the kind of mushroom from image 36 and continues with mushroom number 37.

Mushroom number 64 is unknown to the user and therefore he clicks on the image and gets to the website it is from and reads further details about it. The Associative Search is finished at that moment.

Search: The objective of the search is well defined and the search should lead to a result that is fast and determined. The search is performed in order to look for a specific motive or for results that represent a determined category. User searching in this focused way usually do so when the search is part of a bigger function (e.g. preparation of lecture slides).

Narrow and Broad Domains

There are two extreme types of domains a user can retrieve images from. They are defined by the authors of [127] as narrow and broad domains. Both of them are described briefly in the following subsections, but many gradations between those two extreme types of domains exist.

Narrow Domain A narrow domain contains images with limited variability in their descriptive aspects. The images differ marginally, as for every image a similar setting is used (e.g. same il- lumination, background, perspective). Passport images, herbarium images, logos, and product images are examples of narrow domains.

Broad Domain A broad domain contains images with an unlimited, unpredictable variability. The images differ strongly, even though they might show the same concept semantically (cf. differing perspectives). Images of landscapes are very representative examples of broad domains. The illumination depends on the current solar radiation, the background depends heavy on the chosen motive, and the perspective is freely selectable.

Result Presentation

Results can also be visualized for the user in different ways, both in text based systems and CBIR systems. The main presentation type would be a list, ordered by the found images' relevance. In the approach of this thesis this is the return type that is analyzed. The images of these lists can also have associated score values, which will improve the quality of our analysis results or have no associated score value. In this case the algorithm utilizes only the ranking of the images. Other presentation types exist:

Time-ordered Images are not ordered by their similarity or relevance to the query, but rather by their creation date.

Clustered Images are clustered by their metadata or content. This can be visually appealing to a user, but is hard to process using an algorithm as no linear ranking can be extracted. A survey of different methods can for example be found in [39].

Hierarchical Here, images could be arranged in a tree order constructed from their metadata, for example.

If a CBIRs supports merely time-ordered, clustered or hierarchical result presentation, it is not analyzable nor usable for the approach that is developed in this thesis. This is due to a lack of information that the approach requires such as the scoring or at least a totally ordered rank for each image.

2.3.4. Application Examples

Even though every major search engine competitor has by now implemented image retrieval, the search for images in the World Wide Web is just one major example of use . The other major use is as follows: many image services are for instance specialized on image searches in specific domains (icons[2], photography[3] or food[4]) in order to satisfy the needs of a specific, most commonly paying, costumer group. These services give the user the option to illustrate, e.g. a cook book or a magazine, without having to take a photo theirselves.

2.4. Fundamentals of Content Based Image Retrieval

As already mentioned in the beginning of the precedent section the amount of digitally available images has increased tremendously in the last decade. In order to make those images searchable based on content, Content Based Image Retrieval (CBIR) has received a lot of attention in recent years. The theoretical background which was developed during these years has been exhaustively explored in [22] and [127] and various systems implementing CBIR were examined in [139] and [64]. A Content Based Image Retrieval system (CBIRs) allows the searching of images based on their extracted features (low- or high-level [63]) instead of textual descriptions. The retrieval process matches the extracted features of the stored images to those of a query image, thereby calculating its score and rank, which subsequently results in a list of best matches. The general mechanism of the search process is very similar for all CBIRs. This chapter introduces the historical and theoretical development of Content Based Image Retrieval and its most important fields of application.

2.4.1. History

Since the beginning of the 90s, CBIR has developed into a broad and intensive field of research [20, 147]. It combines techniques from the fields of image processing, computer vision, statistics, and database management. Image searching, which relies solely on the visual content of the images, was rarely explored at this time. The methods developed at this time were predominantly text based [127] and relied on appropriate annotations of the images. While research focused on efficient storage and organization of images in pictorial databases in the mid-80s, it was not until years later that research was faced with the problem of a meaningful annotation and the actual search [112]. The desired objective description of an image in CBIR is in contrast to the subjective perception of a human being. The wealth of information of an image is reduced by the annotator to a few, subjectively important aspects. In addition, the process of manual indexing was context dependent and often incomplete in terms of the need for a cohesive taxonomy of tags used [131].

With the increased support of multimedia content on the web and the active participation of users in social networks, the amount of multimedia content available worldwide has increased dramatically for digital imagery in the last decade. The manual tagging of these image archives is hopeless because of its enormous volume [107]. Various web-based tagging systems such as Flickr[5] or Picasa[6] still try to meet this challenge with an active community, the content is

[2] https://www.iconfinder.com/
[3] http://www.fotosearch.com
[4] http://international.stockfood.com
[5] http://www.flickr.com
[6] http://picasa.google.com

provided with arbitrary keywords and thus the search can improve the semantic level. The syntactic description of the image content, such as the spatial arrangement of objects, is in this case usually neglected [3].

The difficulties of the objective textual description of image contents on the one hand, and a greater availability of computing power on the other hand, drove the development of content-based image search systems in the early 90s. In contrast to text-based image retrieval, the basic philosophy of the content-based method is the description of the image content by means of the embedded features [48]. This, for a specific image description, is referred to as the feature. With appropriate algorithms of digital image processing, pattern recognition and computer vision features can be extracted directly from the digital data of the image by analyzing the properties and relations of the discrete image points. Due to the subjective perception of the image content in different contexts, no single feature content can comprehensively cover all aspects of an image. A variety of advanced visual features are specialized for different visual properties of an image to extract a more complete description. The theoretical background of CBIR is intensely explored in [22, 127] and various systems implementing CBIR were surveyed in [139].

2.4.2. Definitions

This section should give a general view of the research field of Content Based Image Retrieval. In order to introduce the nuances of Content Based Image Retrieval this subsection defines its most important terminology:

Features are in pattern recognition and machine learning n-dimensional vectors of numerical features that represent an object. In CBIR those features are used in order to make images comparable using feature metrics. Those metrics usually calculate numeric values between two images. Mostly, these values are smaller the more similar the two images are. A perfect match would therefore be next to zero.

Low Level Features describe images in the lowest level of visual features; color, shape, and texture. Features are built using the color distribution, the most dominant color, or the number and location of edges.

Mid Level Features are local features combined into a global image representation suited to recognition using a common classifier such as a support vector machine [16].

High Level Features describe images in the highest level of visual features; elements and context. E.g. a car and a person that are depicted in an image and the additional information that the person gets from the car at that moment.

Global Features are composed of color, shape and texture features that are computed on the entire image. As only a part of an image is typically relevant for an image (e.g. in medicine) those features are not always sufficient. But, as they are easy to implement and they are comparably efficient, they are nevertheless still frequently applied.

Local Features are based on the premise that images can be characterized by attributes computed on regions of the image. The most prominent examples for this type of feature might be Scale-Invariant Feature Transform (SIFT) and Speeded Up Robust Features (SURF) .

2.4.3. Theory of the Fundamentals of Content Based Image Retrieval

In recent years, as surveyed in [139], a large number of CBIRs have been developed, which differ mainly in the features used, indexing methods, and the similarity in search methods [93]. From each image which is contained in the database of the CBIR system, a quantity of different features are initially extracted and stored together with a reference to the image. By using several features of the image, it can be described from various points of view and the image search can be aligned to certain visual aspects. In addition to these usual features, also additional content-independent metadata is stored. The user formulates his information needs as a search query and passes it to the CBIR system. From the query, the system's known features are extracted and the similarity of the features of the query with the respective features of all data stored in the database images are calculated according to their features. The result of this search is an organized list of images from the database, which is ranked by similarity. In theory all images of the database are included in the result set, but in reality only the highestranked images or images that are inside a certain similarity threshold of the image are displayed for reasons of clarity and usability. In [64] several research approaches, like Tattoo, LIRe, SIMBA , SIMPLicity, ImageFinder, PictureFinder, Oracle Multimedia, and many more were introduced.

In the following sections the aspects of CBIR which are relevant for this thesis will be discussed. The first section therefore thematizes the Semantic Gap and its importance to this thesis. This section is followed by two paragraphs that discuss features and distance measures. Eventually the last two sections talk about different occurrences of search strategies and standards that can be implemented to interact with various CBIRs.

Semantic Gap

The problem of the semantic gap, defined in section 2.4.2, has existed since the conception of CBIRs. The semantic gap is described as the existence of a divergence in meaning between the system and the user [89, 38]. [127] provides the following definition for the semantic gap:

> *"The semantic gap is the lack of coincidence between the information that one can extract from the visual data and the interpretation that the same data has for a user in a given situation."*

The pixels of an image with their color information and their position are at first a grid for itself. Using this information, operations can be performed that restructure, reorganize, and eventually summarize it to consequently generate a new image description. Yet this description still determines merely from the available pixel data and the calculation formula of the particular syntactic feature. It is unique, reproducible, and reflects the arrangement of the pixels, the colors used, and the edges and shapes for this image in a certain way.

The search for a picture is, however, based on the subjective meaning of the image for the user. S/he searches for images that contain certain objects, convey a particular message, or are (in his view) associated with emotions. The formulation of his search is performed on a semantic level and abstracted from the concrete representation of this concept. Thus, a user might attach to his search image meanings for which no direct visual correspondence exists. Accordingly, the automatically extracted syntactic image description does not necessarily coincide with the substantive interpretation of the user.

As an example of this phenomenon is called the description of an image with natural language [127]. It is easy to understand that different users can represent the same image with

different linguistic descriptions. The expressiveness of natural language supports a subtle differentiation of the substantive aspects and their classification in the interpretation of the whole image content. Some parts of the content may be particularly emphasized and others are neglected, and this can still give an adequate description. It is almost always context-dependent, subjective, and captures the content as it appears useful for the user in the current situation. The gap between the various user-dependent meanings of an image (High Level Semantics) on the one hand and the rigid representation of an image in the form of features (Low Level Features) on the other hand, is the stated goal of the CBIR [22]. This is the aspect where the weighting of features becomes important. Images can at least partly be grouped into categories with unified semantic concepts using these weightings. The features that are important in the included images can then be examined for each of these picture's categories. This search for the essential visual aspects thus becomes a search for the optimal feature weighting of the image's category.

Features

The basis of any CBIRs is the extraction of features from the available data. The computational processing is done on the basis of digitized images, which consist of an array of discrete pixels with associated color values. A technical system is inferior in terms of the detection accuracy and speed of the human perceptual apparatus in disrepair. Tasks such as segmentation, recognition of concepts, and subsequent substantive interpretation are dealt with by the visual cortex in near real time and without noticeable effort. This stands in contrast to the human's superior ability to determine quantitative values accurately and objectively from images. Here, the detection apparatus of people can only make relative and subjective information.

The semantic interpretation of image data is the current state of research only in terms of content limited application areas, possible under strictly defined environmental conditions (see, face recognition [155, 135, 100, 82, 47], handwriting recognition [133, 62, 35, 113, 18], iris recognition [80, 79, 54, 94, 138], cancer recognition [136, 26, 105], medical image analysis in general [81, 101, 99, 27, 46], and medical diagnostic devices [93]). The previously mentioned limitation arises from the fact that images of a narrow image domain have a limited and predictable variability to their appearance in all relevant aspects of the image content. Their semantic meanings are in general homogeneous and objectively determinable. In contrast, the variability of the external appearance tends to be higher for images of a broad image domain and consequently expect an increasing heterogeneity of occurring semantic concepts.

Broadly speaking, in general application areas where the formation of a specialized model is not possible due to lack of a priori knowledge, one will have to initially concentrate on feature-based method to use syntactic features for indexing of images [127]. Since the dawn of CBIR, a variety of features have been developed, and each is different for each task area of visual properties of an image out-set [25]. Obviously there is no single feature that will meet all requirements.

Therefore, as a rule, there are several different features that each encode special geometrical, topological, and statistical characteristics, combined into a meaningful image description. Although these characteristics always extract properties that correspond with those features perceived by the human eye, a descriptive interpretation of factors is often difficult because of the complex calculation process and the relational nature of the information embedded in an image. Features are simply image descriptions or representations.

Feature extraction is a method of transforming the pixel values of an image into a feature

signature, which consists of a vector or a set of vectors. It is possible to assess how similar two images are with respect to a feature by extracting the feature signatures of the two images and applying them to a distance function. A smaller distance function value means that their content is also more similar with respect to the type of content the feature analyzes. There are many different kinds of features and they can be roughly categorized as either low level or high level [63]. Low level features are those representing the color, texture, or shape of an image, while high level features include for example face recognition data. Some CBIR systems use feature vectors derived from the whole images (global features). Alternatively, features can also be used only on image regions (local features), and multiple times for different regions. An image would in that case be segmented first and the feature signature calculated from the separate segments.

The MPEG-7 Multimedia Content Description Standard was designed in order to provide a standardized way to describe and annotate stored image data, among other things. One part of MPEG-7, the MPEG-7 Visual Standard, contains the color layout, scalable color, and edge histogram image features. These MPEG-7 features (Tamura, Color and Edge Directivity Descriptor, Fuzzy Color and Texture Histogram, Auto correlation Feature) were implemented in Lire [77], and comprise the CBIRs that is used in the implementation of this thesis. In annex A a selection of MPEG-7 features that are implemented by Lire are explained to give a deeper understanding of features.

Distance Measures

The similarity of two images is usually measured in computer science by the distance between features. By extracting the feature signature of both images and subsequently applying a distance measure on the vectors, a feature distance value is calculated. The bigger this value is, the lower the similarity is. Feature authors frequently recommend a distance measure to be used with their feature. In general the recommended measure is specifically designed to return good overall distance values in combination with the developed feature. It is therefore mostly unhelpful to deviate from the recommended measure.

Thus, from the perspective of this thesis' subproblem of analyzing a CBIRs that potentially uses a certain feature, the probability is very high that the distance measure that was recommended by the author is also the distance measure that the analyzed CBIRs uses in combination with that feature. Unfortunately, the eventuality that a different distance measure for a feature can be used cannot be discounted as different distance measures are theoretically applicable for a feature. For example, this can depend on whether the feature signature consists of only one vector, multiple vectors, or another form of signature representation. Feature signatures often consist of one or more histograms, which can be represented as one or multiple vectors.

One problem many distance measures share is that they do not take neighboring histogram bins into account and so return for example too large distances between histograms, if their bins just barely do not match. Examples of distance measures or functions are: First, there are the obvious examples of euclidian distance L2, Manhattan Distance L1, and the maximum norm, all belonging to the Minkowski group of distances. They can only be used to compare two vectors, not sets of vectors. Other distance measures used for image retrieval include the earth movers distance [111], the signature quadratic form distance [9], the Histogram intersection [132] (a generalization of L1), and the Hausdorff distance [55]. A comparison between the earth movers distance, signature quadratic form distance, and Hausdorff distance can be found in [9]. An experimental evaluation of additional distance measures is given in [110].

Search Type

Until now this chapter has spotlighted mainly the technical perspectives of CBIR. There is however a further, so far not accurately studied, element which is integrated in the search process. The user is this last element. Therefore as the final aspect in this theoretical study of CBIR, the varying search types will be discussed below:

- **Free Browsing** In free browsing the user is not querying, but browsing the stored data manually as he does in the 'browsing' retrieval strategy . It is obvious that a directed search for specific images or motives in the dimensions of domains, that were defined at the beginning, is difficultly or impossible [84]. The authors of [84] suggest the optimization of browsing by applying cluster techniques for grouping visually similar images. In case of annotated images, a hierarchical grouping by topic of the images is also possible [22]. An example for such a search is *Wikimedia Commons*. *Wikimedia Commons* has 11.5 million objects that are hierarchically categorized.

- **Text Based Search** As pointed out in chapter 2.2.1, there are many web search machines available that allow users to search for text documents using keywords. It is an obvious idea to modify the same strategy for images [127]. The only necessity for the implementation of this idea is an accurate textual description or some exact catchwords which give information about the motive of the image's metadata. Nowadays every major search engine implements a text driven image retrieval method.

  A text based search, as well as all of the following approaches, can be applied for each of the retrieval strategies that were introduced in subsection 2.3.3. For text based searches and extraction of the metadata, various methods are available. Classically speaking, images are annotated in a manual or a semiautomatic way [22]. This task is however very labor intensive [22, 127]. Luis von Ahn and Laura Dabbish assumed that the accuracy of content-based approaches would not be acceptable for annotating images automatically. Therefore, to annotate a maximum number of images with metadata they developed the ESP-Game.

  This game should playfully animate users to annotate images with the fitting metadata. In [145] the authors postulated an anticipation that 'If the ESP-Game is played by as many players as other online games are played at that time, then every image available on the Internet would be annotated in a few months'. As long as the images that are annotated originate from the web, the metadata can be extracted from the image's name, the alt-tag, and the surrounding text.

- **Interactive Approaches** In the beginning of the research on CBIR it was assumed that current image processing techniques would one day be powerful enough to achieve a sufficiently high quality of image retrieval. These days, in order to improve the retrieval performance, an interactive search process in which the user is involved repeatedly in the retrieval process in a kind of loop began to be investigated in [84]. A system using relevance feedback lets the user review the relevance of the features used, the images found, or single image regions in found images. This approach has resulted in an iterative process, which increasingly limits the results to the desired objects.

  For the user it becomes a simpler and more natural search method, since an exact request must not necessarily be formulated at the beginning, but the requests can successively be 'improved' [84]. Relevance Feedback in general is described in detail in [112]. An

example of such a system is *PARISS* [45]. The user can classify images into two categories for relevant and non-relevant search results. Furthermore, the calculation of the similarity of the individual results can be manipulated to one another: All images that are rated as relevant are displayed on the screen as thumbnails and a user can move each photo closer or further apart according to his perception of their similarity. The system will now generate results on the basis of existing features and the new features by relevance feedback, which reproduces the similarities described by the user.

- **Combined Approaches (e.g. [57])** As previously mentioned, image retrieval has attracted considerable attention over the last decade. So far, however, the majority of the large amount of existing information retrieval systems of request scenarios is spread out, into which requests are directed only to a single search service [28], It would nevertheless be desirable to be able to send queries to multiple databases simultaneously and obtain their results collected in an aggregated form. In this way, a much larger amount could be searched for images using a single request than was previously possible. Problems which have so far largely prevented such applicationsinclude the lack of a unified request format and the variety of image descriptors used.

  Today there is a variety of quasi standards for metadata and low-level descriptors such as*MPEG-7* [134], *Dublin Core* [2], etc. [28, 31]. The goal of a meta search engine is to receive requests and forward them to the registered information retrieval systems. Subsequently, the individual results must be aggregated and returned in an aggregated form to the client.The communication with the individual retrieval systems is executed via query interfaces, which translate queries into the specific request format of the respective retrieval system and transforms the results, and in particular their metadata, into the format used by the metasearch engine. Since not all browsers support all request methods, the metasearch engine should decide which service a specific request can be forwarded to. In addition, the retrieval quality of services can be analyzed and included in the decision. An early implementation of such a meta-search engine was *MetaSEEk*[20]..

  Recognizing the need for a standardized image retrieval distribution, the emph *Joint Photographic Experts Group (JPEG)*[7] started with the definition of the ISO / IEC standards *JPSearch* (ISO / IEC 24800). This defines both interfaces for the communication of search services with one another, as well as a modular, service-oriented architecture, and a proper search engine query data and metadata format.

Standards

Subsequently this section briefly discusses the standards that can be implemented to interact with various CBIRs. Therefore the well known standards MPQF [29] and SQL/MM [86] are briefly discussed as examples.

MPQF is a standard developed by the Moving Picture Experts Group. It was developed because existing multimedia query languages mostly allowed no weighting of individual part queries that mirrored the user's preference for a specific multimedia object e.g image. Furthermore, these days, proprietary metadata formats are used that mostly lack the possibility to combine queries, like a textual query and a content based query. The problem of current xml-based query languages appears to be mostly their ill fit for content based queries [29].

[7]Officially called*ISO/IEC Joint Technical Committee 1, Subcommittee 29, Working Group 1*

MPQF is a query language that defines the query and reply formats exchanged between clients and servers in a distributed multimedia search- and-retrieval scenario. The two main benefits of the standardization are its interoperability in a distributed scenario and platform independence. One further key feature of MPQF is that it allows the expression of multimedia queries that combine the expressive style of information and XML data-retrieval systems. Thus MPQF allows users to combine, for example, keywords and query-by-example, with, for example, XQuery.

SQL/MM is the multimedia descendant of SQL, and like SQL is a multi-part standard. Part 1, known as the framework, provides definitions of the common concepts used in the other parts and outlines the definitional approach used by those other parts. In detail they contain definitions to manage full-text queries, spacial queries, still image queries, and data mining queries [87].

A.I.R

The implementation of namely **A**rchitecture for ***I**nter-operable Retrieval on Distributed and Heterogeneous Multimedia **R**epositories* [129] is a Java based framework that separates the heterogeneity of CBIRs and aggregates their search results. In the implementation of this thesis, an early *A.I.R.* implementation, called *QUASI:A* (see appendix B), was modified in order to obtain a basis for the necessary evaluations. Hereby individual distribution algorithms and aggregation algorithms were developed, implemented, and evaluated. *A.I.R.* uses the *MPEG Query Format* [29] as well as the transformation rules described in the *JPSearch Standard* [29]. As an internal metadata format, it uses the JPSearch Core meta data format (ISO/IEC 24800).

A.I.R. allows the user to send unified multimedia queries to heterogeneous systems. Thereby it is isolated from the individual query protocols as well as the used meta-data formats. *A.I.R.* is used as a starting point for this thesis' implementation. [129] introduces an architecture for a middleware component separating the heterogeneous environment of multimedia data stores. The development of the framework pursues the following main requirements: modular architectural design, a broad scope of multimedia retrieval paradigms (e.g., query by example), unified multimedia requests, and cross system multimedia retrieval (cross metadata as well as cross query language).

Furthermore *A.I.R.* supports multiple query processing strategies (autonomous and federated). Concepts and modules/placeholders for intelligent query segmentation and distribution, as well as aggregation of the result set, are the highlights of the current implementation. However, the actual retrieval process of the multimedia data is performed inside the connected backends. In the following, possible search concepts in this domain are discussed followed by an introduction of the

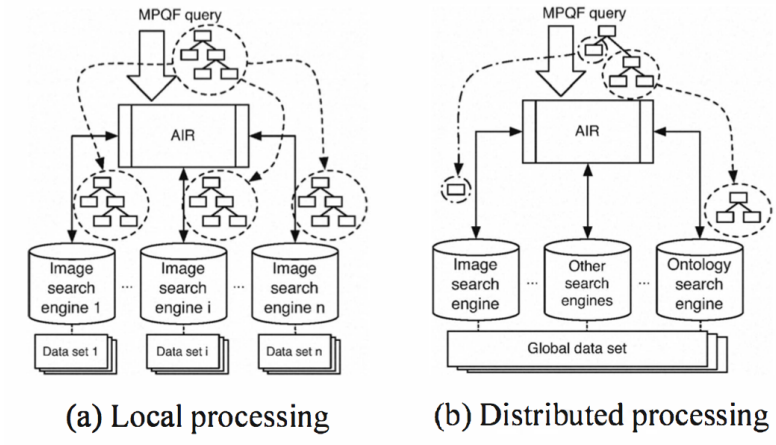


Fig. 2.3.: AIR query processing strategies from [129]

overall system architecture.

A.I.R. can be applied in many different ways within distributed and heterogeneous multimedia search and retrieval. Generally, the tasks of each *A.I.R.* component (see figure 2.4) are highly dependent on the registered databases and use cases. Hereby, distinction should be made between the two query processing strategies (see figure 2.3). The first strategy deals with a scenario similar to that of this thesis. Registered and participating retrieval systems are able to process the whole query locally, see Figure 2.3 (a).

Those heterogeneous systems most problably provide their local meta data format (e.g., Dublin Core, MPEG-7, etc.) and an individual data set. An input query is under these circumstances processed as a whole and the items of the result set are the outcome of an execution of the query. The necessary transformation of used metadata formats is hereby imperatively performed. Additionally the individual result sets may contain duplicates.

However, as the result aggregation process only needs to perform an overall ranking of the resultant items of the retrieval systems involved, the duplicate elimination algorithms may be applied as well. The second strategy, which has no overlaps with this thesis' approach deals with registered and participating retrieval systems that allow distributed processing on the basis of a global data set (see Figure 2.3 (b)). The involved heterogeneous systems might depend on different data representation (e. g., ontology based semantic annotations and XML based low level features) and query interfaces (e.g., SPARQL and XQuery) but use a common global data set.

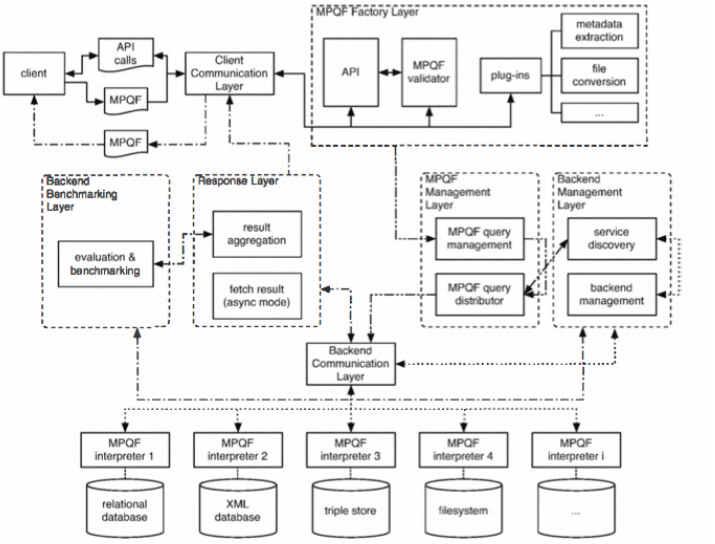


Fig. 2.4.: Overview of the AIR components from [129]

Figure 2.4 illustrates an *A.I.R.* workflow scenario in a distributed multimedia retrieval scenario. Therefore in the following, all components of *A.I.R.* will be briefly discussed. The main functionalities of the **Backend Management Layer** are the (de-)registration of backends.The main purpose of the **MPQF Factory Layer** is the generation and validation of MPQF queries.**The MPQF Management Layer** organizes the registration of MPQF queries and their distribution to the applicable retrieval services. The **MPQF Interpreter** is located at the backend and is supposed to act as a mediator between *A.I.R.* and a particular retrieval service. Its main purpose is to receive an MPQF query and transform it into native calls of the underlying query language (e.g., SQL MM in an Oracle database). After successful retrieval, the MPQF Interpreter forwards the result set (converted in an MPQF response) to the *A.I.R.* framework. The **Backend Bench marking Layer** is responsible for the aggregation of the different result sets. The **MPQF Response Layer**, in general, performs the result aggregation and returns the aggregated result set.

2.4.4. Application Examples

Even though every major search engine competitor has by now implemented Image Retrieval, the search for images in the world wide web is just one major use case. Many image services are specialized on image searches in specific domains (icons[8], photography[9] or food[10]) in order to satisfy the needs of a specific, most commonly paying, customer group. These services give the user the opportunity to illustrate e.g. a cook book or a magazine without having to take a photo themselves.

2.5. Fundamentals of Distributed Retrieval in CBIR

In the following section a basic understanding of the emerging topics around distributed retrieval in CBIR will be established. This will be executed by giving an overview of the key points of distributed retrieval in CBIRs. This overview starts with a brief review of the history of distributed retrieval in section 2.5.1. This is followed in section 2.5.2 by definitions of the technical terms that are used in the subsequent sections of this chapter. Subsequently, section 4.3.1 discusses the theoretical background for distributed retrieval in CBIR. Eventually, section 2.5.4 characterizes and gives examples of the fields of application and current implementations of distributed retrieval in CBIRs.

2.5.1. History

In recent years, a large number of CBIRs have been developed. Most of them are specialized in certain areas of application. Despite the enormous number of different systems [140], their architectural principles like feature extraction, feature comparison, and returned result list are usually very similar [78]. This opens up the possibility to harness the potential of different systems through their integration into a CBIR network (CBIRn) and take advantage of the specialization of each CBIRs [143].

Although a network might have been composed out of many CBIR systems, a uniform treatment of the systems was impossible due to the divergences in the details of its implementations (e.g. query language used). Therefore a branch of the research community developed and implemented meta languages in the last decade that enable interaction with a multiplicity of current CBIRs (e.g. A.I.R. [128]). Thanks to these implementations the search no longer has to be processed by an individual CBIRs but by a combined CBIRn. A preceding component must then determine which of the registered systems in the network provides the best results for the input image. When these systems are automatically selected, the query can be routed appropriately and the relevant results are combined through a merging process.

2.5.2. Definitions

This section should give a general overview of the research field of Distributed Retrieval in CBIR. In order to introduce the particular nuances of Distributed Retrieval in CBIR this subsection defines its most important terminology:

[8] https://www.iconfinder.com/
[9] http://www.fotosearch.com
[10] http://international.stockfood.com

CBIRn: A Content Based Image Retrieval network (CBIRn) is a network of CBIRs that can be queried for an input image. These CBIRs compare the input image to their stored images in order to identify and return similar images and return them to the querying instance.

Enrollment: Enrollment is the process a CBIRs has to go through in order to integrate itself into a CBIRn. There are two divergent types of enrollment. In a cooperative enrollment scenario the enrolling CBIRs is willing to provide its configuration to the CBIRn whereas this is not the case in the uncooperative enrollment scenario.

Normalization: Normalization has to take place in the uncooperative enrollment process. This means adjusting the values measured on different scales to a common scale. In order to be able to compare scores of various CBIRs, a normalization step is applied.

Configuration Detection: Configuration Detection takes place in the uncooperative enrollment scenario. In this scenario the CBIRs does not provide information about its internal configuration. Therefore the CBIRn has to compute the configuration of the enrolling CBIRs by evaluating the result lists that are returned to predefined query images, which is called configuration detection.

Concept Detection: Concept Detection is the extraction of a semantic concept by analysis of an image. It is one of the first steps performed in the CBIRn that will be defined in the second part of this thesis. It is done by extracting the image's features and processing the information through previously trained algorithms.

2.5.3. Theory of the Fundamentals of Distributed Retrieval in CBIR

The following section discusses the theoretical background of distributed retrieval. It starts with the enrolling process that integrates CBIRs in the CBIRn, in both its cooperative and uncooperative versions. Subsequently the concept detection, which becomes necessary for the distribution process of a query in the CBIRn, is spotlighted. Eventually this section discusses the research merging approaches that are necessary for distributed retrieval.

2.5.3.1 The Enrolling Processing

The first contact between a CBIRs and a network of CBIRs is established by the enrolling process. Depending on the involved CBIRs' readiness for cooperation, a cooperative - as well as an uncooperative - scenario is possible. In a cooperative scenario, a CBIRs contacts the enrolling process and offers all details about its configuration (e.g. used features, metrics, or feature weight). After having received the configuration details, the enrolling process forwards it to a knowledge database. The knowledge database in this case is merely a database that registers configurations of CBIRs. In the uncooperative scenario, a CBIRs contacts the enrolling process but does not offer details about its configuration. The enrolling process hereon sends queries to the CBIRs with the objective to determine the CBRIs' configuration from its replies. Concluding its function, the CBIRs' determined configuration is forwarded to the knowledge database. The following paragraphs will highlight different aspects of these two scenarios.

The cooperative Enrolling Process

The cooperative enrolling process assumes the CBIRs will voluntarily reveal its configurations as well as details about its internal operations. However it is unlikely that every company

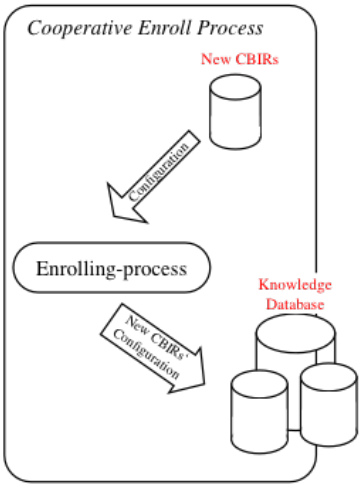


Fig. 2.5.: The cooperative Enrolling Process

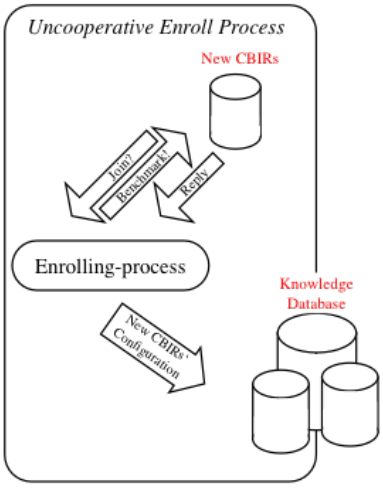


Fig. 2.6.: The uncooperative Enrolling Process

would be willing to lay open its CBIRs' operating mode. Nevertheless there is still a small opportunity for public CBIRs to become dominant. Therefore the cooperative enrolling process is merely superficially discussed in the following section. The cooperative enrollment process is visualized in figure 2.6 on the left side. Here, a CBIRs contacts the enrolling process in order to indicate that it wishes to enroll. Hereby it already transmits all the information about its configuration. The enrolling process is now in possession of all details about the CBIRs configuration. These configuration details are forwarded to the knowledge database. The CBIRn can take advantage of this information pool for every necessary purpose - e.g. query forwarding and merging. This step concludes the whole enrollment process. As previously mentioned, this is the most elementary case that can occur for the enrolling process.

The uncooperative Enroll Process

In the uncooperative enrolling process, as in the cooperative process, the CBIRs sends a requests to join the CBIRn. Yet it does not offer any details about its configuration. This scenario appears to be quite probable for the reasons given in the last paragraph. Nevertheless, it is possible to estimate the CBIRs' configuration. As mentioned in the preceeding paragraph, in order to join the CBIRn, a CBIRs has to send a request to join the service of the CBIRn, which is responsible for the enrollment of the CBIRs in the CBIRn. This service is called the enrolling service. After having received the uncooperative enroll request, the enrolling service queries the CBIRs for characteristic input images. The CBIRs hereon calculates a similarity, called a similarity score, for its images respective to the input and returns result sets of the most similar image sets. Based on the images returned, an estimation of the CBIRs' configuration takes place. The uncooperative enrolling process concludes by sending the estimated CBIRs' configuration to the knowledge database.

In the following paragraph a formal definition of the problem is given. Let S_α be the set of images I that are stored in a CBIRs α, and n_{S_α} be the number of images stored in α. In this case the definition of S_α would be:

$$S_\alpha = \{I_{1..i} | i \in \{1..n_{S_\alpha}\}\} \; with \; n_{S_\alpha} \in \mathbb{N} \tag{2.5}$$

Each image I of S_α is represented internally by a set of feature vectors $Rep_\alpha(I)$. Let F_α be the set of features used by α, $|F_\alpha|$ be the number of features used by α and $f_\mathrm{j}(I)$, with $f_\mathrm{j} \in F_\alpha$, be the j^{th} feature vector of image I. Then the definition of $Rep_\alpha(I)$ is as follows:

$$Rep_\alpha(I) = (f_\mathrm{j}(I))_{j \in \{1..|F\alpha|\}} \tag{2.6}$$

In contrast from now on in formulas the enrolling service is referred to as ω, representing an image I internally as

$$Rep_\omega(I) = (f_\mathrm{j}(I))_{j \in \{1..|F\omega|\}} \tag{2.7}$$

The fact that F_α and F_ω can be disjunct, equal, inclusive, or plain intersects leads to one conclusion. Simple reverse engineering, by trying to find the features in ω that are used by α, cannot be applied as α might use features that are unknown to ω. This might happen as proprietary CBIRs do eventually use proprietary features even though they use mostly individually weighted public features.

The only information α allocates that allows conclusions to be drawn about $\alpha's$ configuration is the similarity score (2.8) of the s most similar images. This similarity score is defined as follows: Let $score_{\alpha\,/\mathrm{Q}}(I)$ be the similarity score of α regarding a query image Q and the image I and $\Delta_{\alpha_\mathrm{j}}$ be the distance function of two feature vectors of the feature $f_\mathrm{j}()$. Let $w_{\alpha\mathrm{j}}$ be the weight assigned by α to the feature $f_\mathrm{j}()$ and $I_{\alpha/\,\mathrm{Qleast}}$ be the least similar image returned by α compared to the query image Q then the similarity score of α regarding the query image Q and I is:

$$score_{\alpha\,/\,\mathrm{Q}}\,(I) = 1 - \frac{\sum_{j=1}^{|F_\omega|} w_{\alpha\mathrm{j}} \Delta_{\alpha_\mathrm{j}}(f_\mathrm{j}(Q), f_\mathrm{j}(I))}{\sum_{j=1}^{|F_\omega|} w_{\alpha\mathrm{j}} \Delta_{\alpha_\mathrm{j}}(f_\mathrm{j}(Q), f_\mathrm{j}(I_{\alpha/\,\mathrm{Qleast}}))} \tag{2.8}$$

Depending on the similarity of two images the similarity score can take a value between 0.0 and 1.0. An image compared with itself would result in a similarity score of 1.0. But, from system to system, different normalization techniques - the formula under the fraction line in (2.8) - might be applied, but can be limited to the following established major techniques:

No Normalization Before discussing normalization algorithms, it should be noted that [130] propose means to aggregate query results without using a normalization algorithm. Those authors present result merging algorithms that do not use a score as the foundation of their calculation. Either they profit from the internal result ranking of every individual result set, or they use a fraction of the beginning of every result set and agglomerate them, or they interleave them using the similarity score with no regards to the lack of comparability as this is assumed.

Standard Normalization This method to normalize the obtained scores was proposed in [153] and [90]. The normalized score is calculated as follows.

$$normalized\_score = \frac{unnormalized\_score - minscore}{maxscore - minscore} \tag{2.9}$$

Equation 2.9 visualizes an example result set on the left side and the same result set after the transformation on the right side. The standard normalization subtracts the minimal score value that is reached by any result item of the result set from every score value of every result item. It subsequently divides this by the maximum score value that is reached by any result item. Through this method, a shift of the whole value range can be ignored as the described operation antagonizes it. Therefore it is shift invariant. As the algorithm scales every score value in a range between 1 and 0, the spreading of the score values is absorbed and therefore the algorithm is scale invariant. Nevertheless, the formula depends on the minimum and maximum values, and is therefore sensitive to outliers. If a result set was received that had items with the scores of 0.9, 0.6, and 0.2 the standard normalization would subtract 0.6 from every item resulting in scores of 0.7, 0.4, and 0.0. Eventually these scores are divided by 0.7 and result in normalized scores of 1.0, 0.57, and 0.0.

Sum Normalization This technique, proposed by [90], is an attempt to improve the outlier insensitivity of the standard normalization. It eliminates the sensitivity to the maximum score value by using the statistic sum. The sum over the similarity score values of the results not normalized is statistically more robust than the minimum. In practice the minimum score is not an outlier due to the fact that ranked lists are truncated to only return a certain number of documents. So in practice, the sum normalization is fairly outlier insensitive. It calculates scores as follows.

$$normalized\_score_x = \frac{unnormalized\_score_x - minscore}{\sum_{i=1}^{n} (unnormalized\_score_i - minscore)} \tag{2.10}$$

The sum normalization subtracts the score value of every result and, like the standard normalization, it is shift invariant. It subsequently divides the scores by the sum of all score values for result items. Therefore the algorithm is scale invariant. Even if the technique is practically insensitive to outliers it is nevertheless theoretically sensitive to the minimum in the worst case.

ZMV Normalization This transformation, proposed by [90],discards the sensibility to the minimum and the maximum score values, as it does not directly depend on either of them. The two selected aggregation statistics are the mean and the variance, as they are both aggregate and thus more robust.

$$mean = \frac{\sum_{i=1}^{n} (unnormalized\_score_i)}{n} \tag{2.11}$$

$$variance = \frac{\sum_{i=1}^{n} (unnormalized\_score_i - mean)^2}{n - 1} \tag{2.12}$$

The following formula shows the calculation of a normalized score:

$$normalized\_score = \frac{unnormalized\_score - mean}{variance - mean} \tag{2.13}$$

As this technique uses only agglomeration and no separate values, it is shift invariant, scale invariant, and outlier insensitive.

2.5.3.2 Concept Detection

In the early years of Content Based Image Retrieval, a distinction of the image domains in narrow and broad domains was defined by Smeulders in [126]. Narrow domains contain images with limited variability in their descriptive aspects. The images differ marginally, as for every image a similar setting is used (e.g. same illumination, background, perspective). [126] defined them as follows:

> *"A narrow domain has a limited and predictable variability in all relevant aspects of its appearance."*

Examples for narrow domains would be biometric passport photos, company logos, or product photos. A broad domain in contrast contains images with an unlimited, unpredictable variability. The images differ strongly, even though they might show the same concept semantically (cf. differing perspectives). [126] defined them as follows:

> *"A broad domain has an unlimited and unpredictable variability in its appearance even for the same semantic meaning."*

Examples of such broad domains would be images of buildings, forests, and of sports. The reason for this distinction was the realization that those two domain classes have to be treated in orthogonal ways. In narrow domains, analyzing algorithms can focus on the few differing aspects in the images - e.g. color of the skin, distance between the eyes, or form of the chin in the narrow domain of biometric passport photos. In contrast, for broad domains, analyzing algorithms have to take into account every possible different aspect in the images. A more effective and a more efficient analysis can therefore be performed for images of narrow domains.

In both types of domains, and those that are existing in between, a lot of research [21, 42, 144] has been accomplished and good results were achieved in different concept detection challenges (e.g. ILSVRC2011 & ILSVRC2010, FRGC [102]), i. e. Borth et al support the user in [14] with the process of a concept-to-query mapping. *Lookapp* is a system for the interactive construction of web-based concept detectors. The system provides mechanisms to find a proper query formulation for a given concept, by means of expanding the initial query with automatically assigned keywords and the inference of a corresponding category.

The detection of semantics in broad image domains becomes even more challenging with the occurrence of several concepts in a single image. There is hence a varying interpretation from the user's perspective. A stand-alone thesaurus could model these interdependencies but is rather large in size and might contain too much redundant information to be of practical use. To render CBIR possible, [53] presents a hybrid image retrieval system which incorporates statistical algorithms for semantic grouping in the concept space through relevance feedback in the image space. Aside from learning the semantic relations among concepts, the system can even adopt the user's search habits incrementally and thus improve the performance of subsequent retrieval tasks. [13] presents probabilistic topic models, providing both a predictive model of future text and a latent topic representation of the corpus.

2.5.3.3 Collection Fusion

In the following section different approaches to overcome the collection fusion problem are discussed. The authors of [146] defined the collection fusion problem as follows:

> *"The goal of a collection fusion technique is to combine the retrieval results from multiple, independent collections into a single result such that the effectiveness of the combination approximates the effectiveness of searching the entire set of documents as a single collection."*

In order to give an overview of the existing collection fusion technique, the following section is subdivided into the discussion of combination techniques, weighted combination techniques, Borda Fuse Voting Model, Shadow Document Method, Raw Score Merging, and the Round Robin phenomenon.

(A) Combination Techniques

The most simple, popular, and effective result aggregation algorithms to date were presented in [44] and further investigated in [69]. These techniques, namely CombMIN, CombMAX, CombSUM, and CombMNZ differ only in the ranking of the resultant items which have one or more duplicates in other databases. An explanation of every algorithm in particular, which will be given in the following subsections, will outline where these differences originate.

CombMIN The following formula describes how scores are associated with the documents using CombMIN. In the formula, x stands for the document and n for the number of involved databases. The method $score_i$(x) stands for the score of a document x associated by database i.

$$score(x) = \min\{score_i(x) | 0 < i < n \wedge \exists score_i(x)\}$$

The formula can be summarized as follows. Every document is associated with the smallest score value given by any of the databases involved (see figure 2.7).

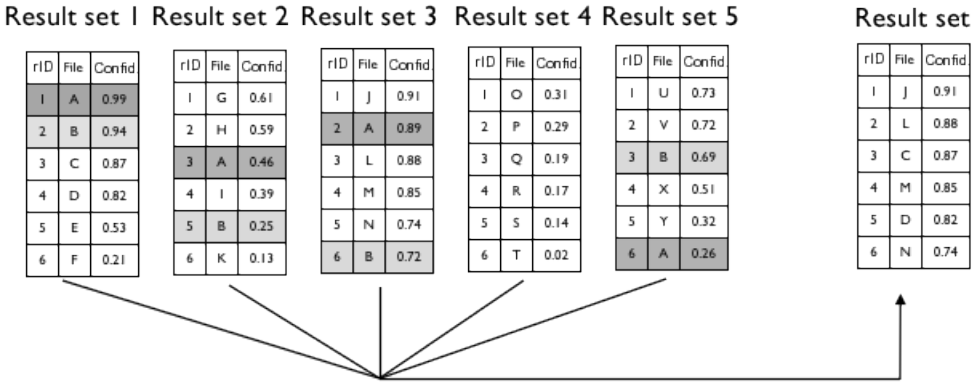


Fig. 2.7.: CombMIN Example

CombMAX The following formula describes how scores are assigned to the documents using CombMAX. The variables are, now and in the following subsection, defined similarly to the

variables in the formula of the precedent subsection.

$$score(x) = \max\{score_i(x) | 0 < i < n \wedge \exists score_i(x)\}$$

In contrast to CombMIN, in CombMAX every document is associated with the biggest score value given by any of the databases involved (see figure 2.8).

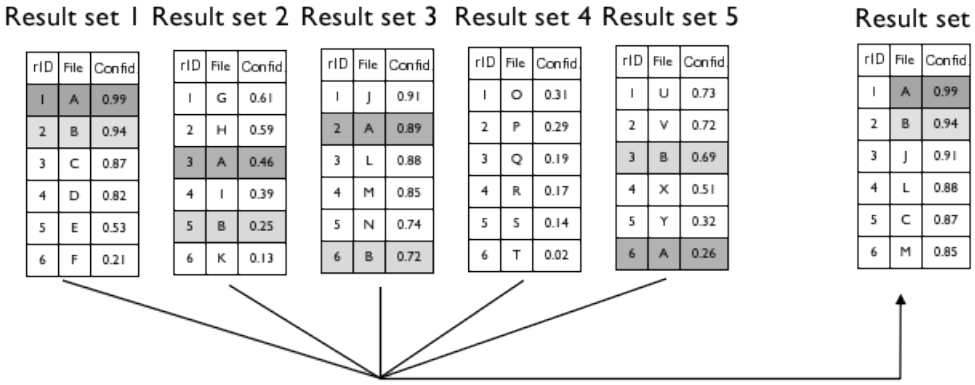


Fig. 2.8.: CombMAX Example

CombSUM The following formula describes how scores are allocated to the documents using CombSUM.

$$score(x) = \sum_{i=1}^{n} score_i(x)$$

Every document is associated with the sum of the score values given by all of the CBIRs involved (see figure 2.9).

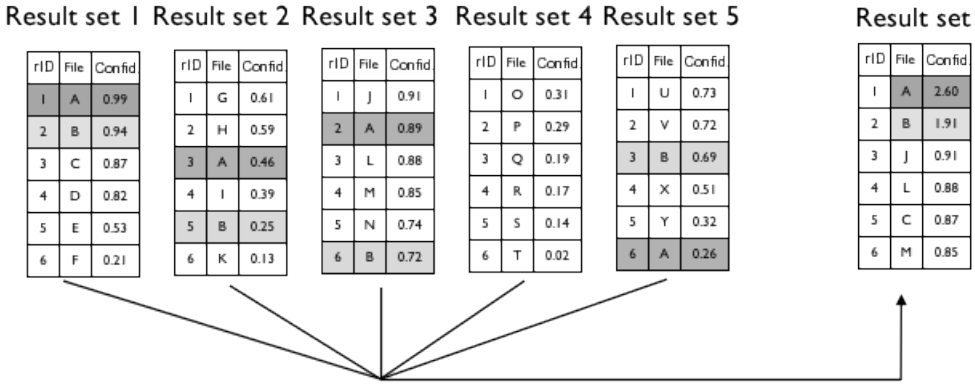


Fig. 2.9.: CombSUM Example

CombMNZ The following formula describes how scores are associated with the documents using CombMNZ.

$$score(x) = \sum_{i=1}^{n} score_i(x) * \sum_{i=1}^{n} \{1 | \exists score_i(x)\}$$

Every document is associated with the sum of the score values given by all of the CBIRs involved - $\sum_{i=1}^{n} score_i(x)$. The resulting value is then multiplied by the number of databases which assigned a scoring value to the document - $\sum_{i=1}^{n} \{1 | \exists score_i(x)$. Figure 2.10 visualizes this formula.

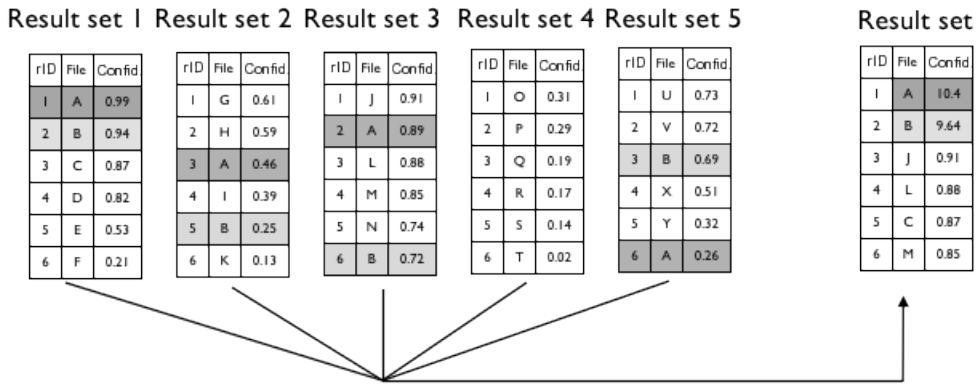


Fig. 2.10.: CombMNZ Example

Summary

In [69], it was shown that CombMNZ is the best out of these combination merging algorithms, followed by CombSum. However the problem of these combination techniques is that they rely on a lot of overlap. They could also be applied when the overlap is not as significant, however the final result list might be of lesser quality.

(B) Weighted Combination Techniques

As already mentioned in this section CombSUM and CombMNZ are the superior combination techniques. One explanation for this is given in [153], where it is the described as the chorus effect. This infers that the multiple appearence of an item in different result sets is a stronger evidence of relevance than a single appearance. Therefore [153] tried to improve CombSUM and CombMNZ.

For the calculation of the weight a modality was invented that needs no training and no additional information. For every result set the occurrence of its resultant items in other result sets is computed. A mean value of this number concerning all the databases is calculated. In reference to this mean, two thresholds are defined partitioning the databases into 3 different weighted groups.

In the following examples the sum of the occurrences in result set one to three is 11, in result set 4 it is 9, and in result set 5 it is 6. The calculated mean value is 9.6 and the thresholds are 110 % and 90 % of the mean. Therefore result sets one to three are in one partition, whereas result set four forms another partition, and result sets five comprises the final partition.

WSUM As said in section 2.5.3 the CombSum method was improved and is now called the WSUM procedure. The following calculation is used to associate the score.

$$score(x) = \sum_{i=1}^{n} (score_i(x) * w_i)$$

Every document is associated with the sum - $\sum_{i=1}^{n}$ - of the score values of the document - $score_i(x)$ - given by all of the involved databases multiplied with the weights of the databases - w_i. Figure 2.11 visualizes this formula.

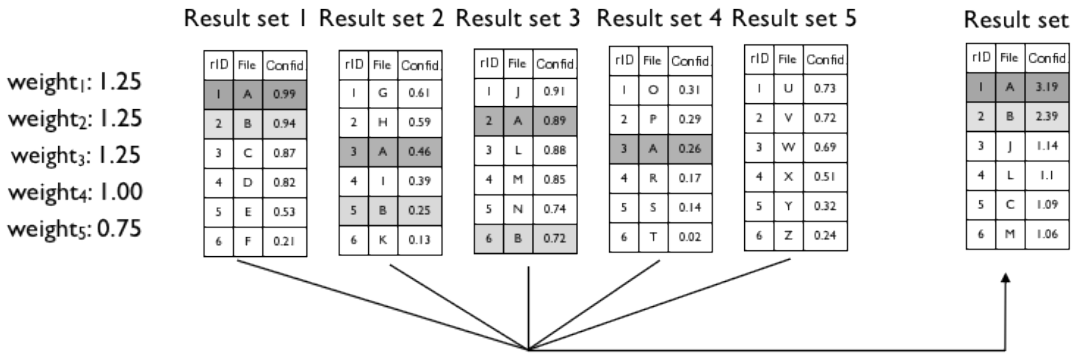


Fig. 2.11.: WSUM Example

WMNZ

As described in section 2.5.3 the CombSum method was improved and is now called the WSUM procedure. The following calculation is used to specify the WSUM score.

$$score(x) = \sum_{i=1}^{n} score_i(x) * \sum_{i=1}^{n} (w_i | \exists score_i(x)\})$$

Every document is associated with the sum of the its' score values given by all of the databases involved - $sum_{i=1}^{n} score_i(x)$. This sum is multiplied with the sum of the weight of all the databases - $\sum_{i=1}^{n} (w_i | \exists score_i(x)$. Figure 2.12 visualizes this formula.

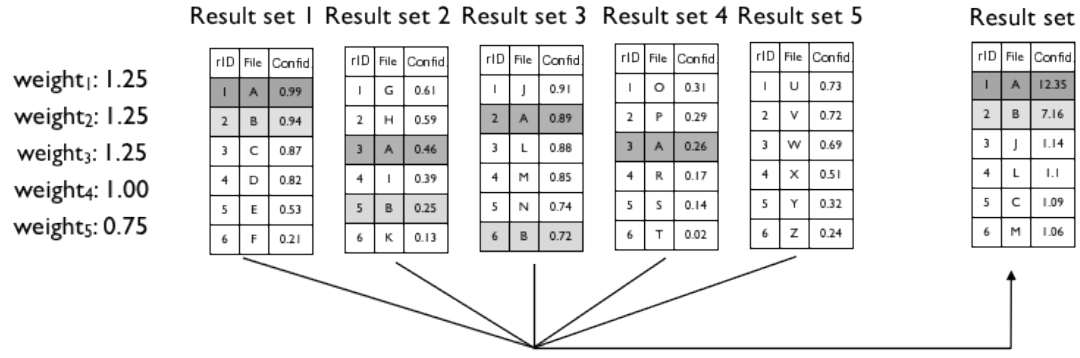


Fig. 2.12.: WMNZ Example

WCOMB Furthermore, a combination of WSUM and WMNZ, WCOMB was specified in [69]. The following formula describes how score values are associated with the documents using WCOMB.

$$score(x) = \sum_{i=1}^{n} (score_i(x) * w_i) * \sqrt{\sum_{i=1}^{n} \{1 | \exists score_i(x)\}}$$

Every document is associated with the sum of the score values given by all of the databases involved multiplied by the weight given by every individual database. This sum is multiplied with the square root of the number of databases that assigned a score value to this document to value these appearances (see figure 2.13).

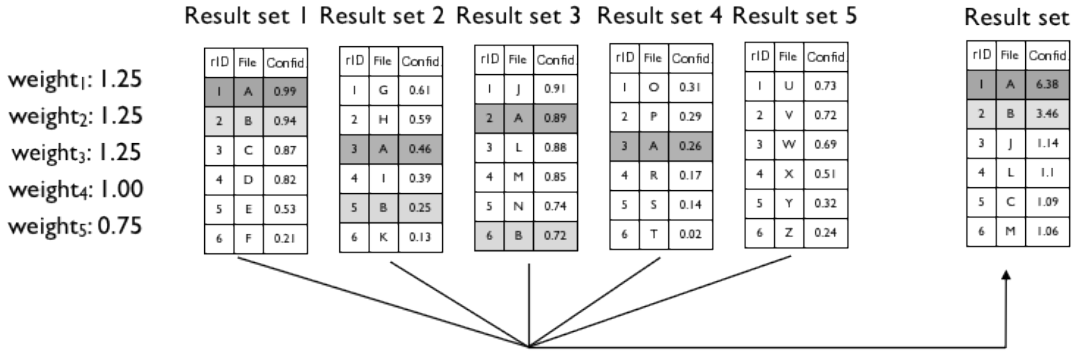


Fig. 2.13.: WCOMB Example

Summary The weighted combination techniques are almost as simple as the original combination techniques. Therefore they do not immoderately increase the computational expenses. Most of the time the results do not differ significantly, but if they do the weighted combination techniques outperform even ComMNZ [153].

(C) Borda Fuse Voting Model

This procedure was introduced for metasearch engines by [90] in 2001 and is based on democratic election strategies. In the Borda Count voting procedure used, the databases rank the resultant items according to their preferences.

Each element gets c points minus their rank points, whereas c is a predefined variable bigger than 0. These votes are summed up according to the following formula.

$$votes(x) = \sum_{i=1}^{n} votes_i(x)$$

The method $votes_i$(x) stands for the score of a document x associated with database i.

Finally all the items of the different result sets are ranked by their total points in decreasing order (see figure 2.14).

This algorithm requests only the rank information for the aggregation process. Thanks to the simple and efficient voting process the Borda-fuse voting is competitive in its performance [90].

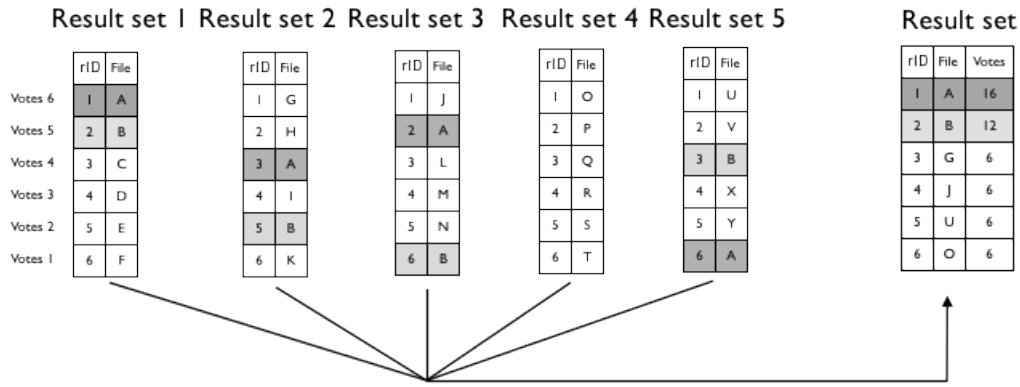


Fig. 2.14.: Borda Fuse Example

(D) Shadow Document Method (SDM)

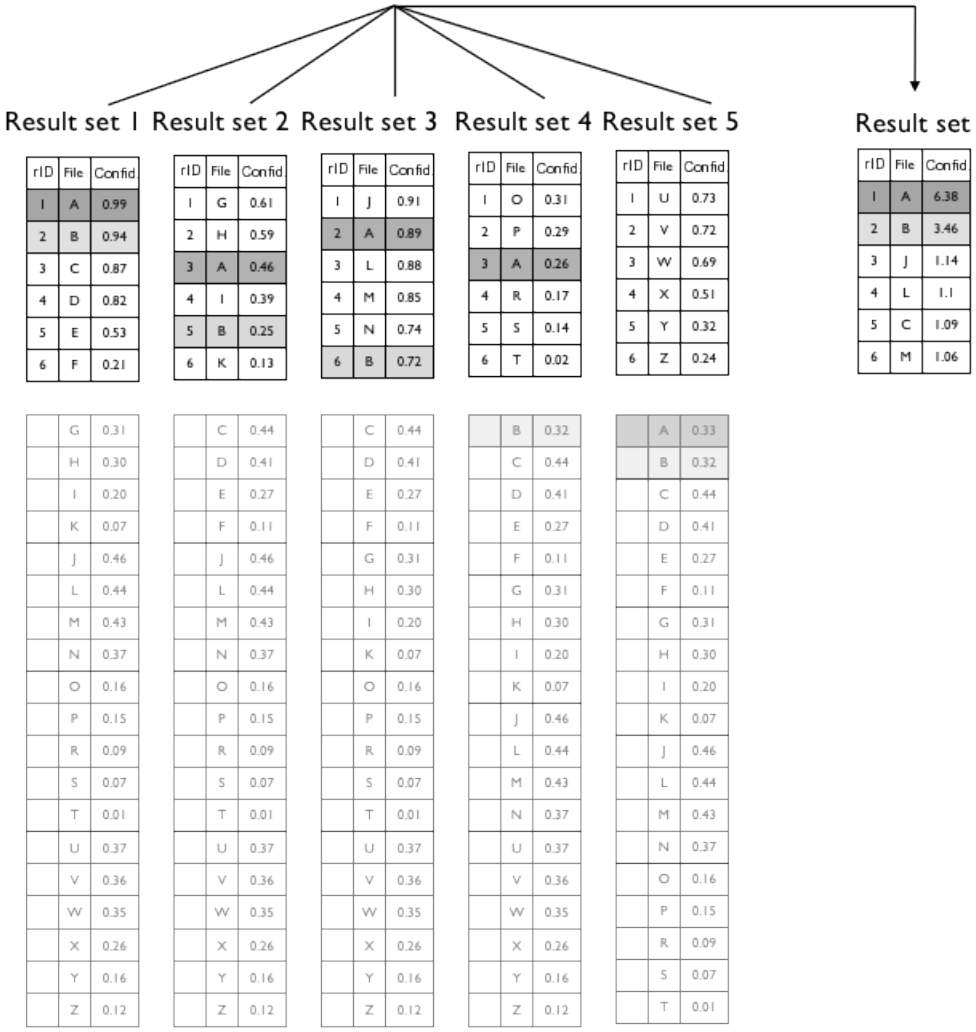
This method was proposed by [154]. The mode of its operation is as follows:

- If a document d is contained in the results of databases A and B, the score values of d from both databases are added.
- If d is only retrieved from database A and not from B, it is assumed that a shadow document of d exists in B with a score value of k. Then the score of A and B are summed up. The value of k has to be determined through empirical tests or can be derived from the degree of overlap between the two databases.

The advantage of this merging method are that no training data or downloads are necessary. The only information needed from the retrieved results are the score values of the documents. This is a formula showing the possible ways to calculate the score of a document. In this formula n is the number of participating databases and m the number of databases that contain a duplicate of this document.

$$score(x) = \sum_{i=1}^{n} score(x)_i + (n - m * k)$$

The formula is Visualized in figure 2.15 using the factor k = 0.5.



rID	File	Confid
1	A	0.99
2	B	0.94
3	C	0.87
4	D	0.82
5	E	0.53
6	F	0.21
	G	0.31
	H	0.30
	I	0.20
	K	0.07
	J	0.46
	L	0.44
	M	0.43
	N	0.37
	O	0.16
	P	0.15
	R	0.09
	S	0.07
	T	0.01
	U	0.37
	V	0.36
	W	0.35
	X	0.26
	Y	0.16
	Z	0.12

rID	File	Confid
1	G	0.61
2	H	0.59
3	A	0.46
4	I	0.39
5	B	0.25
6	K	0.13
	C	0.44
	D	0.41
	E	0.27
	F	0.11
	J	0.46
	L	0.44
	M	0.43
	N	0.37
	O	0.16
	P	0.15
	R	0.09
	S	0.07
	T	0.01
	U	0.37
	V	0.36
	W	0.35
	X	0.26
	Y	0.16
	Z	0.12

rID	File	Confid
1	J	0.91
2	A	0.89
3	L	0.88
4	M	0.85
5	N	0.74
6	B	0.72
	C	0.44
	D	0.41
	E	0.27
	F	0.11
	G	0.31
	H	0.30
	I	0.20
	K	0.07
	O	0.16
	P	0.15
	R	0.09
	S	0.07
	T	0.01
	U	0.37
	V	0.36
	W	0.35
	X	0.26
	Y	0.16
	Z	0.12

rID	File	Confid
1	O	0.31
2	P	0.29
3	A	0.26
4	R	0.17
5	S	0.14
6	T	0.02
	B	0.32
	C	0.44
	D	0.41
	E	0.27
	F	0.11
	G	0.31
	H	0.30
	I	0.20
	K	0.07
	J	0.46
	L	0.44
	M	0.43
	N	0.37
	U	0.37
	V	0.36
	W	0.35
	X	0.26
	Y	0.16
	Z	0.12

rID	File	Confid
1	U	0.73
2	V	0.72
3	W	0.69
4	X	0.51
5	Y	0.32
6	Z	0.24
	A	0.33
	B	0.32
	C	0.44
	D	0.41
	E	0.27
	F	0.11
	G	0.31
	H	0.30
	I	0.20
	K	0.07
	J	0.46
	L	0.44
	M	0.43
	N	0.37
	O	0.16
	P	0.15
	R	0.09
	S	0.07
	T	0.01

rID	File	Confid
1	A	6.38
2	B	3.46
3	J	1.14
4	L	1.1
5	C	1.09
6	M	1.06

Fig. 2.15.: Shadow Document Method Example

(E) Raw Score Merging

This method is a simple approach. It is based on the hypothesis that the same search model is used in all databases and thus the score values returned are comparable. However, since this is almost never the case, this technique is seldom used [11]. A handling of duplicates does not take place. So if there are any duplicate elements in the individual results, they will also occur as duplicates in the merged result (see figure 2.16).

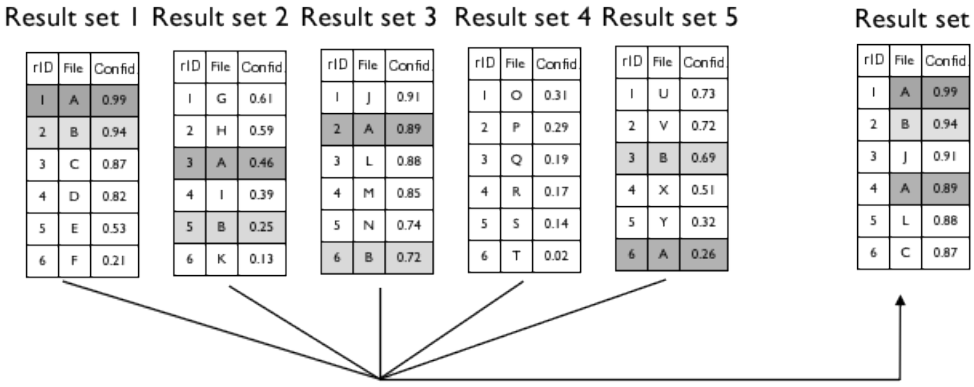


Fig. 2.16.: Raw Score Merging Example

(F) Round Robin

This algorithm is one of the most well-known approaches and it is quite simple [10]. A handling of duplicates does not take place. So if there are any duplicate elements in the individual results, they will also occur as duplicates in the merged result. The algorithm works in a straightforward manner: the final result organizes the items in rank order. The first element of a chosen individual result list becomes the first element of the final result. Then follows the first element of another result set, and so forth, until the first elements of all individual result sets are added to the new list. Now the second elements will be added.Thus, the Round Robin algorithm works well if there are no duplicates. It also performs adequately if the individual result lists from the databases contain an almost equal amount of elements and if the result lists have an almost equal score range. As this is almost never the case, in the configuration of this thesis the algorithm would fail to produce the desired result (see figure 2.17).

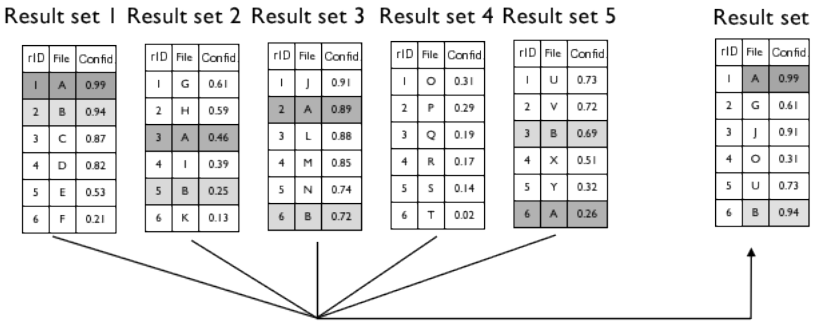


Fig. 2.17.: Round Robin Example

(G) Conclusion

After reviewing all of these merging techniques, the CombMAX (see page 34) Algorithm was selected to be applied. This merging technique promises to be an efficient and computationally inexpensive solution. Taking into account that the normalization problem is considered to be solved, the results are expected to be very good.

2.5.4. Distributed CBIR Systems

The most prominent field of application for distributed retrieval in CBIRs is obviously the Content Based Image Retrieval in the Internet. There are a number of applications that exist in the Internet attempting to make the enormous mass of images searchable. Due to the magnitude of the images, these applications must use some kind of distributed scenario. One prominent application that works using a distributed scenario is Google's image search. In the following implementations, scientific approaches that engage the distributed retrieval in CBIRs are discussed.

In [122], a peer-to-peer CBIRs network is designed that was inspired by the Metric Social Network (MSN). the principles of social networking support the self-organization and self-adaptability of the network. The interconnection of peers in the manner of a social relationship is due to the ability of peers to gather knowledge about previously answered and sent queries. An adaptive query routing algorithm exploits the query history to choose the most promising peers and thus forwards the query accordingly. The suggested IR-Network is verified using a real network consisting of 2,000 peers, containing descriptive features of 10 million images from the Flickr Photo Sharing system. Nevertheless compared to this thesis's approach, [122] does not consider the semantic aspect of images in CBIR.

In [158], it is investigated how to enhance semantic relevance in the retrieval process by semantic feature extraction and relevance feedback. In addition, a prefetching and scalable image delivery is proposed to reduce the network latency in the retrieval process and to adapt the retrieval process to the network bandwidth and the capabilities of the user device. So [158] focuses more on the join optimization of semantic relevance and retrieval latency. This thesis' approach in contrast does not take into account user feedback and it merely concentrates on the retrieval performance and efficiency.

2.6. Summary

This chapter presented the fundamentals of computerized Information Retrieval from its conception to now. It moreover highlighted in each section the innovations that are relevant for this thesis. To lay the foundations for the further discussion this chapter has subsequently discusssed information retrieval approaches and their theoretical background that were developed in the 19^{th} and 20^{th} centuries. It then outlined current web retrieval strategies and web image retrieval strategies focusing on their relevant underlying theory. In its subsequent sections, relevant current developments in Content Based Image Retrieval, e.g. of Low Level Features, were discussed. Hereafter, in section 2.5.1 the historical beginning of the fundamentals of distributed retrieval in CBIR were presented. The scientific terminology of distributed retrieval in CBIR approaches and their theoretical background were outlined in the following two sections in order to make the later proposed approach easily comprehensible. The theoretical background encompasses the standard of CBIR meta language *MPQF*, the applied

enrolling process, the used concept detection environment *WEKA* and the applied collection fusion method *CombMAX*. In its concluding section, the current developments of distributed retrieval in CBIR were discussed and compared with this thesis approach.

3. MeRRSe: Merge Returned Result Sets

In recent years a variety of search engines have emerged that make it possible to search the internet for every possible type of content. On the one hand, some of these search engines are, due to their specialization or operating size, merely aware of subnets of the Internet. On the other hand some search engines withhold results, for example due to personalized queries. It is therefore desirable to query multiple search engines.

In order to perform a search like this automatically, the merging of responses from multiple heterogeneous CBIRs responding to the same query has to be addressed. This merging is referred to as the collection fusion problem. The responding CBIRs use each of their individual search measures to compute their results and therefore the returned result sets are usually incomparable. To solve this problem in the environment of distributed CBIRs, this chapter of the thesis develops a new algorithm that deploys the first algorithm - respective to the state of the art (chapter 2) - the three steps of normalization, merging, and filtering. The originality of the idea is hereby provided by the additionally appended filtering step.

The feasibility of the approach is demonstrated by the implementation of a prototype and its evaluation using 7200 pictures whereas 72 images were similar to each other. In these preliminary tests 20 test runs were conducted. Hereby an augmentation of the number of returned images that are of the same semantic concept as the input images is achieved by up to 12 % compared to a solely application of a CombMAX.

3.1. Prologue

The first part of this thesis discussed the state of the art conditions of the scientific domains that were important for this thesis. The second part discusses in detail the approach that was developed in the course of this thesis. It starts therefore with an overview of the approach in this prologue. This overview is subdivided into two sections. The first section comprehensively states the problem that this thesis focuses on. Subsequently the second section discusses the solution developed to deal with the antecedent stated problem.

Problem Statement

In this day and age, if a user is involved in a process that requires images and s/he has no adequate images by hand what would they do? They would most likely take advantage of a search engine that operates via keywords in order to acquire the image. For what reason does this occur?

The results of current commercial Content Based Image Retrieval (CBIR) engines[1], which have mostly emerged in the last three years, are quite often unsatisfying [19]. As already discussed in subsection 2.4.3 they are unsatisfying due to the search engine's inability to understand the needs of the user and the semantic content of the multimedia files. Therefore [64] and

[1] e.g. Google (`http://www.google.com`), Bing (`http://www.bing.com`), Baidu (`http://www.baidu.com`), Fotolia (`http://www.fotolia.com`)

[139] outlined recent research attempts as well as recent commercial efforts concerning Content Based Image Retrieval systems (CBIRs). In contrast to current keyword driven search engines, they deliver, independent from the textual context, visually similar pictures to the search picture [64]. However, they still suffer from the semantic gap. Nevertheless when such a search engine tries to specify a universal solution that allows users to search all image domains simultaneously the results were not satisfying. Yet by dividing the whole image domain into narrow and broad search domains [126], and by specializing CBIRs to search the narrow domains (exemplarily presented in [68, 102]) the problems of a universal search strategy were bypassed and very good results were obtained.

Therefore, to bring these heterogeneous specialized CBIRs to work together, this thesis proposes a semantic *semantic **C**ontent **B**ased **I**mage **R**etrieval **n**etwork (**CBIRn**)*. *Semantic* signifies in this context that the search effort of a plurality of CBIRs, which are each specialized on individual semantic concepts, is combined in a single ***CBIRn***. This system, from now on known as a *semantic **CBIRn***, will not solve the problem of the semantic gap, but allows users to search the mass of images that are available online more efficiently. In order to realize the approach explained in the precedent subchapter the following research questions, which have already been outlined in detail in chapter 1, had to be addressed:

- How can the semantic analysis and modeling of a CBIRn be implemented?
- How can the configuration of a CBIRs, that should be incorporated in the *semantic **CBIRn***, be detected?
- How can inquiries in a *semantic **CBIRn*** be exclusively forwarded to relevant CBIRs?
- How can the results of similarity searches of individual CBIRs be merged in a *semantic **CBIRn***?

Solution Strategy

As discussed in chapter 2 many efforts have been undertaken to narrow down the semantic gap with moderate success. Recently CBIRn strategies were enhanced by developing new protocols, distributing content in a more advantageous way, and applying different design paradigms, and through extensions of all the achievements that were made in these research fields. As previously stated, to bring these heterogeneous specialized CBIR to work together, this thesis proposes a *semantic **CBIRn***. The approach of this thesis is divided into three sub-approaches, each framing a particular work process. These sub-approaches, color coded in figure 3.1, are called ***Me**rged **R**eturned **R**eslut **Se**ts* (***MeRRSe***), ***Ge**t **C**BIRs **Co**nfiguration* (***GeCCo***) and ***Que**ry **Di**stribution (***QueDi***)*.

Each one of these approaches addresses one of the research questions that were stated in chapter 1. The ***GeCCo*** sub approach (color coded green) addresses the research question of the analysis of external and therefore unknown CBIRs that should be integrated in the network. Furthermore, it transfers the gathered information about the CBIRs to a central unit of the CBIRn for further disposition. This central unit is embedded in the ***QueDi*** approach which is color coded red in the diagram.

It is the responsibility of the ***QueDi*** approach to analyze the incoming query image and forward it to the CBIRs considered to be the best choice. Additionally the ***QueDi*** approach determines in a training unit which configuration is best for the registered semantic concepts.

Thereby the ***QueDi*** approach addresses the research question of how inquiries can be exclusively forwarded to relevant CBIRs in a *semantic* ***CBIRn*** and how a *semantic* ***CBIRn*** has to be specified. Eventually the remaining research question is answered - how the results of individual CBIRs can optimally be merged - as the gathered result sets are merged and returned by the ***MeRRSe*** approach, color coded yellow.

In the progress of this section each of these colored areas has one dedicated chapter. These chapters are to be seen as context independent until chapter 5.5 eventually combines all of these sub-approaches to present a holistic view of this thesis's approach in its entirety.

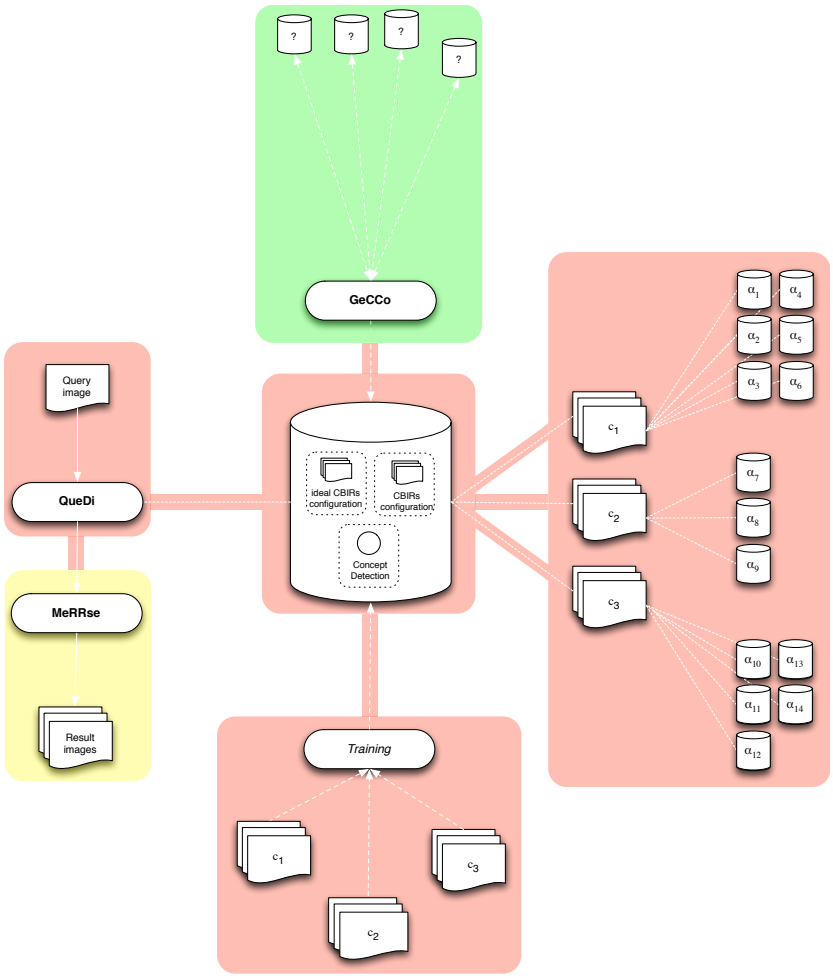


Fig. 3.1.: This Thesis' Solution Strategy

3.2. Introduction

It was already elaborated in chapter 2.4 that CBIRs reply to queries by returning result sets consisting of a ranked list of result items that are, in the context of this thesis, assumed to each be enriched by a query based score. As these scores are most commonly not comparable from the beginning, section 2.5.3 discussed the theoretical background of the currently available collection fusion approaches. Consequently this chapter introduces a collection fusion approach that uses different techniques to those developed by the earlier approaches.

This approach (see figure 3.2) performs the successive steps that are required to compute a single consistent results list from the incoming result sets of different data sources. It is assumed that this approach deals with typical multimedia similarity-based query results, where each item is part of a single result list and will be associated with a similarity score. In order to make the similarity scores returned by heterogeneous data stores comparable, they are normalized. Therefore a merging technique is applied, which makes use of the similarity score of items that show up in multiple result sets and recalculates a new similarity score for each image. It subsequently re-ranks the item in a newly generated global result set. Eventually a filter carries

out a re-ranking on the global result set, using a global search measure. This re-ranked global result set can finally be sent back to the client.

The remainder of this chapter is organized as follows: Section 3.3 and 3.4 highlight the methodology and the outline of the developed approach. Section 3.5 defines the settings of the evaluation and will analyze its results. Finally in section 3.6, this chapter is concluded and possible future adaptions are discussed.

3.3. Methodology

This chapter's approach is visualized in figure 3.2. The images that are applied in this figure are images from an original test scenario of ***MeRRSe***. The approaches' steps are explained one by one in the course of this section. The next paragraph of this section thus starts with an elaboration of the distribution process of an input image to multiple CBIRs. Even though this step is not directly interconnected to the merging process it is included in this figure. This is due to the fact that it is obvious that in ***MeRRSe*** each CBIRs is queried with the same input image. In the subsequent paragraph, the ranking process of the CBIRs and the returned result sets are closely examined. The discussion of this step forms the basic understanding for the merging process that will take place in a later step of the ***MeRRSe***. Hereby, the importance of the existence of duplicates especially is discussed. Subsequently the applied regression analysis is assessed. Eventually the subsequent reordering is elaborated by applying CombMAX on the newly calculated global ranks using a global search measure.

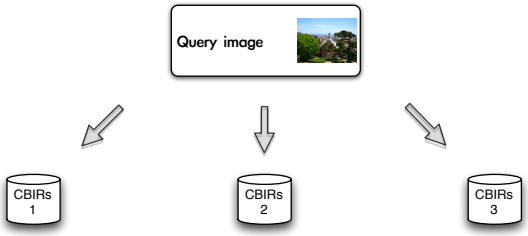


Fig. 3.3.: Query Distribution

In the beginning of this chapter's approach, as visualized in figure 3.3, a query image is distributed to diverse CBIRs. The methodological description of this approach refrains from the details of the complexity of the technical elements of query distribution except for the following statement. CBIRs might differ very strongly from each other in the communication protocol they are using, the way they are calculating their result sets, and even the returned result sets. However, as discussed in section 2.4.3, several standards can be applied as protocol wrapper middleware, and therefore in this thesis we can abstain from these discrepancies. As visualized in figure 3.3, by omitting these discrepancies, the input image can be sent to the CBIRs directly. This is the query distribution approach that is applied in this chapter's approach. A more detailed specification of this step will follow in the outline of the approach in section 3.4.

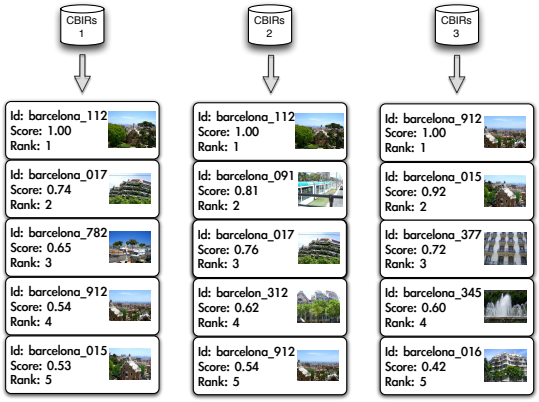


Fig. 3.4.: Responding CBIRs

The following step, visualized in figure 3.4, shows the diverse CBIRs responding to the previously forwarded query image. It is hereby assumed that each CBIRs applies distinctly weighted features and metrics to calculate a ranking that appears best for its individual application area. This assumption is due to the

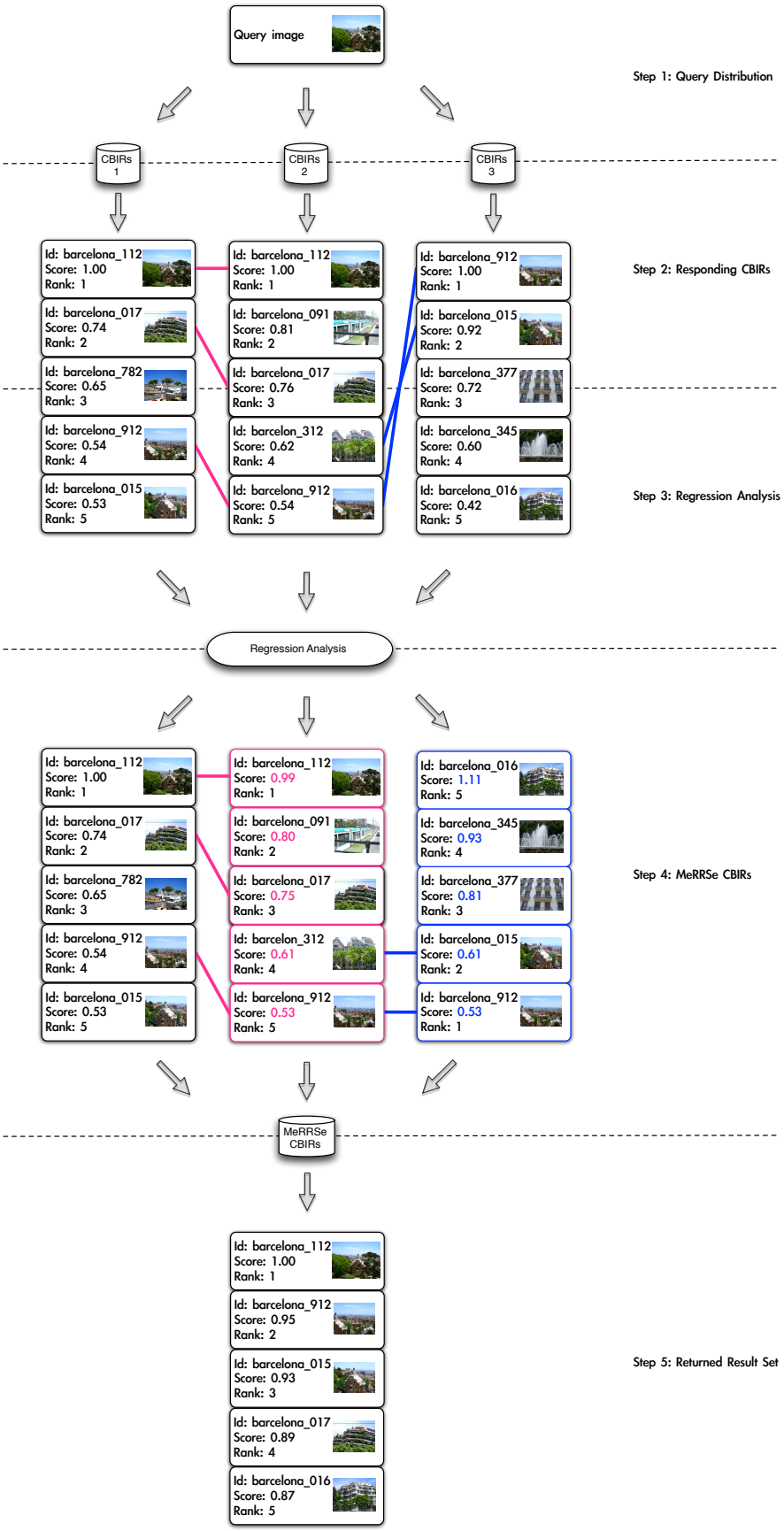


Fig. 3.2.: The *MeRRSe* Approach

deliberation that every operator ofa CBIRs tries to optimize its CBIRs as best to its specialized application area as possible. Examples for specialized application areas would be image search on specific elements of buildings or plants.

In the application area of searching building images, CBIRs would focus more likely on edge focused features. In contrast, in the application area of searching plant images, a CBIRs would concentrate more on color or shape focused features. But despite the many variations of CBIRs that exist, there are certain characteristics that remain equal over the whole range. It is assumed in this thesis that every CBIRs, after the search process is finished, returns a result list of ranked result items - which is coherent with the behavior of most CBIRs. It is furthermore assumed that each of these result items features an id, a score, and a rank that is derived from the comparison of the input image with the result items image.

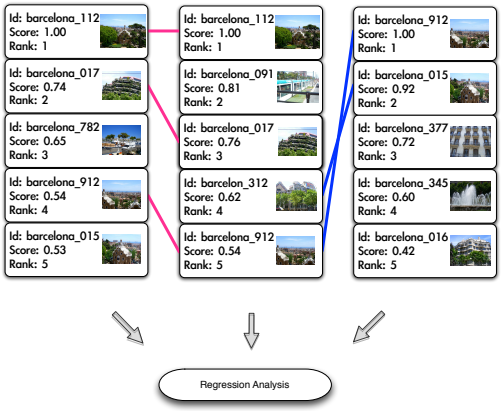


Fig. 3.5.: Regression Analysis

Considering that the state-of-the-art research CBIRs like Lire [77] satisfy these assumptions, they would appear to be reasonable. Of major importance for the approach of this chapter is the fact that the intersection of these returned result sets is defined as being not empty. As this intersection can be set at will in this thesis' approach, this assumption can be considered satisfied. A more detailed specification of this step will follow in algorithm 1 in the approach outline in section 3.4.

Using the returned result sets a linear regression analysis is conducted. This is carried out to straighten out the distinctions in the ranking processes that take place in the distinct CBIRs. This is executed by using the previously mentioned duplicates as anchors. In the beginning of the regression analysis, each result set is searched for duplicates. In figure 3.5 these duplicates are connected by pink and blue lines. In this first implementation of the approach the scores of these duplicates are subsequently used for a linear regression analysis. Linear regression was used due to the fact that the score is calculated as stated in equation 2.8 as weighted linear combination. It is furthermore assumed that for each CBIRs duplicates are existing for at least one other CBIRs. The linear regression analysis subsequently takes place between the two CBIRs that have the most duplicates in common at that point.

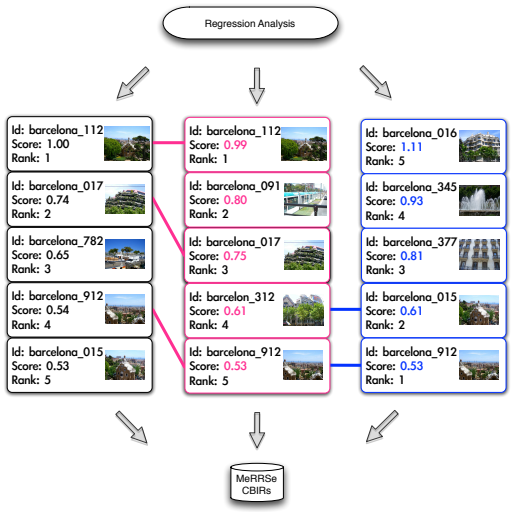


Fig. 3.6.: MeRRSe CBIRs

After calculating the regression of the scores of the duplicates, a recalculation of the scores for the images of the two involved CBIRs without duplicates is performed. As a result, a chain of CBIRs that have already been regressed is built and the scores of every image become comparable. A reordering of the images consequently takes place and the scores are normalized. A more detailed specification of this step will follow in algorithm 1 in the approach outline in section 3.4. After the linear regression analysis has been performed - visualized in figure 3.6 - the result sets of one CBIRs stays exactly the same, while

the scores of the other CBIRs are modified to correspond to the CBIRs they were regressed with. These returned result sets are processed by a ***MeRRSe*** internal CBIRs.

The ***MeRRSe*** internal CBIRs is a CBIRs that is created and applied in order to rearrange the best result items in a uniform way. Therefore, at first, result sets are used as attached images to the ***MeRRSe*** internal CBIRs, which is then queried for the input image, and the individual result items are eventually re-ranked based on their score to one single list of result items. Secondly a predefined number of result items, in this case 30, are taken from the newly constructed list. 30 result items were used due to reason that less than 7 % of the people that search the web look at more than 30 result items. This number of result items is processed by an internal CBIRs and a new ranking is calculated.

In this method, discrepancies that were not wiped out by the precedent linear regression analysis are finally settled and a new global ranking can be calculated. This ranking might strongly differ from the previously received list of result items. The previously mentioned number of predefined result items is selected higher, so that in a new ordering step erroneously retrieved images can still be sifted out. A more detailed specification of this step will follow in algorithm 1 in the approach outline in section 3.4. In a final step, visualized in figure 3.7, the previously derived result set is returned from the ***MeRRSe*** system to the client. The MeRRSe system returns the re-ranked result set with the additional information of which source it came from. Originally, in figure 3.3, the three CBIRs were queried for an image of a specific house which is located in Park Güell in Barcelona. In figure 3.4 the same three CBIRs return result sets that reply to the query. Each separate result set contains one image of the desired building. In the merged result set in figure 3.7, the desired building is contained three times. Even the two images in the returned result set that were not of the desired house contain images of buildings by the architect Antoni Gaudí that planned the house in the input image.

Fig. 3.7.: Returned Result Set

It is therefore apparent that the quality of the search result - considering the result items from this original test scenario of ***MeRRSe*** - is enhanced as the first three images are images of exactly the building the user was looking for. A more detailed specification of this step will follow in algorithm 1 in the approach outline in section 3.4.

3.4. Approach outline

The approach that was drafted in section 3.3 will be explained on a theoretical basis in the following section. The section starts with an elaboration of the challenges that had to be dealt with when creating this chapter's approach. This elaboration is followed by a comprehensive discussion of the approaches' prerequisites. Subsequent to this discussion the detailed illustration of the MeRRSe approach is given. Eventually, the section is concluded by considerations concerning a continuous monitoring of previously analyzed CBIRs.

3.4.1. Challenges

When approaching the challenge of collection fusion, several sub-challenges have to be considered simultaneously. The first challenge was the content overlap of the involved CBIRs and the therefore necessary handling of image duplicates. The next was the problem of the normalization of scores. Finally, the problem of result aggregation was encountered and effectively dealt with. This section gives a brief introduction into this chapter's challenges.

When focusing on a set of CBIRs, as they are used in this chapter's approach, there is usually a data content overlap. These overlaps classify the CBIRs as either identical, overlapping, or disjointed. Subject to this classification, differing specialized techniques have to be applied for the normalization of scores as well as for the result aggregation. Furthermore a possible overlap causes the necessity to interpret and react to multiple occurrences of identical images, from now on referred to as duplicates. These duplicates could, on the one hand, be considered as arbitrary and therefore able to be ignored, however on the other hand they could be seen as hints of higher relevance of the duplicates' representing images. The validity of the second hypothesis is, however, highly dependent on the degree of overlap of these CBIRs' content. If the first of the previously mentioned hypotheses is considered to be true, it makes it possible to use these duplicates as anchor points to correlate the scores of the involved CBIRs. The details of this accruing strategy will be discussed in subsection 3.4.3.

When drawing attention to smaller units, from result sets to result items to be more exact, a further problem becomes obvious. When comparing a selection of the previously introduced duplicates it becomes apparent that the scores they are assigned are incomparable. Due to different scoring algorithms, CBIRs can vary strongly in the scores they assign to the same image. Therefore merging of multimedia query results usually starts with a normalization phase. There are three possible ways to correlate the scores produced by heterogeneous CBIRs. Downloading the whole files and comparing them, obtaining statistics like the score etc. for every result image from every CBIRs and comparing those, or a combination of both techniques.

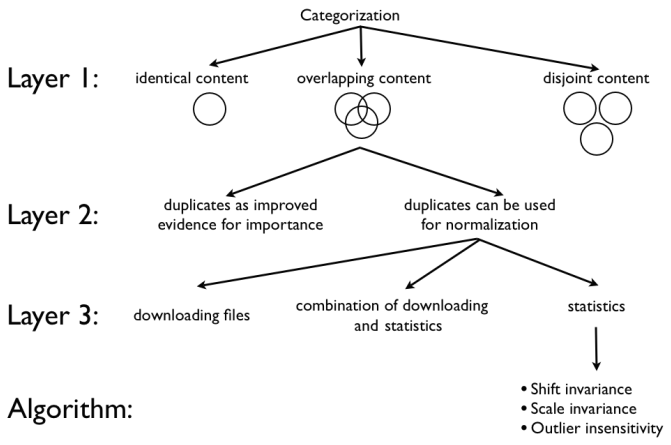


Fig. 3.8.: Categorization of Normalization Algorithms

In the case of multimedia content, neither downloading, nor a combination of downloading and core statistics is feasible. Indeed, the generated network traffic would be unacceptable.

Therefore under normal circumstances the statistical approach can be applied. As visualized in figure 3.8, and related work proposed in section 2.5.3, there are three desirable qualities for normalization algorithms: shift invariance, scale invariance, and outlier insensitivity. If a normalization scheme is shift or scale invariant this means that it is insensitive to its input being shifted by an additive constant or to being scaled by a multiplicative constant. Outlier insensitivity means that the normalization is not sensitive to the similarity score of a single result item. Therefore adding an outlier to a result set does not crucially change the normalized similarity score for the other result items. The details of this accruing strategy will be further discussed in subsection 3.4.3.

Eventually, when every other challenge has been coped with, there is the last challenge of merging the returned result sets. At this point of the approach, the duplicates are dealt with and normalization has already taken place. To visualize possible ways to go from here, figure 3.9 outlines standard categories of result aggregation techniques which were closely discussed in 2.5.3. As with normalization algorithms, distinctions are made between CBIRs containing identical, overlapping and disjointed data sets. The desired output of these algorithms is a ranked document list, sorted in descending order of relevance. This ranking is computed using a merging technique that again uses the rank or the similarity score assigned by the individual input CBIRs.

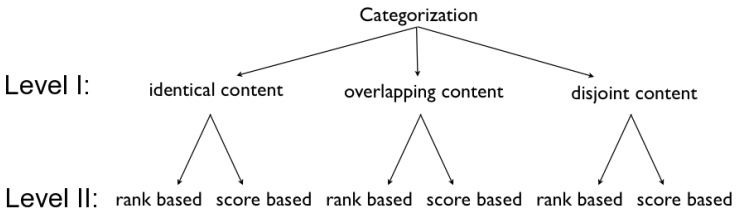


Fig. 3.9.: Categorization of Merging Algorithms

3.4.2. Prerequisites

The previously developed result set for the merging approach needs to follow certain prerequisites in order to perform in the desired way. These prerequisites encompass, among other elements, a certain amount of overlap between the CBIRs, a linear dependency between the scoring algorithms of the CBIRs, and a good performance of the CBIRs involved. In the following subsection these prerequisites will be discussed.

As pointed out in the precedent paragraph, ***MeRRSe*** needs a certain amount of overlap between the CBIRs so that there is a higher probability that duplicates show up in results sets, as this enables ***MeRRSe*** to perform properly. These duplicates are necessary to perform the linear regression analysis. Whenever a certain threshold of overlap has fallen, the approach is not usable anymore. For the implementation of this chapter's approach, the threshold was taken from empirical evaluations and defined as 30 %. This overlap has to be utilized in order to guarantee a sufficiently high enough number of duplicates of result items in the various result sets. The percentage is that high due to the small number of 30 result items, discussed in section 3.4, that are collected from every queried CBIRs in the empirical studies of this approach. By augmenting the number of collected result items, this percentage can be reduced.

A further necessity arises from the implementation of the approach. It is inevitable for the current implementation that the scoring algorithms satisfy the requirements of the calculation of the similarity score, as defined in formula 2.8. This arises from the fact that a linear regression is used in order to perform the straightening out of the scoring discrepancies for the different

CBIRs. However in further implementations, arbitrary additional regression analysis modules can be added and this limitation can be eliminated.

The final prerequisite of ***MeRRSe*** is the quality of the CBIRs involved. If the result sets returned from other CBIRs are not of good quality, ***MeRRSe*** can only perform damage containment. A specific threshold for this quality - like the minimum number of returned images with a certain similarity to the input image - is impossible to define, but it has to be clear that only when the ***MeRRSe*** approach is getting good input can it perform what it was built for. Its principal task is to merge the best result items from a multitude of CBIRs and return a single ranked list.

3.4.3. Result Merging

Based on the analysis of the state of the art conditions, an algorithm was designed that uses a pairwise disjunct linear regression algorithm as its normalization algorithm. The pairwise disjunct linear regression was chosen in the primary conception of the algorithm. This was the case due to the expectation that more accurate regression would be obtained by regressing result sets that have a higher number of result items in common. In future work however this expectation will be put to the test by implementing and testing the performance of other regression algorithms (e.g. the regularized regression [96]).

To merge results, the CombMAX algorithm [69], which ranks all the items depending on their similarity score was used, whereas items that would show up multiple times in the ranking are inserted just once at their highest possible rank. CombMAX was chosen because of its simplicity, after comparing with a multiplicity of other algorithms introduced by [69], [44], [153] and [154].

The normalizing and merging steps are present in the most common approaches. The addition of a re-ranking step to the process was proposed. This third and last step in the algorithm provokes a re-ordering of the received result items. It is not a part of any algorithm known up to now. The re-ranking reorganizes the result items with respect to a global search criterion that is available for every result item.

In the implementation of this chapter a histogram comparison, using the metric applying the p1-norm, was implemented. In this chapter's implementation, the color histogram was used, as it was available for the whole test data. The top results were expected to be rated best, but it was not expected that the top results would be in the optimal order. That is why an additional re-ranking step was added, to optimize the ranking of the top ranked items. The meta-code for the proposed approach is shown in Algorithm 1.

Algorithm 1 search(searchItem, resultSets)

Require: $searchItem, resultSets$
Ensure: $searchItem \neq null, |resultSets| > 0$

```
repeat
    (rs1, rs2) ← getResultSetsWMCI(resultSets)   // returns the two result sets that have the most common items
    regress(rs1, rs2)                            // a linear regression analysis makes result items comparable
until all result sets are normalized
items ← getNBestResultItems(resultSets, n)       // returns a set of the n best result items of all result sets
rerank(items, searchItem)                        // a reranking of the returned set of images takes place
return items
```

The Algorithm starts with a loop. After processing the similarity scores, the result sets are

normalized. The loop is carried out n-1 times, n being the number of result sets. In the first iteration the command getResultSetWMCI (resultSets) returns the two result sets that have the most common items, notably rs_1 and rs_2. Accordingly, the items of rs_1 and rs_2 are normalized by the regression method regress (rs_1,rs_2). In every following iteration, the command getResultSetWMCI (resultSets) returns two result sets. These result sets are the ones with the most common items, with the constraint that one of them (rs_1) has already been normalized and the other (rs_2) has not.

Using this strategy, the method regress (rs_1,rs_2) applies the standard linear regression analysis on the duplicates of rs_1 and rs_2. The hereby computed regression coefficients are applied to re-calculate the scores of the result items that are referenced by the result set rs_2. The duplicates' scores are re-calculated as well. A possible difference between the scores of the duplicates of one image is eliminated by using CombMAX in the merging step. By repeating the regression process, all result sets are eventually normalized and become comparable using their similarity score values. After having processed the loop, getNBestResultItems (resultSets, n) merges the items of all the result sets that have the n highest similarity score using CombMAX. These items are stored in a ranked list and after their reordering by rerank (items, searchItem) using the ***MeRRSe*** internal CBIRs, they are returned to the client.

3.5. Evaluation

The general setting of this chapter's experimental evaluation is the simulation of a set of MPQF queries, aimed at a distributed system composed of heterogeneous (in terms of query processing engines) MPEG-7 CBIRs.

A user interface module was implemented to provide a convenient test environment. It offers the possibility to define the following test case variables: image that has to be searched for; average percentage of coverage between CBIRs; number of CBIRs; and number of items per CBIRs. Another module, the test case generator, assembles a test case. To this end, a predefined set of MPEG-7 files is used. The result is stored in an SQLite database. This database is used to simulate the behavior of several MPEG-7 CBIRs.

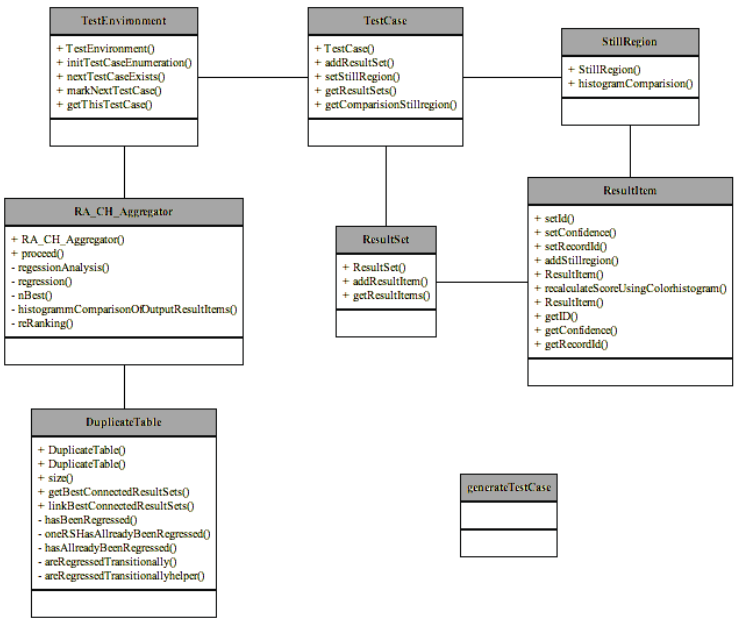


Fig. 3.10.: UML Diagram of ***MeRRSe***

Different p-norms (p=2, p=2.5 and p=3) were applied to implement metrics on the scalableColorType attribute. The MPEG-7 scalableColorType attribute was chosen due to its good retrieval performance caused by its minor complexity. They are variously used to

simulate a heterogeneous environment. Figure 3.10 visualizes the structure of the UML Diagram of ***MeRRSe***. The *RA_CH_Aggregator* is the main class for the implementation. It loads the test environment files and hence initializes the neighboring *TestEnvironment* class. The *TestEnvironment* class then loads different test cases in the *TestCases* class which were previously generated using the *generateTestCase* class. These test cases contain result sets and their result items in the corresponding classes. Additionally the *StillRegion* class contains the input image that was used by the *generateTestCase* class to create the result sets.

For the first test run, CombMAX was used as merging technique. For the second test run, which is noted in the second line of table 3.1, the result sets were pre-processed using the regression analysis and subsequently merged. For the third test run, which is noted in the third line of table 3.1, the additional re-ranking step of ***MeRRSe*** was added. As the majority of users of research engines do not consult more than 30 results, this step was applied to the best 30 results that were returned by CBIRs in the previous step. The best 30 items of the resulting merged global result set were then sent back. These results of the different test runs were then compared. As the foundation for the evaluation, the three image sets of the car, duck and the burger of [92] were applied.

The algorithm was developed and tested on a MacBook1.1. It had a processor speed of 2 GHz and a memory of 1 GB with an L2-Cache of 2 MB. The Bus speed was 667 MHz and the operating system used was Mac OS X 10.5.6 running on the kernel version Darwin 9.6.0.

Performed Steps \ Picture	*car*	*duck*	*burger*
CombMAX	+0 %	+0 %	+0 %
CombMAX & Normalization	+5 %	-25 %	-4 %
CombMAX & Normalization & Reranking	+7 %	+12 %	+5 %

Table 3.1.: Amelioration of the precision

A test set of 7200 pictures is used, whereas every set of 72 pictures are similar to each other. Having implemented the 3 previously discussed metrics, a set of 3 CBIRs is chosen to be used, each with a different metric to ensure a certain level of heterogeneity. As already discussed in section 3.4.2, 30 % of coverage between each CBIRs with at least one other CBIRs is defined as necessary. In these preliminary tests, a number of about 800 images are used per CBIRs. This allowes the CBIRs in the test cases to simulate an image base of sufficient size. On the other hand a certain variance of the images is given for the different test cases. Three different pictures with very different characteristics in terms of color, intensity, and shape (car, duck, and burger) are chosen. In these preliminary tests, 20 test runs using a manually constructed ground truth are made. The obtained improvements are shown in Table 3.1.

An improvement in all test cases for the complete test run was obtained. A search for the picture of the one colored item showed a 7 % +/- 2 % increase of the precision. The picture of a two colored item demonstrated a 12 % +/- 3 % increase, and the picture of a multi-colored item exhibited an increase of 5 % +/- 1.5 %. This means that statistically in average, from a list of 30 result items, 2 incorrect items are removed and 2 correct items are added.

An interesting aspect is that in two cases the similarity score after the normalization decreased compared to the simple merging case. This was however, an expected behavior caused by the reordering of the regression analysis which sometimes ranks images that are not relevant

to the query better than a relevant image, or ranks images better that are placed too low in the result set. This is due to the fact that the normalization is a strictly mathematical approach that tries to approximate the scorings of the result sets without taking into account the similarity to the input image. However this issue is very difficult to tackle when taking only a very small number, as only the first 30 result items were included but in general an approximation of the result sets is obtained. This actually shows the interest of the ***MeRRSe*** internal CBIRs, which reorders these mistakes.

3.6. Conclusion

This chapter presented an algorithm that aimed at the problem of collection fusion for distributed and heterogeneous multimedia CBIRs. A study of the challenges and the prerequisites initiated this chapter. Subsequently the five steps of a proposed algorithm were discussed. An initial distribution of the query image, wich is answered by a set of result sets of the requested CBIRs, is linearly regressed by ***MeRRSe***. The 30 best of each of these regressed result sets' items are subsequently allocated to a ***MeRRSe*** internal CBIRs, which is then queried for the original query image. The returned result set is eventually returned to the user. In order to assess this approach, an experimental evaluation was conducted. An improvement of up to 12 % the results was demonstrated compared to a solely application of CombMAX.

The algorithm proposed in this chapter has the advantage of being able to deal with large amounts of multimedia data as it works without a learning phase, nor reference statistics, downloading parts of the files,or prior knowledge of the component CBIRs. Moreover, it is compatible with the international standards MPEG 7 and MPQF. Planned future work includes a more detailed study of the runtime and implementation of specific measures for dealing with outliers in the result sets.

4. GeCCo: Get CBIRs Configuration

Many Content Based Image Retrieval systems (CBIRs) have been introduced in the last decade and this trend seems to be continuing. Though the search process is very similar for each CBIRs, the calculation of rankings and scores is always determined by the comparison of features (low-, mid-, high-level). Nevertheless, their respective realizations lead to different results. This is even the case when two non-identical CBIRs base their search on the same image data set. Knowledge about these internal configurations (features, weights, and metrics) would be beneficial in many usage scenarios.

This thesis specifies in the following chapter an approach that makes the automatic detection of the configuration of CBIR systems possible. It is demonstrated that the problem can partly be traced back to a vector optimization problem. Therefore, the performance of several optimization algorithms is evaluated in chapter 4.4. The feasibility of the approach is demonstrated by a prototypical implementation and its evaluation using the *ImageCLEF*[1] [24] test set. Although the approach of this chapter will eventually, in chapter 5.5, be combined with the approaches of chapters 3 and 5, the approach of this chapter is for the moment to be seen as context independent.

4.1. Introduction

As already discussed in chapter 2.4 there are a multitude of available CBIRs in research, as well as in the market economy. Even though the overall approach of all of CBIRs is mostly the same, specializations regarding their configurations are necessary. As explained in chapter 2.4, more specialized solutions promise to return better results in narrow domains than generalized approaches. Therefore, specialized solutions have attracted a lot of scientific attention in recent years. Therefore, chapter 2.4 further explained how features, metrics, and weighting work together and why, as well as how, the retrieval result profits from the reconfiguration of their constellation.

Although the specialization of CBIRs seems very promising, a problem emerges when trying to manage the sum of all kinds of images available in the Internet. For that reason there are other research approaches that avoid the specialization of their applied CBIRs. This chapter however develops an approach that applies specialized CBIRs. Therefore, it has to be considered that by definition, when specialized, a CBIRs loses its capacity to deal with the broad spectrum of images available on the Internet. A specialized CBIRs is tuned to distinguish images of one semantic concept as exactly as possible, and consequently images of other semantic concepts are not taken into consideration. Even though CBIRs basically all work the same way and, as summarized in chapter 2.4, there are communication protocols that allow us to address these CBIRs in an implementation independent way; even though programs were built that were used to construct networks of CBIRs there was so far no possibility to analyze them.

Such knowledge could allow the meta-search engines to establish a content-sensitive CBIR selection process for images. Thus, CBIRs that would not be of additional value to the query

[1] `http://www.imageclef.org/2011`

response can be ignored altogether. Furthermore, the gathered information can be used to improve the result aggregation process of the different retrieved result sets.

Consequently, this chapter proposes a novel approach for an automatic detection of the configuration of CBIRs. The detection process is based on the analysis of a small set of test queries that are executed on the CBIR system in question. It uses an optimization algorithm and filter strategies in order to identify the best feature-weight combination.

The remainder of this chapter is organized as follows: Section 4.2 and 4.3 spotlight the methodology and the outline of the developed approach. Section 4.4 defines the settings of the evaluation and analyzes its results. Finally, a conclusion and possible future adaptions are provided in section 4.5.

4.2. Methodology

This chapter's approach is visualized in figure 4.1 in order to give a brief overview of the approach in the beginning of this section. The steps of the approach are one by one explained in the course of this section. Comments on the image test set that was applied in the beginning of the approach are first provided. Furthermore, the querying processing employed by the CBIRs will be discussed. Hereby, especially the image test sets' number of contained images and the selection criteria for these images are discussed. Subsequently the internal processing of the queries, including the construction of the returned result sets, is discussed. The following paragraph elaborates the functionality of the applied filter.

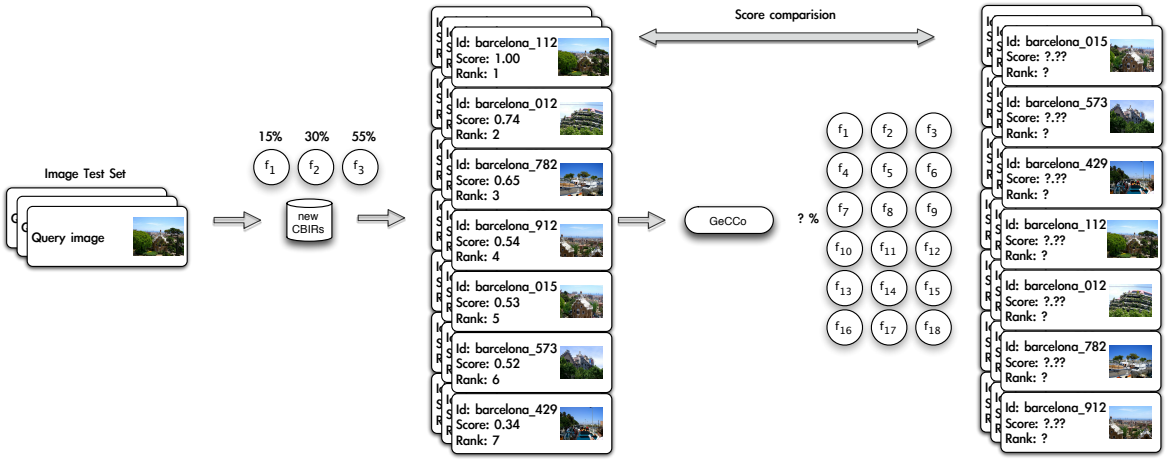


Fig. 4.1.: The GeCCo Approach

The set of test images that is employed in the approach and visualized on the left side of figure 4.1 is from now on referred to as ITS (**I**mage **T**est **S**et). In the beginning of the approach these images are used to query the CBIRs individually.

The following step shows the processing of the query images by the CBIRs that is going to be analyzed. In the preceding step, the ITS' images were sequentially sent to the CBIRS. In this step the CBIRs responds with ranked lists which consist of ranked result images and their associated scores. Hereby, images are internally processed by the CBIRs in the way that was elaborated on in detail in chapter 2.4.3. In order to make the explanation of the ***GeCCo*** approach

more comprehensive, this processing step is nevertheless integrated in the visualization of the approach and its extracts in figure 4.1. However the following visualized percentile weightings are **not** known to the ***GeCCo*** approach as they were defined while the CBIRs was instantiated and have to be analyzed by the ***GeCCo*** approach.

The features of the CBIRs, that are going to be analyzed, are visualized in figure 4.1 as circles containing $f_1$to f_3, and are extracted from each single query image. Subsequently an evaluation of the weighted sum of the distances of the input images' features and each, in the existing CBIRs image data pool, images' features takes place. The weighting of the features is visualized as the percentile value above the feature circles in figure 4.1. Based on the preceding evaluation, a ranked ordering of the CBIRs' image data pool regarding the similarity to the input image is calculated. Determined by the search paradigm used by the CBIRs, a list of possibly varying lengths for every input image is returned by the CBIRs. Those lists of images, and associated scores and ranks is referred to as a result set.

However for the analysis process, explained in the following paragraph, the varying length of the returned result sets is merely irrelevant as long as the returned number of results exceeds five items for at least three result sets. Preliminary experimentations have shown that, for the experimental setup of section 4.4, three result sets were sufficient in order to obtain meaningful results.

Using the returned result sets, a new feature weighting vector for a ***GeCCo*** internal mutable CBIRs is calculated. This ***GeCCo*** internal mutable CBIRs has implemented a multitude of features. These features are visualized in figure 4.1 as circles containing $_1$to f_{18}. In an optimal case, this set of features that is implemented by the internal ***GeCCo*** CBIRs is similar or a superset to the CBIRs that will be analyzed. The most obvious way to accomplish such an analysis is through a brute force approach.

That brute force approach would be to query every possible configuration of the ***GeCCo*** internal mutable CBIRs for each image of the ITS. The sum of images that were sent in the result sets that was visualized in figure 4.1 of the CBIRs as image data pool are used. This data compares the output of the previously explained queries to the result of the CBIRs in question. The goal is to calculate a percentile feature weighting that, when confronted with the same queries and works on a subset of the images from the original CBIRs, returned ranked result lists and scores will be similar to the ones from the original CBIRs. For the case that there is no scoring available, the same calculation can be made by an application of the ranks of the returned result sets. As this method results in a less accurate performance of the approach the scoring method was preferred. The previously briefly discussed calculation of the feature weightings is achieved by minimizing an objective function which evaluates possible weight vectors for the feature set. This is obtained by comparing the calculated scores to the result scores of the CBIR system. In this way, the problem can be traced back to a vector optimization problem and an optimization algorithm can be used to find a good weighting vector. This will preferably be one that is very similar to the one that is internally used by the CBIRs. This optimization is performed for every query in order to be able to statistically evaluate the calculated optimal weight vectors. Finally similarity score rating values are assigned to every feature according to different criteria. A weights' standard deviation across the multiple weight vectors might be such a criteria. This rating is used to filter irrelevant features. Once there are no more features to filter, the feature weight configuration of the CBIRs is calculated using the arithmetic average.

Figure 4.1 merges all the previously mentioned steps to the ***GeCCo*** approach. It is visualized that the CBIRs in question replies to every single query image of the ITS with a ranked list of

result images and their associated scores. In series, the feature weight that generates the scores is approximated. On this basis the ranking is calculated, as similar as possible to the result sets' scores. Finally similarity score rating values are assigned to every feature to enable the filtering of irrelevant features. A theoretical discussion of this approach will be provided in chapter 4.3.

4.3. Approach Outline

The approach that was drafted in section 4.2 will be explained on a theoretical basis in the following section. The section starts with the classification of the possible feature distribution when analyzing a CBIRs. Definitions of the key components of this approach will be included. This classification is followed by a comprehensive discussion of the prerequisites for the approach. Subsequent to this discussion, the detailed illustration of the ***GeCCo*** approach is presented. Eventually, the section is concluded by considerations concerning a continuous monitoring of previously analyzed CBIRs.

4.3.1. Classification of Feature Distribution

In formula 4.1 the configuration of a CBIRs α is formalized as a triple. This triple consists of a set of features F_α, a set of feature metrics Δ_α and a set of feature weights W_α. Chapter 2.4 discusses the detailed definitions of the metrics, features, and weightings, in general as well as for this thesis specifically. These sets are defined as follows (whereas $n \in \mathbb{N}$)

$$Config(\alpha) = (F_\alpha, \Delta_\alpha, W_\alpha) \tag{4.1}$$

$$F_\alpha = \{f_1, ..f_{n_\alpha}\} \tag{4.2}$$

$$\Delta_\alpha = \{\delta_1, ..\delta_{n_\alpha}\} \tag{4.3}$$

$$W_\alpha = \{w_1, ..w_{n_\alpha}\} \tag{4.4}$$

Features F_α are most likely digital representations of color, edge, or shape characteristics. This approach concentrates on CBIRs that are using Low Level Features (LLF) to compare query images with images that are stored in their system. Therefore, formula 2.6 in section shows the internal LLF representation of an image I in a CBIRs α. The image I is represented as a vector of feature vectors f_j. In the context of this thesis, the calculation of CBIRs α's score regarding the query input image I_{Input} and a stored image I are defined as follows: Let $score_\alpha(I_{Input}, I)$ be the score of α respective to a query image I_{Input} and a stored image I, and δ_{f_j} be the distance function used for a feature f_j. The score of image I regarding the query image I_{Input} is proposed to be calculated as defined in formula 2.8, as this is the method that current research is using [77] to calculate their score.

Actually fusions of the previously mentioned characteristics - color, shape, and edge - can be represented in one feature. Two CBIRs which are using the same set of features could be dissimilar as they might assign different weights, or metrics, to their features. This basically means that the configuration of CBIRs is defined by more than just features. It is defined in the context of this thesis as it was defined by formula 4.1

The basis of this approach is the exploitation of a rich set of known, by the ***GeCCo*** internal mutable CBIRs implemented, features and feature metrics. These were applied to calculate the feature distances. However, for reasons of simplification, the description of the implemen-

tation explained in this chapter focuses exclusively on the feature weight calculation that is additional to the feature detection, as opposed to the detection of the metric. Nevertheless, a metrics detection mechanism has also been integrated in the implementation and can be used by assigning multiple metrics to a feature. To be more precise, assigning multiple metrics to a feature can be rendered possible as explained in the following.

As the visualization of the version of the approach that is able to determine features, feature weightings, and metrics, Figure 4.1 has to be interpreted in the following way: Every deliberation concerning this chapter's algorithm that was carried out in section 4.2 remains the same except for those concerning the circles, marked with $f_1 to f_{18}$. In the simplified version of the approach that is discussed in this chapter, these circles each represent a single feature. In a version that relinquishes on this simplification these circles each represent a feature metric combination.

A direct consequence of this alteration in the interpretation of the algorithm's visualization is in contrast to the simplified version of the tuples of feature-metrics and feature-metric weightings. When this approach should consider metrics as well, the circle, marked with f_1, would for example be replaced by circles, marked with tuples of (f_1, δ_1) to (f_1, δ_4). The only alternation this adaption of the approach would make is that it would become more resource intensive. The calculations of features, metrics and feature weightings or features and feature weightings would be considered similarly.

For the reason of comprehensibility the calculation of features, metrics, and feature weightings was not selected for the illustration of this chapter's algorithm. The circles, marked with $f_1 to f_{18}$, are therefore under the current circumstances to merely be interpreted as features. Using the feature set these features emerged from, defined in 4.2, the internal representation of an image in the ***GeCCo*** approach is analogous to the representation of the CBIRs introduced in the methodology (section 4.2). Formula 2.7 in section shows the internal representation Rep_ω of an image I for the GeCCo approach.

Since knowing all features, especially the proprietary ones that could be used in a CBIRs, is not feasible, different classifications are considered. Five cases are possible (see also figure 4.2):

1) $F_\alpha = F_\omega$: In this case the ***GeCCo*** internal mutable CBIR implements exactly the same features that the CBIRs is using and only the weights for the features have to be found.

2) $F_\alpha \subset F_\omega$: In this case the features which are not used by α have to be identified and the weights for the remaining features have to be found.

3) $F_\alpha \supset F_\omega$: In this case the CBIR system in question uses features that are not present in the internal mutable CBIRs of ***GeCCo***. The current focus of the implementation is the detection of this case in order to avoid pretending that a configuration of the CBIRs in question could be analyzed.

4) $F_\omega \cap F_\alpha \neq \emptyset$: Not included in this case are constellations that are included in cases 1, 2, or 3. Similar to case 3, the current focus of the implementation is the detection of this case.

5) $F_\omega \cap F_\alpha = \emptyset$: In this case, as in cases 3 and 4, the current focus of the implementation is the detection of this constellation.

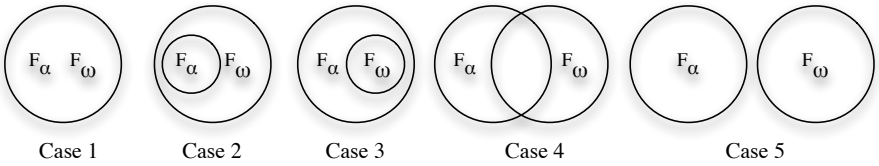


Fig. 4.2.: Distinct cases for the detection of the configurations

4.3.2. Prerequisites

The previously developed configuration detection approach needs to follow certain prerequisites in order to perform in the desired way. These prerequisites encompass, among others, a weighted linear score calculation in the CBIRs in question, sufficient knowledge about current features and metrics, as well as a result list that contains at least five images per query. In the following subsection these prerequisites will be discussed in detail.

The first prerequisite was deduced from formula 2.8. This prerequisite states that CBIRs could be exclusively used for the approach of this chapter that base their score calculation of their ranked scoring list on a linear combination of weighted feature distances. Therefore, for instance CBIRs that base the calculation of their ranked scoring list on exponential combinations could not be dealt with in the course of this chapter's approach. As this scoring appears to be the scoring that is used by state of the art research CBIRs [77], this is not considered a big restriction.

As pointed out in the precedent section the internal mutable CBIRs of ***GeCCo*** uses features and weightings in order to calculate a configuration that is similar to that of the CBIRs in question. ***GeCCo*** achieves this through the approximation of the score calculation algorithm of the CBIRs. For this approximation, features that are as similar as possible to the originally used by the CBIRs in question are necessary. Therefore, sufficient knowledge about the implementations of conventional features and metrics is indispensable.

The final prerequisite is that every CBIRs in question replies with a result set of at least five images for each of the three queries. As discussed in detail in the second paragraph of section 4.2 it has been empirically evaluated that for this thesis' experimental setting the range of five result images is broad enough to analyze every possible CBIRs. Depending whether the CBIRs implements a nearest neighbor query or a range query, a varying number n of images are sent back to the client. If more than five images are returned, no matter which of these query types is used, the n images assigned with the smallest score values are chosen as the most similar ones to the query image. It is assumed that the scoring is normalized. This normalized scoring is subsequently published by the CBIRs with the results.

4.3.3. Automatic Configuration Analysis

In the following subsection the two major algorithms, which the automatic configuration analysis of the ***GeCCo*** approach is based on, will be discussed in detail. Each algorithm's paragraph will be followed by a paragraph that discusses a simplified example of the algorithm. It is simplified exclusively in the aspect of the number of involved low level features.

The, on the next page, following algorithm 2 outlines the cornerstones of the described approach. The algorithm's goal is to find optimal weight values for every feature, in case every feature is known in the representation. Conversely, it is desirable that the algorithm reports if

the analyzed CBIR system uses unknown features. The algorithm is initialized with all features that were implemented by the previously mentioned mutable CBIRs (line 2). The *while*-loop in line 3 is repeated until no more features can be discarded due to a low rating. The first *for* loop optimizes an objective function obj (line 7) for every image I_{k} in the image test set (*ITS*). The objective function is used to evaluate the fitness of possible feature weight vectors. It is listed in formula 4.5, with the feature weight vector of ω being tested, n being the number of result images returned by the CBIRs, and $score_\alpha(I_{Input}, I)$ being the returned score value for result image I respective to input image I_{Input} (see formula 2.8).

$$obj(I, \alpha, \omega) = \frac{\sqrt{\sum_{k=1}^{n} (score_\alpha(I, I_k) - score_\omega(I, I_k))^2}}{n} \tag{4.5}$$

As mentioned above, the objective function obj calculates a distance value for the scoring obtained by using a feature weight vector of ω and the actual scoring of the CBIRs. Larger distance values mean that a tested vector results in self-computed scores that are less similar to those returned by the CBIRs. This might mean there are larger differences between $score_\alpha$ and $score_\omega$ for the different result images. Furthermore, optimization algorithms - namely Particle Swarm Optimization [32], Cuckoo Search [157], Multi Directional Search [95], and Nelder-Mead Method [95] - were applied to find a feature weight vector which most closely resembled the CBIRs' configuration.

Particle Swarm Optimization hereby operates in the following way: Candidate solutions (called particles) of a population (called a swarm) are moved around in the search-space according to a few simple formulae. These particles' movements are driven by their individual best known position in the search-space as well as the whole swarm's best known position. This method discovered improved positions that come to guide the movements of the swarm. By repeating this process it is hoped, but not guaranteed, that an optimal method is discovered.

Cuckoo Search in turn imitates the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds. In the beginning a number of candidate solutions in the search space (eggs) are defined (are each put in a nest). In each iteration, a predefined number of cuckoos lay one egg at a time, and dump their eggs in a randomly chosen nest (replaces a worse solution with a better one). Hereby, the egg laid by a cuckoo is discovered by the host bird with a probability of $p \in (0, 1)$. These nests are abandoned and new ones are built. The best nests with the highest quality of eggs will carry over to the next generation. When the maximum number of iterations has passed, or another stop criterion is reached, the algorithm stops.

The Multidirectional Directional Search method is an optimization algorithm that does not depend on derivatives of the objective function. At any iteration $k \geq 0$, the Multidirectional Directional Search requires $n + 1$ points which define a non-degenerate simplex in $\mathbb{R}^n$. Non-degenerate denotes that the set of n edges adjacent to any given vertex in the simplex spans $\mathbb{R}^n$. The edges of the simplex are used to define the search directions, the orientations of the search directions, and the step size in each direction. Computing the function values at all $n+1$ vertices in the original simplex is the first step. Using this function information, the algorithm distinguishes the best vertex in the simplex, where the best vertex is defined to be the vertex having the smallest function value. The n edges connecting the best vertex to the remaining n vertices determine a set of linearly independent search directions.

The Nelder-Mead Method constructs an initial working simplex S to begin with. It repeats the following steps until the termination test is satisfied. It then calculates the termination test

information. If the termination test is not satisfied then transformation of the working simplex is executed. The best vertex of the current simplex S and the associated function value are returned.

Each of these optimization algorithms has been evaluated and the results are presented in section 4.4. This optimization is performed in line 7 of algorithm 1 for every image in *ITS*, so *FeatureWeights* consists of vectors with each vector containing the optimal weight values for one image in *ITS*. The reason for calculating an optimal feature weight vector for multiple images is to be able to conduct a statistical analysis of these values afterwards, which is done in the second for loop starting in line 9.

Algorithm 2 analyze(ITS, F_ω, CBIRsResults[])

```
1:  doContinue ← true
2:  PossibleFeatures ← F_ω          // initialization of Possiblefeatures with all features of GeCCo's mutable CBIRs
3:  while doContinue do
4:     doContinue ← false           // set continue to false by default
5:     for i = 1 to i = |ITS| do
6:        I ← ITS[i]                // allocation of the i^th image of the ITS
7:        FeatureWeights[i] ← optimize(obj(I, α, ω))   // optimization of the feature weighting of ω regarding input image I
8:     end for
9:     for i = 1 to i = |PossibleFeatures| do
10:       Ratings[i] ← rateFeature(i, FeatureWeights)   // rating of every feature in PossibleFeatures
11:       if (Ratings[i] < minRating) then
12:          PossibleFeatures.remove(i)                 // removing of i^th feature
13:          doContinue ← true                          // in this case set continue to true
14:       end if
15:    end for
16: end while
17: if PossibleFeatures.isEmpty() then
18:    return null                  // return that a calculation of the feature weighting was not possible
19: else
20:    return average(FeatureWeights)   // return the calculated feature weightings
21: end if
```

FeatureWeights contain multiple weight values for a single feature, one for every image in *ITS* (for example see figure 4.1 where for f_A three different score settings have been detected). As discussed in the following section, *rateFeature* assigns a rating value between 0 and 1 to every feature, depending on different configurable criteria (line 10). Currently, a higher standard deviation of the weight values of one feature for different images results in a reduction in its rating, as does too small an average weight.

An average weight of 5 % and higher is rated with 1. An average weight of less than 5 % is rated 0.2 less for every 0.1 % it is lower than 5 %. In general, a very low average weight can occur if the features that are required to approximate a CBIRs result set are not contained in F_α. Moreover, the optimized function tries to compensate for that fact by applying other features for every image of the ITS. The ratings of all features are also lowered if the distance values returned by the objective function are higher, which signifies that a weight vector cannot reproduce the CBIR system's behavior well enough.

If any feature has a rating smaller than *minRating* (line 11) it is not used again for future

executions of the outer *while* loop. This is because a smaller rating suggests a lower probability of a feature being used by the analyzed CBIRs. The *minRating* was defined to be 0.1 in respect to the significance of the previously discussed Feature ratings. The outer while of the algorithm is only repeated if at least one feature has been discarded in the current run (line 13). When the *while* loop finishes, depending on whether there are still remaining features in *PossibleFeatures*, an average of the previously computed optimal feature weights for the remaining features is returned. Otherwise it is reported that the analyzed CBIRs uses unknown features (lines 17-20).

The following paragraph illustrates the preceding algorithm by one of the first tests that were run in order to verify the functionality of its implementation. For that test run, ***GeCCo***'s internal mutable CBIRs used the following low level features: Edgehistogram, Colorhistogram, Tamura, and Gabor. The same features were used by the CBIRs that was analyzed and the applied weighting for these features was: 25 %, 25 %, 25 % and 25 %. Algorithm 2 applied the following method. The algorithm initialized with the features Edgehistogram, Colorhistogram, Tamura, and Gabor (line 2).

Subsequently the algorithm entered the *while*-loop (line 3). The first *for* loop (line 5 to 8) optimized the feature ratings using the objective function obj (formula 4.5) for every image in the image test set (*ITS*). This instance will be closely discussed in subsequent sections. In the present case this caused the algorithm to produce as many feature weightings each containing approximately 25 % as there are images in the ITS. This was possible as this was a really easy feature weighting to approximate.

A statistical analysis of these values was done in the second *for*-loop starting in line 9. *RateFeature* assigned ratings approximate to 1 to every feature (line 10) as no recognizable deviations between the feature weights of the different feature weighting vectors were present, and no feature was under the threshold of 1 percent. As no feature had a rating smaller than *minRating* (line 11) no feature was discarded. Therefore the outer *while*-loop of the algorithm ended. As the while loop finished and there were still remaining features in *PossibleFeatures*, an average of the previously computed optimal feature weights (approximatley 25 % for each) for the features of ***GeCCo*** was returned.

As mentioned in the beginning of the precedent paragraph, the function *objectiveFunction* is listed separately in algorithm 3. In order to provide a function taking a weighting vector and returning a distance value to an optimization algorithm, it first has to be initialized with the query image sent to the CBIRs, the CBIR system's results for that image, and the features currently used by the middleware. In the first example these were Edgehistogram, Colorhistogram, Tamura and Gabor. In the second example it was Edgehistogram, Colorhistogram, Tamura, and Gabor in the first repetition of the *while* loop and only Edgehistogram and Colorhistogram the second and third repetitions of the while loop. The objective function can,

Algorithm 3 objectiveFunction(weightVector)

1: // Note: TestImage, CBIRsResult and PossibleFeatures have already been initialized in Algorithm 1
2: **for** $i = 1$ to $|CBIRsResults|$ **do**
3: $ResultImage \leftarrow getImage(i, CBIRsResult)$ // choose the i^{th} image of the $CBIRsResult$
4: $SelfComputedScores[i] \leftarrow getDistance($
$TestImage, ResultImage, weightVector, PossibleFeatures)$ // calculation of the score for the i^{th} image
5: **end for**
6: $SelfComputedScores \leftarrow normalize(SelfComputedScores)$ // normalization of the computed scores
7: **return** $scoreDistance(getScores(CBIRsResult), SelfComputedScores)$ // return of the score distances

using this input, be called by the optimization algorithm in this initialized state to evaluate different feature weights. Each time it is called with a weight vector, an individual ranking of all images returned by the CBIRs is calculated in the *for* loop (line 3 and 4). To assign a self-computed score value to every result image, *getDistance* is called to calculate a distance value between the test image and the result image using formula 2.8. This distance value is calculated with respect to the currently used features and their weight values, with the test vector representing the weight values. The score values then have to be normalized such that they range from 1 to 0 for the most similar and most dissimilar result image, respectively.

This normalization has, at this point, already been applied to the CBIRs result's scores. Therefore, if the feature weights represented by the test vector are very similar to the feature weights used by the CBIRs, the computed scores of the ***GeCCo*** internal CBIRs should also be very similar to the CBIR system's result scores. The function *scoreDistance* is used here to calculate the distance between the two score-lists, returning a small value if the self-computed scores are similar to the result scores and a larger value otherwise. This distance value is then returned to the optimization algorithm.

4.4. Evaluation

The evaluation section is divided into three parts. The first part addresses the runtime of the implementation of this chapter's approach. The second part evaluates the performance of the presented approach respective to the detection of the features that were relevant for the analyzed CBIRs. Finally, the last part focuses on the accuracy of the weight detection for the detected features.

In order to prevent misunderstanding, it is stated that the ***GeCCo*** approach is addressing the problem of detection for configuration of a CBIRs. An improvement of the CBIRs search engine is not in the scope of ***GeCCo***'s approach. The subsequently used precision and recall measurements refer to the feature detection performance of the ***GeCCo*** approach and not to the retrieval performance of the participating CBIRs.

The tests used the publicly available image set of the ImageCLEF[2] benchmark [24]. The full set of 20,000 images was used. As a CBIR system, Lire [77] was used. LiRe[3] is a state of the art Java CBIR library. It extracts the image's features and stores them in a Lucene index. LIRe also provides an API for searching this index. No database is necessary, and only the integration of one Jar file is required in order to be able to apply the possibilities of a classical CBIR approach. Lire comes with a set of pre-implemented features. Namely those features are Colorhistogram, CEDD, FCTH, Fastcorellogram, Gabor, Tamura, and Edgehistogram. These features, including the definition of the term feature in general, are comprehensively discussed in section 2.4.3 and appendix A. Furthermore, Lire was chosen as it is an extensible library and could therefore be adapted with little effort.

4.4.1. Algorithm complexity

One possible way of approximating the configuration of a CBIRs is to try a number of feature weight vectors one by one, using brute force. In order to calculate the runtime of the brute force approach at hand, it can be broken down to the mathematical combination scenario referred to as k-combination with repetition. In this scenario, n is the number of elements that can be

[2] http://www.imageclef.org/2011
[3] http://www.semanticmetadata.net/lire/

chosen whereas k is number of times an element is chosen. Equation 4.6 visualizes the number of possibilities of the k-combination within the repetition scenario [15].

To interpret the formula correctly the variables have to be matched in the following way: k stands for the number of features (urns) and n stands for the granularity of weight values (balls). Seeing n from another view, it would mean that k^{-1} is the minimal percentile difference that can occur between two features. E.g. a granularity of 1 would enable the algorithm merely to identify whether a feature-metric combination was chosen with a weighting of 100% or 0%. By contrast, a granularity of 100 would make it possible to allocate individual weight values in the weight vector in steps of 1%.

$$t(BruteForce)) = \left(\!\!\binom{n}{k}\!\!\right) = \binom{n+k-1}{k}. \tag{4.6}$$

It visualizes the possible feature weighting combinations of the brute force approach of this chapter's approach, as previously discussed.

Taking into account these runtime deliberations using the brute force approach would lead to a prohibitively large runtime under realistic circumstances. The application of different optimization algorithms, which were comprehensively discussed in section 4.3.3, solved this runtime problem. An implementation of Cuckoo Search [157] as well as the implementation of Particle Swarm Optimization [32], the usage of Multi-Directional Search, and the Nelder-Mead Method from [95] all needed approximately 15 seconds for the analysis of one CBIRs. Depending on the test scenario and regarding the feature detection the optimization algorithms obtained an average accuracy of over 90 %. However, approximately two minutes of additional time was required for extracting the feature vectors from images and pre-calculating feature distance values between images. In order to make the optimization algorithms comparable to the brute force approach, the brute force approach was merely used with a granularity of 10 which resulted in the usage of 50.000 evaluations of the objective function and a similar runtime of about 15 seconds.

4.4.2. Feature Detection

At the time of the implementation of this chapters' approach, a limited number of 7 features - Colorhistogram, CEDD, FCTH, Fastcorellogram, Gabor, Tamura & Edgehistogram - were implemented by the applied CBIRs Lire. 500 different configurations - composed of weighted combinations of the previously enumerated 7 features - provided an adequate test environment in terms of quality of results and computation time. In the preparation of these evaluations tests with 10 to 5000 different configurations were conducted. After the evaluations of these tests, 500 emerged to be a sufficiently high number to obtain reliable stable results and still operate with a reasonable runtime of 2 hours. The evaluation of the feature detection performance of the ***GeCCo*** approach therefore involved the analysis of 500 different configurations of a Lire-CBIRs.

The 500 total configurations consisted of a set of 100 different configurations for each of the 5 classification cases visualized in figure 4.2. These configurations constitute the basic truth of the following evaluations. Subsequently a CBIRs was instantiated for every configuration and the ***GeCCo*** approach was applied to detect the features. For example in a test case a CBIRs was configured to apply a color histogram feature and an edge histogram feature.

As it is, the ***GeCCo*** approach aims to detect the features that are applied by the CBIRs and their weightings, so ideally ***GeCCo*** should detect that the CBIRs applies a color histogram

feature and an edge histogram feature and can determine its features. In another test case a CBIRs was configured to apply a color histogram feature and a new unknown feature. For this test case ideally ***GeCCo*** should detect that it is not able to determine the configuration of the CBIRs.

The image test set in question was thereby provided by a selection of 5 arbitrary images from the ImageCLEF test set. In the preparation of these evaluations tests, different evaluations using 1 to 100 arbitrary images from the ImageCLEF test set were conducted. After an evaluation of these pretests it was surprisingly discovered that already 5 images returned a very good result. Those 5 unequal arbitrary images of the ImageCLEF were delivering very good performance (see figure 4.1, 4.4 and 4.3) in a reasonable timeframe of a few seconds. By augmenting the number of images, no improvement of the performance was achieved.

Furthermore, for every optimization algorithm 50.000 evaluations of the objective function were performed. In the preparation of these evaluations tests, 1000 to 500.000 different evaluations were conducted. After an evaluation of these pretests, 50.000 evaluations produced a good performance in a reasonable timeframe. By augmenting the number of evaluations no noticeable augmentation of the performance was achieved. This section discusses the feature detection performance of the ***GeCCo*** approach. It is not until section 4.4.3 that the weighting detection performance of the ***GeCCo*** approach will be discussed.

Figure 4.3 shows the average precision and recall percentages regarding the correct identification of features for 100 different configurations of Lire. Each of these configurations is defined according to the definition of **case 1** in this chapter. Hereby the recall is defined as the percentage of the CBIRs' features that are included in the set of features that the ***GeCCo*** approach has identified. Whereas the precision is defined as the percentage of the features that the ***GeCCo*** approach has identified that are applied in the configuration of the CBIRs. False negatives are defined as features that were erroneously not identified as features used by the CBIRs. These false negatives are visualized in the graph as the percentage that the recall bar is lacking to 100 %.

To ensure equal opportunities in the evaluation, every optimization algorithm was limited to perform up to a maximum of 50,000 iterations of the objective function. In order to satisfy this criterion, the brute force approach had to be limited to a granularity of 10 %. Even though the weighting detection performance of the ***GeCCo*** approach will not be discussed until Section 4.4.3, the consequence of this limitation can be anticipated now. This limitation means that a feature weighting of 57 % could be approximated by the brute force approach to exactly as 60 % or 50 %. As a consequence, weightings that are smaller than 5 % are nearly impossible for the brute force approach to detect. This limitation was found for both the feature detection and the weighting detection functions (See section 4.4.3).

In the current case, **case 1**, all of the optimization algorithms returned mostly the same features as the selected features in the CBIRs configuration. This means that for nearly every possible configuration the optimal feature weighting calculated by the algorithms used all the features that were implemented by ***GeCCo***'s internal mutable CBIRs. This is the desired behavior, as ***GeCCo***'s internal representation implemented exactly the same features as the Lire-configurations in this case. False negative features, which are not marked as detected but are in fact used by the CBIRs, did not occur often. These few false negatives, reflected in the marginal deviation from 100 %, were mostly caused by configurations where one of the features used a weight percentage of 1 % or less. The precision of the brute force algorithm therefore stands as an exception. As explained in the precedent paragraph the brute force approach is limited to a granularity of 10 % in order to provide the same runtime to every algorithm. Due to this

limitation, a deviation of 9 % occurs for the precision bar of the brute force approach in figure 4.3.

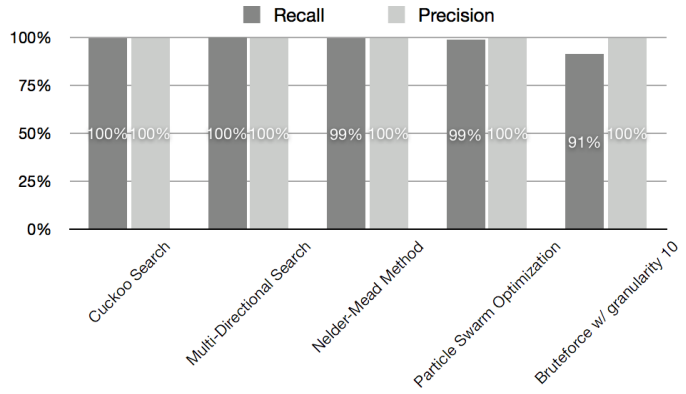


Fig. 4.3.: Precision and Recall: Case 1

Figure 4.4 shows the average precision and recall percentages over the 100 different configurations of case **case 2**. In this case the internal representation implements more features than the CBIRs provides. Here, false positives - features that are marked as detected but are not in fact used by the CBIRs - can occur in addition to the previously described false negatives. Cuckoo Search as well as Multi-Directional Search exhibited very good detection performance. For all the tests, every feature used was detected and false positives occurred in only a few cases.

The performance of the Nelder-Mead Method and Particle Swarm Optimization were within the testing accuracy of this framework nearly identical. They both had a very small percentage of false negatives (see recall) and a small percentage of false positives (see precision).

The false negatives and positives belonged to tests in which only a very small weighting value was assigned to a feature. This can be conceptually reproduced by the following reflection: What happened if the brute force approach was confronted with a weighting of 3 %. The brute force approach would have to determine whether the weighting of the feature is 0 % or 10 %. If the algorithm calculated the weighting to be 0 % it is shown in the graph as a false negative. In a real world scenario, though, it would not be very detrimental to the performance of the ***GeCCo*** approach to be not able to correctly identify features with very small weights or to incorrectly assign very small weights to irrelevant features. So these small deviations do not cause a problem.

The focus of the analysis of cases **cases 3 to 5** is not to understand which features were used but rather to discover that the configuration cannot be detected. This is the case du to the reason that in cases **cases 3 to 5** ***GeCCo***'s internal mutable CBIRs is merely aware of a fraction, or even none of the features of the CBIRs that is to be analyzed (see figure 4.2). Attempting to identify features that are not implemented by the ***GeCCo*** internal mutable CBIRs approach would inevitably fail. Therefore, it would not have been useful to calculate recall and precision values for these cases. For this reason, in table 4.1 only the success rates of the different optimization algorithms for the remaining cases are illustrated. A test case was counted as successful if our implementation returned that it was not able to determine the CBIR system's configuration.

Line 2 of table 4.1 shows the average success rate over the test runs for every algorithm of case **case 3**. Here, each of the algorithms, in about 90 % of the test cases, detected that the internally implemented features were not a superset of or equal to the CBIR system's features. **case 4** is very similar to case 3 in most tests cases. As the mutable CBIRs had a larger amount of features

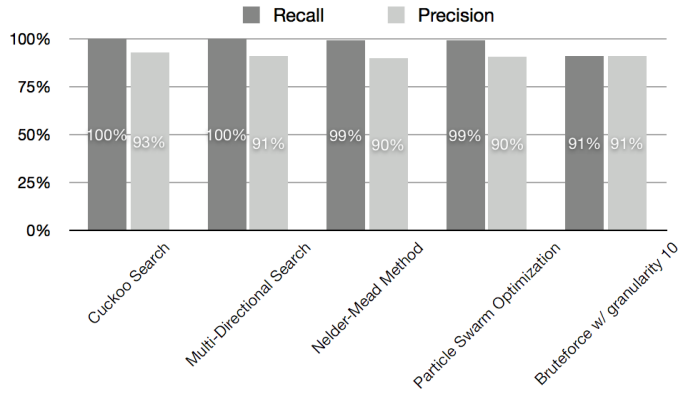


Fig. 4.4.: Precision and Recall: Case 2

	Cuckoo Search	Multi-Directional Search	Nelder-Mead Method	Particle Swarm Optimization	Bruteforce (Granularity 10)
Case 3	*91.00%*	*91.00%*	*91.00%*	*91.00%*	*89.00%*
Case 4	88.00%	90.00%	87.00%	87.00%	90.00%
Case 5	100.00%	100.00%	100.00%	100.00%	100.00%

Table 4.1.: Sucess rates of Cases 3, 4 and 5

available to approximate the CBIR system's behavior, in marginal cases a CBIRs's configuration was sometimes incorrectly considered to be detected. In **case 5** all the applied optimization algorithms, as well as the brute force approach, were able to detect this classification class.

4.4.3. Weighting Detection

In this subsection the weighting detection performance of the approach is evaluated. This is done by using the test configuration and the data of the previously discussed feature detection evaluation. Thereby the deviation of the detected feature weighting of the CBIRs, and the feature weighting that the CBIRs was instantiated with, was calculated using the Euclidian distance. Consider the example that two features, referenced as f_1, f_2, had a weighting of 0.4 and 0.6 and a detected feature weighting of 0.5 and 0.5. In this case the weighting deviation is calculated as the euclidian distance between the vectors (0.4, 0.6) and (0.5, 0.5). Consider the example that three features, referenced as f_1, f_2, f_3, had a weighting of 0.1, 0.2 and 0.7 and a detected feature weighting of 0.3, 0.0 and 0.2. In this case the weighting deviation is calculated as the euclidian distance between the vectors (0.1,0.2, 0.7) and (0.3, 0.0, 0.5).

Figure 4.5 shows a visualization of the deviation for every implemented algorithm of test case 1. All of the algorithms used had a small weighting deviation. Cuckoo Search had a deviation of 0.0175, whereas the Multi-Directional Search and the Nelder-Mead Method had a deviation of 0.0217 and 0.0271, respectively. PSO and the brute force approach had a deviation of 0.0322 and 0.0637, respectively. The bigger deviation of the brute force approach is explained by its' limitation to the 10 % step for the reason of the time limitation. This was put in place to guarantee equal opportunities for each algorithm.

In Figure 4.6 the average deviation for test case 2 is visualized. Again, all of the algorithms have a small weighting deviation, though most are slightly larger than in case 1. All of these deviation values are relatively small, meaning that all of the algorithms are able to approximate

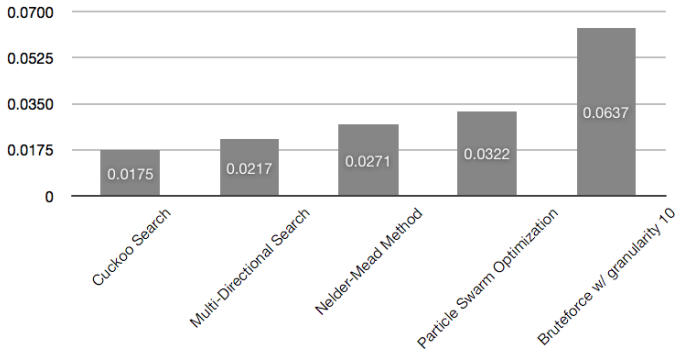


Fig. 4.5.: Averaged weighting deviation in Case 1

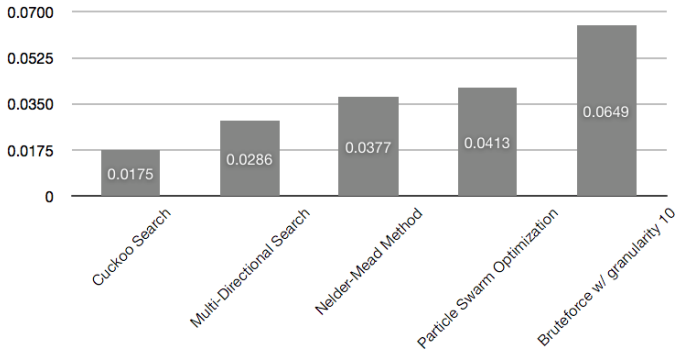


Fig. 4.6.: Averaged weighting deviation in Case 2

CBIRs configurations well if all used features are known. The excellent performance of the algorithms has to be attributed, on the one hand to the quality of the approach, on the other hand to the small number of features that were implemented by the Lire library in the preliminary implementation of this chapter's approach.

Under these circumstances, the calculation was easier and the results obtained were more accurate than when a bigger number of features is applied. It is to expect that the accuracy of the algorithms decreases when the number of features augments and therefore the number of evaluations or the number of images in the ITS would have to increase. Considering the results presented and discussed in figures 4.3, 4.4, 4.1, 4.5 and 4.6, and the fact that Cuckoo Search performed best in every single test case, except for test case 4. Cuckoo Search can overall be considered as the best algorithm in this evaluation.

4.5. Conclusion

This chapter presented a novel approach for the detection of CBIRs' configuration. The focus of this chapter's approach was on the correct identification of feature settings and their assigned weights. Therefore in section 4.3.3 an algorithm was defined that reduces a set of potential features and computes the feature weightings of the features of the CBIRs that is to analyze. This is done by applying a previously defined objective function, an optimization algorithm and a predefined rating strategy. Five different classes of cases have been highlighted. The proposed approach is able to reduce the number of features and calculate their weights in two

of the classified cases. In the three remaining classes of cases it can merely detect and inform that it is not able to determine the CBIRs configuration. Moreover, it was demonstrated in this chapter that the problem can be traced back to an optimization problem. The evaluation showed a high accuracy of the feature-weight detection and the capability of the system to identify cases that cannot be dealt with a precision. It demonstrated good performance by the use of all the four tested different optimization algorithms. Hereby it was shown that Cuckoo Search can overall be considered as the best algorithm in this chapter's evaluational set up.

5. QueDi: Query Distribution

Due to the immense growth in sales of smartphones and their mobile applications in recent years[1] , images are distributed worldwide in various kinds of repositories and search of these leads to a distributed search scenario. Therefore, this thesis specifies an approach in the following chapter, called ***QueDi***, which is improving the effectiveness of query forwarding in CBIRs as well as augmenting the number of relevant results. This is achieved by specifying an architecture for - as well as a method to query - a semantic CBIRn. The feasibility of the approach is demonstrated by the implementation of a prototype and its evaluation against a self defined non-semantic CBIRn using 87,500 classified images of *ImageNet*. Hereby, an augmentation of the number of returned images that are of the same semantic concept as the input images is achieved by a factor of 4.0. Although the approach in this chapter will eventually be combined with the approaches of chapters 3 and 4 in section 5.5, this chapter's approach is until then to be seen as context independent.

5.1. Introduction

It was already mentioned in section 2.3 that the majority of images available in the Internet are predominantly searchable in a text-based way. To make these images searchable via text, now and in the concept's early days, a method had to be applied to discover, extract, and identify text that described these images. As solely automated methods could not deal with this amount of images, and due to the fact that web 2.0 images are frequently not necessarily described but merely commented on, CBIR has caught a lot of scientific attention in recent years.

Section 2.4 explained how CBIR enables users to search for images using example images instead of using textual descriptions. It is extensively explained that the LLFs of stored images are extracted and matched with the extracted LLFs of query images. By that means a scoring and a ranking of the images is calculated and subsequently transmitted to the clients. CBIR is a research field that has been intensely explored in the last decade. Research has found a huge amount of effective and efficient automated LLF extraction processes. The theoretical background of CBIR and systems implementing CBIR, so called CBIRs, is intensely explored in [22, 127] and various systems implementing CBIRs were surveyed in [139].

However, although the CBIRs strategy seems very promising, the total of all images available in the Internet could neither be managed by one database nor by one LLF. That is because, on the one hand the number and the size of all the images could not be managed by one CBIRs. On the other hand different LLF are good for describing different semantic concepts (see chapter 2.4). For the preceding reason, it would be ineffective to exclusively use instances of one CBIRs, or one LLF, in a possible Content Based Image Retrieval network (CBIRn). Therefore the construction of a CBIRn that combines a variety of image retrieval systems appears to be necessary.

As discussed in chapter 2.5 the general manageability of CBIRn has been proved by now (i.e. in [6, 103]). In chapter 2.5, implementations of CBIRn, distinct CBIRn architectures, and

[1] `http://www.statista.com/statistics/74592/`

query distribution protocols, were discussed. Unfortunately these networks were a mere connection of similar CBIRs entities and therefore the implementations ignored the complexity and chances that emerge in the integration of a variety of CBIRs. To render the integration of distinct CBIRs possible, chapter 2.5 discussed protocols that allow the integration of distinct CBIRs as well as protocols that render it possible to query CBIRs in heterogeneous CBIRn.

Consequently this chapter proposes a query distribution approach in an original semantic Content Based Image Retrieval network (semantic CBIRn) that will be introduced in this chapter. Logically this is achieved by proposing a design of a semantic CBIRn that consists of a central unit that unites CBIR subnets, where each take care of distinctive semantic concepts (e.g. cars, elephants, trees). Each of these logical CBIR subnets use CBIRs that are individually preconfigured for one specific semantic concept. These preconfigured CBIRs in turn hold extracted low level feature (LLF) vectors of - and references to - images that correspond to their semantic concept.

The approach developed in this chapter makes a first step towards the automatic distribution of CBIR queries in such a semantic CBIR network. It does so, inter alia, by analyzing the input image and detecting its semantic concept. Subsequently this detected semantic concept is matched to the configurations of the CBIRs for the CBIRn. Eventually, the query image is forwarded to those CBIRs whose configurations are most suitable. The key advantage of this approach is the exclusive usage of CBIRs, whose configurations favor images of the query image's semantic concept. This way CBIRs that would not add any benefit to the query response can be omitted beforehand. Furthermore the knowledge about the configurations of selected CBIRs could be used to improve the result aggregation process of the retrieved result sets.

The remainder of this chapter is organized as follows: Section 5.2 and 5.3 outlines the assumptions contained and explains the developed approach. Section 5.4 defines the setting of the evaluation and analyzes the results. Following section 5.5 will eventually combine ***QueDi*** with ***MeRRSe*** and ***GeCCo*** in an evaluation and discuss the results. Eventually in Section 5.6 this chapter is concluded and possible future adaptions of the approach are discussed.

5.2. Methodology

An overview of the proposed approach is given in figure 5.1. The figure's components are explained one by one in the course of this section. The next paragraph of this section thus starts with a sketch of the query flow. This paragraph is followed by a paragraph explaining the idea of the semantic CBIRn. In particular its subdivision into semantic concepts and the specialized CBIRs are illustrated. The following paragraphs elaborates the idea of the central unit that unites several crucial services for the semantic CBIRn. In these paragraphs the stored information about, and the usage of, the ideal CBIRs configurations are drafted. In addition, the configurations of the CBIRs involved and the concept detection approach used to analyze the input image are outlined.

QueDi's search process, that takes advantage of all the previously mentioned components, is visualized in figure 5.1. In this process the query image is chosen by a user or a user instance

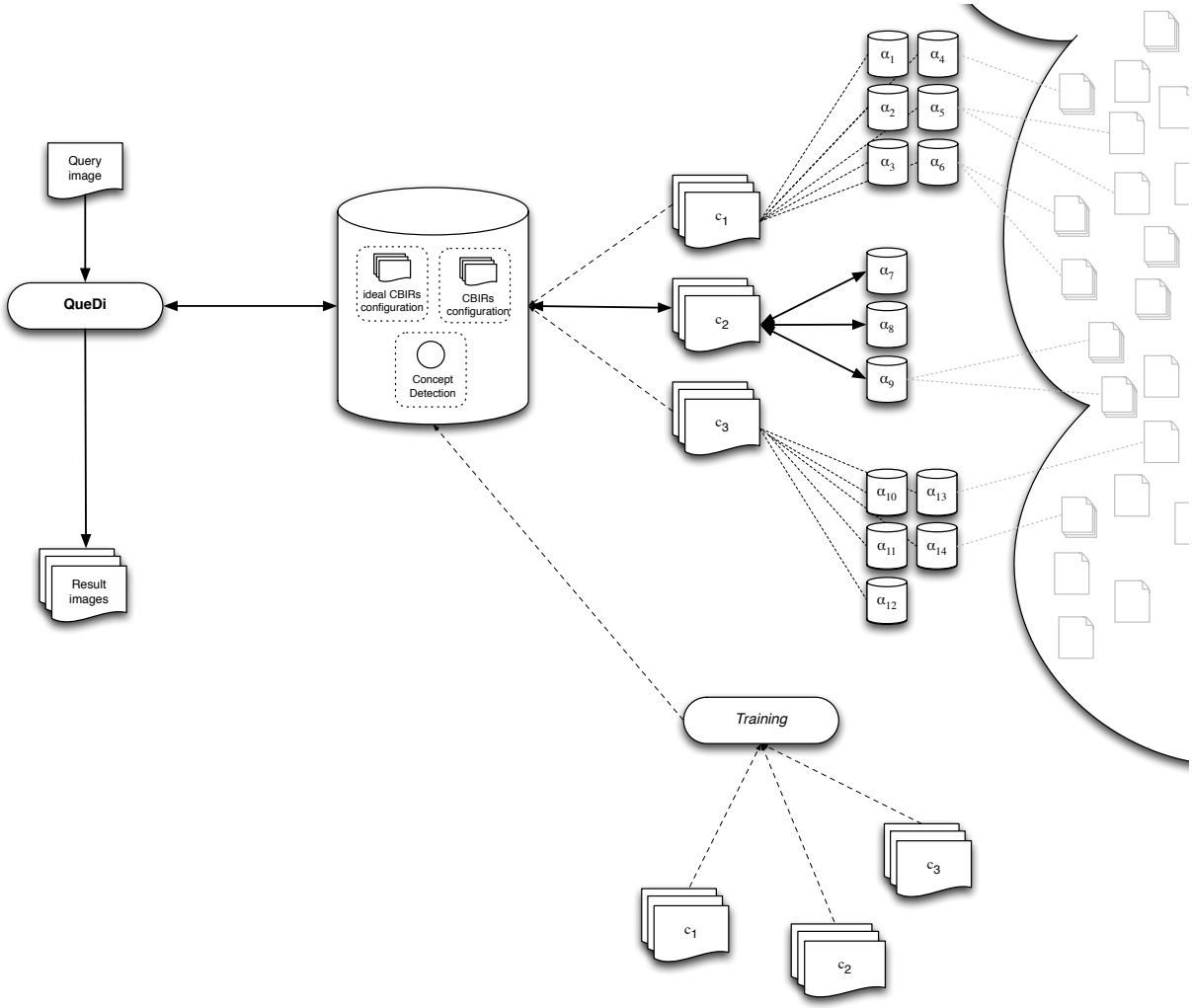


Fig. 5.1.: The Quedi Approach

of a system beforehand. At the beginning of the search process this query image is sent to the query distribution unit. By taking advantage of a concept detection algorithm the semantic concept of the query image is identified. Subsequently the registered ideal CBIRs configuration for this semantic concept is identified. An ideal CBIRs configuration for this semantic concept is hereby considered as follows: It gives images of the specific semantic concept that it is defined for a significantly better rating when it is queried for an image of this semantic concept. By verifying this ideal configuration against the configurations of the semantic CBIRn's CBIRs, ***QueDi*** distributes the query image only to CBIRs whose configuration match the input images' semantic concept while taking into consideration every CBIRs of the semantic CBIRn. Subsequently the returned images are in the currently examined ***QueDi*** approach merged - using round robin - into a single comprehensive result list. This comprehensive list is finally sent back to the user or the user instance of a system. The key components of this approach, the semantic CBIRn and the central unit, are visualized in figure 5.2.

Figure 5.2 shows a simplified version of the semantic CBIRn in terms of number of semantic concepts and CBIRs. A user would experience it like a standard CBIRs. A predefined graphical user interface (as per annex B) would be deployed, thus the complexity of the underlying structure is not visible. However as illustrated righthand of the central unit, the semantic CBIRn is subdivided into semantic concepts - in figure 5.1 and the following figures marked with a $c_{i\in\mathbb{N}}$. These semantic concepts in turn are represented by numerous CBIRs each - in figure 5.1 and the following figures marked with an $\alpha_{j\in\mathbb{N}}$.

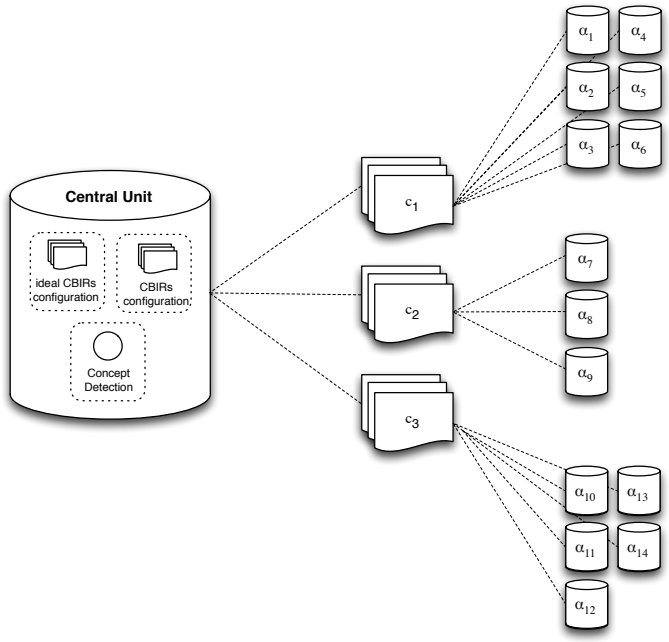


Fig. 5.2.: The central unit and the semantic CBIRn

The central unit that is visualized on the left side of figure 5.2 basically combines three functionalities. It continuously gathers information about the configurations of every CBIRs that is integrated in the semantic CBIRn on the one hand. It stores the ideal CBIRs configuration depending on the semantic concept employed on the other hand. Additionally it guards the information that is necessary for the concept detector.

Firstly, the part of the central unit which is occupied with the CBIRs' configurations is discussed. The information about the configuration of the CBIRs that were registered in the semantic CBIRn is continuously kept up to date. This is due to the reason that whenever a semantic concept is added to, or removed from, the semantic CBIRn all of the ideal CBIRs configurations have to be recalculated as the optimal configurations relate to all the used semantic concepts. Additionally it has to be taken into consideration that the ideal configurations are not exclusively used to identify CBRs that match best the semantic concept of a given query image. Besides, when the CBIRn is extended in terms of number of images or semantic concepts,

specific additional CBIRs have to be instantiated or added to the CBIRn in order to cope with the quantity of the images and the semantic concepts. These configurations are used to configure additional CBIRs, which are then configured with the ideal configuration of the semantic concept they are attached to. Therefore their configurations eventually have to be updated whenever a semantic concept is added or removed from the semantic CBIRn.

Secondly, the part of the central unit which is occupied with the concept detector is discussed. Even the information for the concept detector has to be recomputed for each semantic concept that is added to, or removed from, the semantic CBIRn. Weka [104] was used as concept detector in this thesis. Weka[1] is an open source software developed by the University of Waikato. It provides various machine learning and data mining algorithms. Because of the available API, methods can be easily integrated in existing Java projects [49].

Thirdly, the part of the central unit which is occupied with the ideal CBIRs configuration regarding a semantic concept is discussed. The ideal configuration for a semantic concept is considered to give images of the specific semantic concept it is defined for a significantly better rating when it is queried for an image of this semantic concept. By using this configuration, it becomes less likely for images to be retrieved that are not members of the desired semantic concept. In order to calculate these ideal configurations, the following process takes place: Every image of every semantic concept of the training set is allocated to a CBIRs whose configuration is mutable. Subsequently this CBIRs is confronted with queries for each individual image. This procedure is repeated for all possible configurations of the CBIRs. The configuration which returns the largest number of images of the same semantic concept as the input images' is chosen to be the ideal configuration. Due to the fact that the images contained within these semantic concepts were manually selected, and therefore of good quality, the value of the returned images could hereby be left out of consideration. As this thesis defines a novel approach towards the identification of the ideal configuration of a CBIRs, concerning a semantic concept, a valuation of the ranking positions of the returned result was for now abstained from. Methodologically speaking the process is implemented the following way: A process that calculates the ideal weight for each feature is used. The obtained result consists of a weighting vector. This vector is later, on the one hand, used for the configuration of a specialized CBIRs of the semantic CBIRn. On the other hand this weighting vector is used in order to rate the externally integrated CBIRs in the network. Since the similarity comparison of the CBIRs is based on a linear combination of the individual feature scores, a linear discriminant analysis was applied, which in this case was Fisher-LDA [43]. Fisher- LDA was chosen due to its good and fast performance, as well as appearing to be the most up-to-date tool for this purpose. In support of this claim, it is applied in several algorithms for Face recognition [70, 74, 73] and detection [65, 41], Handwritten digit recognition [121, 156], Palmprint recognition [149], and Seed classification [7].

[1] `http://www.cs.waikato.ac.nz/ml/weka/`

5.3. Approach Outline

The approach that was drafted in section 5.2 will be explained on a theoretical basis in the following section. The section starts with the design specification of the proposed semantic CBIRn, including definitions of the core ideas of this approach. This specification is followed by a comprehensive discussion of the approaches prerequisites. Eventually, the section is concluded by a detailed illustration of the query distribution process.

5.3.1. Network Design

The key component of this chapter's approach, as stated in the previous section, is the semantic CBIRn. This system is then internally virtual partitioned into semantic concepts. Semantic concepts in the context of this thesis are defined as follows:

Definition 1 (Semantic Concept) *A semantic concept is "[..] an idea or mental image which corresponds to some distinct entity or class of entities, or to its essential features, or determines the application of a term (especially a predicate), and thus plays a part in the use of reason or language.[..]". This definition seizes on the dictionary record of the word 'concept' in [52].*

Formulas 5.1, 5.2, and 5.3 define the previous definition formally. Let

$$A = \{\alpha_1, ..\alpha_n\} \tag{5.1}$$

be the set of CBIRs that exist in a CBIRn. Furthermore let

$$C = \{c_1, ..c_m\} \tag{5.2}$$

be the set of semantic concepts that exist in a CBIRn. Assuming that each of the available CBIRs $\alpha_j \in A$ is specified to perform optimally with the images of a certain concept $c_i \in C$, then the CBIRs' of semantic concept c_i are indicated as follows:

$$A_{c_i} = \{\alpha_j | \alpha_j \in A \wedge \alpha_j \; is \; optimized \; for \; c_i\} \tag{5.3}$$

'Sky', 'tree' and 'elephant' are examples of such semantic concepts. By applying clustering or learning algorithms on this trio of semantic concepts they could be identified in many cases by machines nearly as accurately as by human beings. More complex semantic concepts, such as 'abstract painting', 'bedspread' and 'sculpture' are, nowadays due to the diversity of the contents of their images, very hard, if not even impossible, to be identified by machines.

As humankind thinks and searches, in semantic concepts - the idea of employing semantic concepts in a distributed content-based image retrieval scenario has emerged. Due to the circumstance that not every semantic concept is identifiable by machines, the semantic concepts C are defined for the moment by humans. In this approach less complex semantic concepts such as 'sky', 'tree' and 'elephant' are employed instead of semantic concepts such as 'abstract painting', 'bedspread' and 'sculpture'. As stated earlier, these semantic concepts are used to classify the CBIRn. In a real world scenario, every CBIRs is enrolled in the central unit of the semantic CBIRn, meaning that the configuration of every CBIRs of the CBIRn is registered in the central unit of the CBIRn. The configuration of a CBIRs and the ideal configuration of a CBIRs concerning one single semantic concept are in turn used to derive the matching CBIRs for every incoming query image.

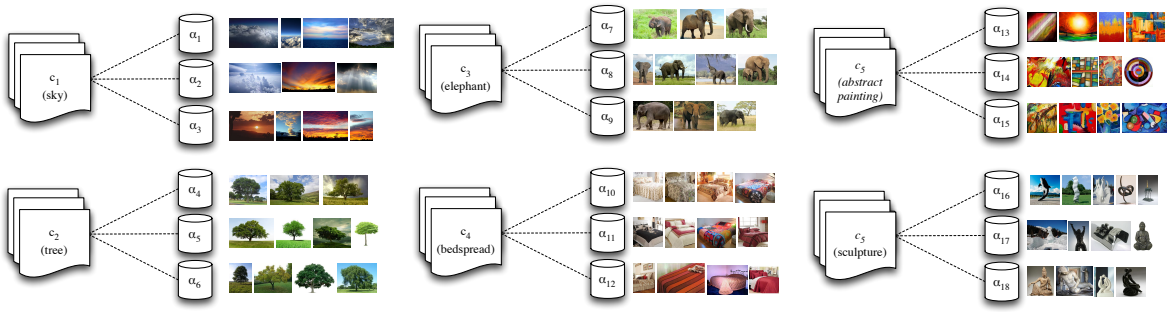


Fig. 5.3.: Example images of semantic concepts segmented to CBIRs

Each CBIRs α_j of the CBIRn, that is instantiated by the semantic CBIRn itself, is configured according to one of these ideal configurations. This way, each CBIRs is configured in a way that favors images of one specific semantic concept, reducing 'noise' images that are irrelevant. External CBIRs are most probably neither specialized on one semantic concept nor instantiated using one of those ideal configurations. But as this thesis focuses on specialized CBIRs that are e.g. applied in museums or crawl the Internet for cartoon images, the fact that a targeted CBIRs is specialized is never the less possible.

Definition 2 (Ideal Configuration of a CBIRs concerning a semantic concept) *The ideal configuration of a CBIRs concerning a semantic concept is the configuration (out of all possible configurations regarding the CBIRs' features) that maximizes the number of images of the semantic concept that the CBIRs returns when it is queried for an image of this semantic concept.*

The images contained by these semantic concepts were manually classified. Thus the classification regarding the semantic concept can be considered to be of good quality.

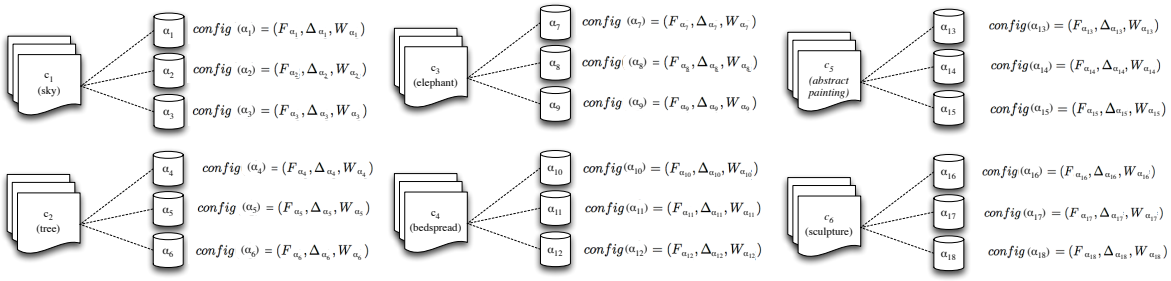


Fig. 5.4.: Distinct configurations of CBIRs for distinct CBIRs

Using this strategy, an input image can instantly be analyzed by the concept detector of the semantic CBIRn and can be categorized into one of the predefined semantic concepts. Currently, if an input image is received whose semantic concept is not taken into account by the semantic CBIRn, it is erroneously matched to a semantic concept and images of this erroneously matched semantic concept are returned. Therefore the development of a fallback strategy and the augmentation of the number of semantic concepts over time is equally important. The concept detection unit is trained for the first time when the semantic CBIRn is generated and is updated whenever a semantic concept is added or removed from the pool of concepts of the semantic CBIRn. This enables the concept detector to stay up to date and make optimal decisions for the later processing.

Similar to the concept detector, the ideal configurations for each semantic concept are calculated for the first time when the semantic CBIRn is created and is updated whenever a semantic concept is added to or removed from the pool of concepts of the semantic network. In order to facilitate the understanding of the calculation of the ideal configuration of a semantic CBIRs a more detailed insight will be given in the following sections.

The ideal configuration of a CBIRs regarding a semantic concept is calculated in the following way (see algorithm 4). Each image of every semantic concept of the semantic CBIRn is allocated to a mutable CBIRs (line 1). Mutable in this case means that the configuration of the CBIRs can be altered. Starting from this situation a naive approach would be the following: query the CBIRs, for an image of a specific semantic concept and count the images of the semantic concept that are contained in the returned images (line 7). This process would be repeated for every image of the semantic concept and every possible configuration (line 4 and line 6). The configuration that returns the highest number of suitable images back is consigned to be the ideal configuration (line 12). This approach would, however, result in an exorbitant runtime.

Algorithm 4 getIdealConfFor($imgSetSemanticConcept$, $imgSetAllSemanticConcepts$)

Require: $mutableCBIRs$

1: $allocateImagesToCBIRs(imgSetAllSemanticConcepts, mutableCBIRs)$ // allocation of images
2: $maxNumberOfReturnedImages = 0$
3: $idealConfigurationForSemanticConcept = null$
4: **for all** $configuration$ of $mutableCBIRs$ **do**
5: $numberOfReturnedImages = 0$
6: **for all** $image$ in $imgSetSemanticConcept$ **do**
7: $i \leftarrow getNoOfImagesOfSemanticconcept(image, mutableCBIRs)$ // number of correct images
8: $numberOfReturnedImages \leftarrow numberOfReturnedImages + i$
9: **end for**
10: **if** $maxNumberOfReturnedImages < numberOfReturnedImages$ **then**
11: $maxNumberOfReturnedImages \leftarrow numberOfReturnedImages$
12: $idealConfigurationForSemanticConcept = configuration$ // storing of current ideal configuration
13: **end if**
14: **end for**
15: **return** $idealConfigurationForSemanticConcept$ // return of the ideal configuration

In order to reduce the runtime that results from the number of queries, the original problem is transformed in algorithm 5 to a pairwise distance calculation between the images cluster centers of the semantic concepts. Using Fisher's LDA as the fitness function, the problem is solved using an evolutionary algorithm. Fisher's LDA measures hereby the quality of the separability of two classes in a feature space regarding a projection direction. In this case the projection direction is defined as a CBIRs weighting vector. A projection direction (weighting vector) would be desirable that minimizes the distribution of the two classes while at the same time it maximizes the mean distance between them. This idea is formalized mathematically by the Fisher's LDA and subsequently maximized using an evolutionary algorithm.

The clustering (alg. 5 line 1 and 2) is carried out by employing the LBG-Vectorisationquantizer [72] named after Linde, Buzo, and Gray. This process is a generalization of the Lloyd Algorithm and divides data points in the same way as k-means into a predefined number of k cluster points (10 in this approaches implementation). The LBG-Vercorisationsquantisizer is easy to implement and performed very well in preliminary tests. For these reasons, this algorithm was

chosen for the first implementation of this chapter's approach. As a result, every datapoint is assigned to the cluster to whose centroid its distance is the smallest. The problem that emerges hereby and has to be resolved, is how to maximize the mean distance between a semantic concept and the other semantic concepts of the semantic CBIRn, while minimizing the distance between clusters in the semantic concept itself.

Algorithm 5 newGetIdealConfFor($imgSetSemanticConcept$, $imgSetAllSemanticConcepts$)

Require: $mutableCBIRsystems$

1: $clusterCentersAllSemanticConcepts \leftarrow cluster(imgSetAllSemanticConcepts)$ // clustering 1
2: $clusterCentersImgSetSemanticConcept \leftarrow cluster(imgSetSemanticConcept)$ // clustering 2
3: $weigtingVectors \leftarrow initWeightingVectors()$ // initialization of the weighting vectors
4: $idealConfiguration \leftarrow vector[0]$ // initialization of the $idealConfiguration$
5: $iLDA = null$ // input variable for the Fisher's LDA
6: $iLDA.add(mutableCBIRsystems)$ // adding of reference to mutable CBIR system
7: $iLDA.add(clusterCentersImgSetSemanticConcept)$ // adding of cluster centers for the input semantic concept
8: $iLDA.add(clusterCentersAllSemanticConcepts)$ // adding of cluster centers for all remaining semantic concepts
9: **repeat**
10: $continue \leftarrow false$ // set $continue$ to indicate that $idealConfiguration$ has not changed in the loop
11: **for all** $vector$ in $weigtingVectors$ **do**
12: **if** $fLDA(iLDA, vector) > fLDA(iLDA, idealConfiguration)$ **then**
13: $idealConfiguration \leftarrow vector$ // set current vector as idealConfiguration
14: $continue \leftarrow true$ // indicate that $idealConfiguration$ has changed in the loop
15: **end if**
16: **end for**
17: **if** $continue = true$ **then**
18: $weigtingVectors \leftarrow recombinate(weigtingVectors)$ // recombination of weighting vectors
19: **end if**
20: **until** $continue = false$ // end if $idealConfiguration$ has not changed in the loop
21: **return** $idealConfiguration$ // return of the ideal configuration

This problem is resolved by applying an evolutionary algorithm and the Fisher's LDA as its fitness function (line 12). Therefore, in line 12, the function $fLDA()$ summarizes the pairwise Fisher's-LDA-calculated values. The values are calculated regarding the weighting vector that has to be verified (second input parameter of $fLDA()$), the cluster centers of the image set that an ideal configuration has to be identified for and the cluster centers of the remaining semantic concepts of the semantic CBIRn (included in the first input parameter of $fLDA()$). Preparations and initializations of the necessary variables take place in line 1 to 8. Hereby the main part of the necessary input variables for the Fisher's LDA are defined in line 5 to 8. In line 3 one hundred weighting vectors are initialized with random weights. The repeat loop (line 9 to 19) is carried out as long as the ideal configuration does not alter (line 13) any more. An alternation of the ideal configuration is indicated by setting the variable $continue$ to $true$ (line 14). An alternation of the ideal configuration is only achieved if for one of the weighting vectors (line 11) the Fisher's LDA is bigger than for the Fisher's LDA of the current ideal configuration. If in line 17 an alternation of the ideal configuration is indicated the evolutionary algorithm is called. In the implemented evolutionary algorithm, a weighting vector corresponds to an organism. A simple crossover, performed in line 18, is chosen for the recombination. Two organisms/ weighting vectors (A and B) become a new organism/ weighting vector through

the recombination of the first part of A and the second part of B, or vice-versa. The choice of the breakpoint happens randomly. In the selection, only those organisms/ weighting vectors A and B are selected that achieve the highest value regarding the fitness function. The remaining organisms/ weighting vectors of the population are replaced by descendants of A and B.

The concept detection, the information about the ideal configuration regarding a semantic concept, and the configuration of the CBIRs successfully empowers the semantic CBIRn to identify the CBIRs that are configured in a way that means they are anticipated to reply with the best results.

5.3.2. Prerequisites

The previously developed network design needs to follow certain prerequisites in order to perform in the desired way. These prerequisites encompass the predefinition of semantic concepts, knowledge about the configuration of the CBIRs, and that whcich are necessary for the ideal configuration of a CBIRs regarding a semantic concept. In the following subsection these prerequisites will be extensively discussed.

A further necessity of ***QueDi*** is a reliable concept detection strategy. As pointed out in the precedent section the concept detection strategy is trained in the beginning of the approach. Furthermore the concept detection strategy needs to be re-trained whenever a new semantic concept is added to the semantic CBIRn.

In order to be able to create a semantic network that is tolerant to erroneously assigned images of semantic concepts, the previously discussed ideal configuration is crucial. There are two major requirements that have to be considered in order to maintain the performance of the ideal configuration. On the one hand, as many features as possible have to be available for the pool of features that can be accessed to compute the ideal configuration of a CBIRs regarding a semantic concept. A bigger amplitude of this pool includes more distinctive features and therefore results in more possibilities of combinations and most certainly in better results for calculations of ideal configurations of a CBIRs regarding a semantic concept.

On the other hand a controlled set of metrics, p-norms (p=2, p=2.5 and p=3) and feature specific metrics, fitted to the pooled features is required. A set of metrics is necessary for the same reason as described for the multitude of features; to have more possibilities to combine and most therefore obtain better results for the calculation of the ideal configuration of a CBIRs regarding a semantic concept. The controlled set of metrics is necessary to keep the computational cost for the ideal configuration down. In the first implementation of this approach 7 features were used, namely Colorhistogram, CEDD, FCTH, Fastcorellogram, Gabor, Tamura & Edgehistogram. An explicit study that correlates the number of used features with the effectiveness of the realization of the ideal configurations for the semantic concepts is still pending.

The preliminary implementation of this approach has been orientated to search for one semantic concept per image.

Finally prerequisites for very important components of the semantic network, and for the employed CBIRs, are discussed. In order to be able to adapt to the ideal configurations these CBIRs have to be able to harness the pool of known features and metrics. Not only must it be possible to define their configuration in the construction phase of the semantic CBIRs, it must also be feasible to adapt the configuration of the CBIRs at runtime. The CBIRs is also required to have the capability to add and remove images that have to be registered or unregistered in its system during runtime. This last precondition becomes necessary whenever a semantic concept is added to the semantic CBIRn, as this might cause various rearrangements of the files

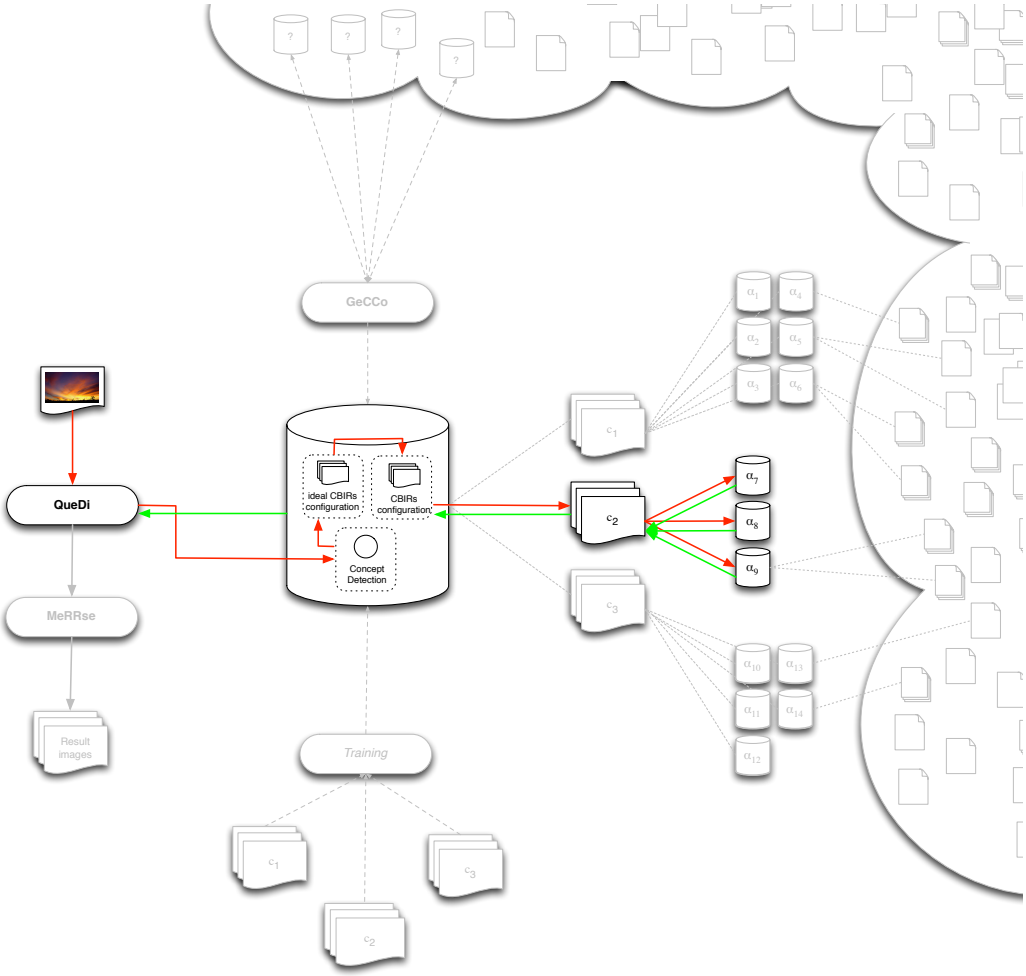


Fig. 5.5.: The path of a query in QueDi

assigned to the CBIRs.

5.3.3. Query Distribution

In the following subsection the query distribution are discussed in detail, following a short overview of the process. In the beginning of this chapter, in the description of figure 5.1, it was explained that images that the semantic CBIRn is queried for are initially analyzed by a concept detection strategy. In the same step of the approach, a mapping to a predefined semantic concept (c_1) takes place. In the subsequent step, the forwarding of the query image to the most promising member CBIR systems of the semantic CBIRn takes place. This mapping is internally conducted by calculating the euclidean distance between the weightings of the optimal configuration for the semantic concept, which has been experimentally evaluated beforehand (see 5.3.1), and the weightings of the enrolled CBIR's configurations. Eventually, the returned result sets are merged and sent back to the user.

At the beginning of the approach, the query distribution process, which is visualized in figure 5.5, receives an input image from the user. In figure 5.5 this action is indicated by the red

arrow that is pointing from the sunset image to the ***QueDi***-box. The adjoining red arrow, which points into the semantic CBIRn - towards the concept detection unit - represents the following handover of the input image to the semantic CBIRn and its immediate analysis by the concept detection unit. As visualized in figure 5.6, the concept detection unit determines a probability for every semantic concept whether the semantic concept occurs in the query. Naturally the concept detection unit can only discover those concepts it was initially trained for. In figure 5.6 the ideal CBIRs configuration is eventually chosen based on the class of probabilities that are returned by Weka. For the moment, the input image is assigned to the semantic concept that shows the highest probability.

In the next step, the ideal feature weightings for this semantic concept are therefore looked up in the central unit. By comparing these ideal feature weightings to the CBIRs' configurations in the semantic CBIRn, CBIRs are identified that favor the images containing the semantic concept of the input image in contrast to 'noise'.

Coming back to the overview of the whole query distribution process in figure 5.5, this step is illustrated by the red arrow that points from the 'concept detection' unit to the 'ideal CBIRs configuration' unit. The next step is expressed by the red arrow pointing from the ideal CBIRs configuration unit to the actual 'CBIRs configuration' unit. It represents the fact the query-specific configuration needs to be mapped to those configurations of the CBIRs that are currently enrolled in the CBIRn. The CBIR systems that match best to the query-specific feature weighting are selected for query processing, and the query image is forwarded to the selected CBIR systems. This step is embodied in Figure 5.5 by the red arrow that points from the CBIR's configuration unit to the semantic concept c_2 and from there on splits to several red arrows to the CBIRs α_7, α_8 and α_9.

These CBIRs, α_7, α_8 and α_9, were instantiated in the training step. All of them were instantiated with an ideal configuration considering a semantic concept which was determined automatically from the training images of the semantic concept c_i (see 5.3.1). Finally the result sets are merged and sent back to the user.

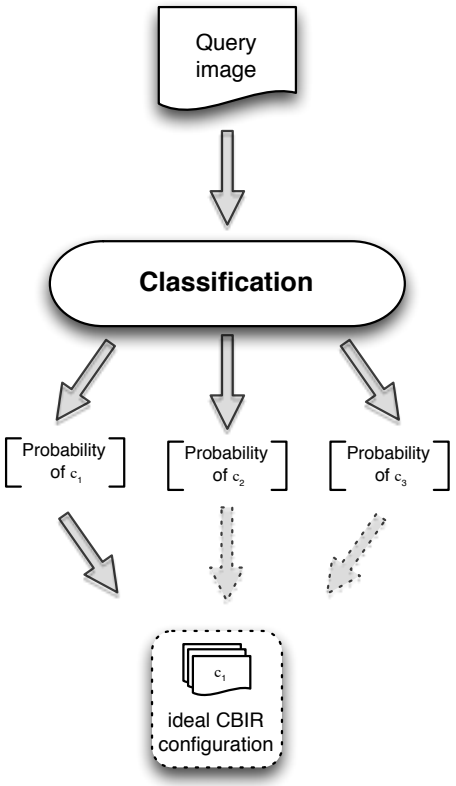


Fig. 5.6.: Query Distribution

5.4. Evaluation

This approach was implemented to prove its feasability. Tests on this implementation were executed using a MacBook Air with an Intel Core i7 2x1.8 Ghz processor, 4 GB 1333 MHz, DDR3 RAM, and Mac OS X Lion 10.7.4 as its operating system.

As this chapter's approach is a first step towards automatic distribution of CBIR queries in a semantic CBIR network, the implementation of this approach began with some narrow domains to scale down the complexity of the problem. The semantic concepts used are: 'animal', 'artifact', 'fungus', 'geoformation', 'person', 'plant', and 'sport', which were selected from Ima-

Fig. 5.7.: Samples of the seven used semantic concepts
Concepts from left to right: animal, artifact, fungus, geoformation, person, plant and sport

geNet[2]. 87,500 images were used for the tests. Samples of these semantic concepts are provided in Figure 5.7.

The individual CBIR systems are represented by different configurations of the open source Lire library[3]. Finally, the semantic concept detection was done using the Weka [49] tool, as already discussed in section 5.2. As this chapter's approach depends heavily on the performance of the concept detection unit, it was designed that any arbitrary technique or even multiple techniques could be used to detect the semantic concepts of the input images. As a result the latest and best performing concept detection technique/s could be integrated in the approach. The SVM has been trained with a total amount of 14000 images, which corresponds to 2000 images per semantic concept. A subsequent evaluation with the remaining 8000 images of each semantic concept revealed a mean precision of 82.29 %, with a standard deviation of 1.94, for the given semantic concepts.

For each of the seven semantic concepts c_i, an equal sized set of images was defined. The maximum number of images per CBIR system was set to 2500 and therefore 35 CBIR instances were used in the test. Thus there were five dedicated CBIRs for each semantic concept. Due to the results that were found in the pre-experiments, as well as it is later shown by the detection performance of Weka, the following fact has to be considered: The enormous amount of images available in the Internet can only be collected by an automatic tool (crawler) that performs some kind of concept detection and these concept detection algorithms might occasionally detect an incorrect semantic concept. Therefore, in order to make the evaluation more realistic a certain rate of 'noise', images of the wrong semantic concept in a CBIRs, has to be considered in the evaluations.

As Weka has a performance of approximately 82 % in this evaluation, the image databases of the CBIRs consist of 80 % of images of the related semantic concept. The last 20 % comprises images containing arbitrary semantic concepts, which is referred to as *noise* in the following explanation. In numbers, each CBIRs' image database contains 2500 images in total (2000 relevant + 500 noise). Ideal configurations are calculated beforehand and assigned to the CBIRs related to the semantic concept it was optimized for. The CBIRs were configured to retrieve a set of at most 20 images.

One hundred images per semantic concept that were not trained at the beginning were manually selected as input images. Hence a total of 700 queries were processed during the tests. The testing procedure was as follows: A query image was passed to the SVM for classification, which assigned a class label to the query image. In the next step the respective ideal feature weightings for this semantic concept were looked up in the central unit. A subsequent euclidian distance calculation between the query-specific feature weighting and the feature weightings of each of the CBIRs' configuration yielded a ranking of the CBIRs in the network. Afterwards,

[2] http://www.image-net.org/
[3] http://www.semanticmetadata.net/lire/

the 5 CBIR systems that were closest to the query-specific feature weighting were selected for query processing. They were queried for the top 20 images. The number of retrieved images of the correct semantic concept are considered successful returned results in this scenario.

The retrieved test results are visualized in Figure 5.8. Two graphs are visualized in figure 5.8 as the test constellations of the semantic CBIRn were divided into two subcategories. In the left graph, the SVM had identified the semantic concept correctly whereas in the right graph the SVM had identified the semantic concept incorrectly. Moreover in these two graphs, two distinct test constellations were tested. 'Semantic CBIRn' is the constellation that was explained in the precedent lines. Whereas, 'CBIRn' means that the used CBIRs did not apply the previously discussed ideal configurations, no concept detection takes place for the input image, the input queries are merely broadcasted to every CBIRs and the result items are merged using round robin. Here the feature weighting of all the features of the CBIR systems were randomly set.

When the semantic concept was correctly identified, which happened in 82 % of the test cases and is visualized in the left graph, it is clearly visible that the semantic CBIRn had the best performance. In 75 % of the queries it returned 90 % to 100 % of images of the same semantic class as the query image. In only a few cases, 20 % to 30 % of noisy images were returned. The 'CBIRn' contrasts strongly with this result. It returned 15 % to 20 % of images of the same semantic class as the query image. This is an excellent result and shows the improvement that can be obtained by the application of a semantic CBIRn in contrast to querying the 'CBIRn'. This very good performance results from two circumstances: First of all, all the CBIR systems in the 'CBIRn' are queried for the query image. As there was no presorting done regarding the semantic concepts, a number of CBIRs that are queried do not even contain images of the wanted semantic concept. Secondly as ***QueDi*** is using round robin in order to merge the returned images, 'CBIRn' returns 35 times 20 images and in order to be comparable to the semantic CBIRn merely the best 100 merged images were examined, therefore solely the best 2 to 3 hits of each CBIRs are considered. In contrast ***QueDi*** considers the 20 best images of the best fitting and specialized CBIRs. Therefore a better result is expected of ***QueDi***.

In the remaining 18 %, the semantic concept was not correctly identified. In this case images are rated as correctly identified if their semantic concept corresponds to the semantic concept that the input image was erroneously assigned to. These results are shown in order to respond to two questions. What happens if an image is assigned to the wrong semantic concept? What happens if a query image does not contain a registered semantic concept?

As visualized in the right graph, merely 50 % of the queries returned 90 % to 100 % of images of the same semantic class as the query image. Another 25 % of the queried returned 78 % to 89 % of images of the same semantic class as the query image. 12.5 % returned between 63 % to 30 % of images of the same semantic class as the query image. The remaining 12.5 %, however, returned between 30 % to 2 % of images of the same semantic class as the query image. In those cases the erroneously assigned query image have, due to the combination of this erroneously assignation and the semantic concept of some of the noise images, favored the semantic concept of images that were originally applied in this CBIRs as noise. But as these cases, where a wrong assignment to the semantic concept happens, are to be eliminated in a further development of the algorithm this causes no problem for these preliminary evaluations. The case for the 'CBIRn' contrasts strongly with these results. In this case, out of all the queries it returned 15 % to 20 % of images were of the same semantic class as the query image. These results show that the concept detection algorithm has a significant impact on the performance of ***QueDi***, as ***QueDi*** strongly increases the number of images of the identified semantic concept in the returned result set. The reasons for this very good performance are the same as the

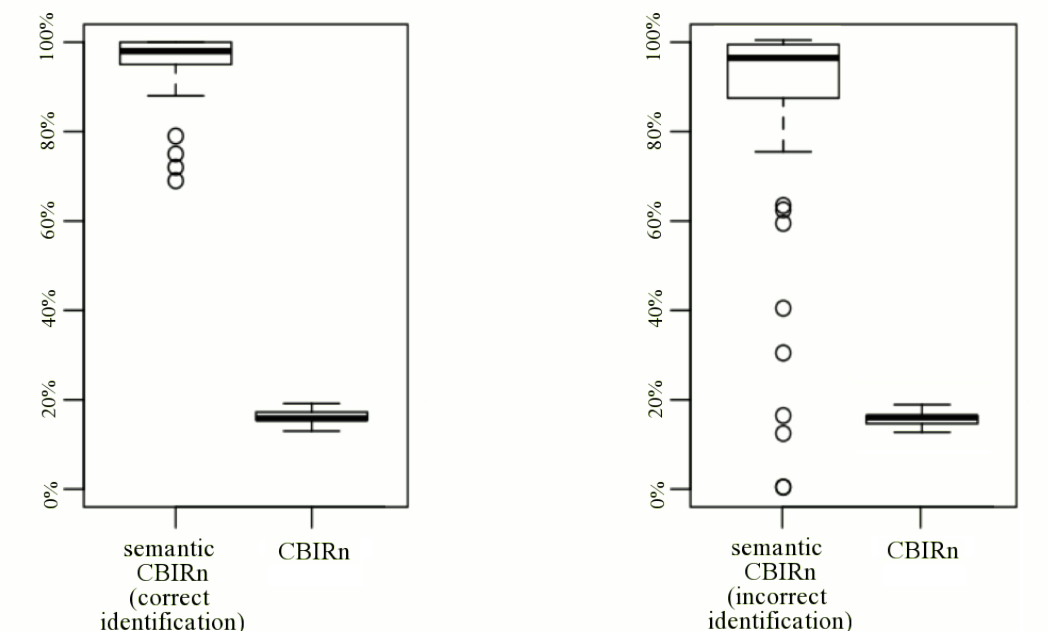


Fig. 5.8.: Retrieval Results
Both graphs show the percentage of returned relevant images to a query using a CBIRs configuration with query-specific feature weighting (l) and uniform feature weighting (r)

reasons two paragraphs earlier.

5.5. Evaluation - Comprehensive Thesis Approach

The second part of the manuscript of this doctoral thesis has so far step by step worked its way through the major problems that arise when trying to provide a comprehensive content based image retrieval search service. Chapter 3 introduced an innovative strategy to merge result sets that were returned in a distributed content based image retrieval scenario. Chapter 4 proposed an algorithm that allows users to analyze and monitor the configuration of a CBIRs and therewith allows them to integrate it to a bigger search infrastructure. Chapter 5 introduced a novel network architecture, including a query distribution strategy, which favors CBIRs that are specialized to the input images' semantic concept over other CBIRs. Eventually all of the highlighted strategies, algorithms and architectures are merged together. Hereby this section comprehensively addresses all of the previously stated research questions that were imposed on this thesis. As the implementation of this thesis (see annex B) is still in a prototypical state , there is unfortunately no API available.

5.5.1. Adaptions

This thesis' approach is visualized in figure 5.9. The next paragraph of this section outlines the interface between ***GeCCo*** and ***QueDi***. A focus will hereby be on the storage strategy of the CBIRs configurations. The interface between ***MeRRSe*** and ***QueDi*** will be discussed in detail in the next paragraph. Thereby, attention will be given to the way the result sets are handed over.

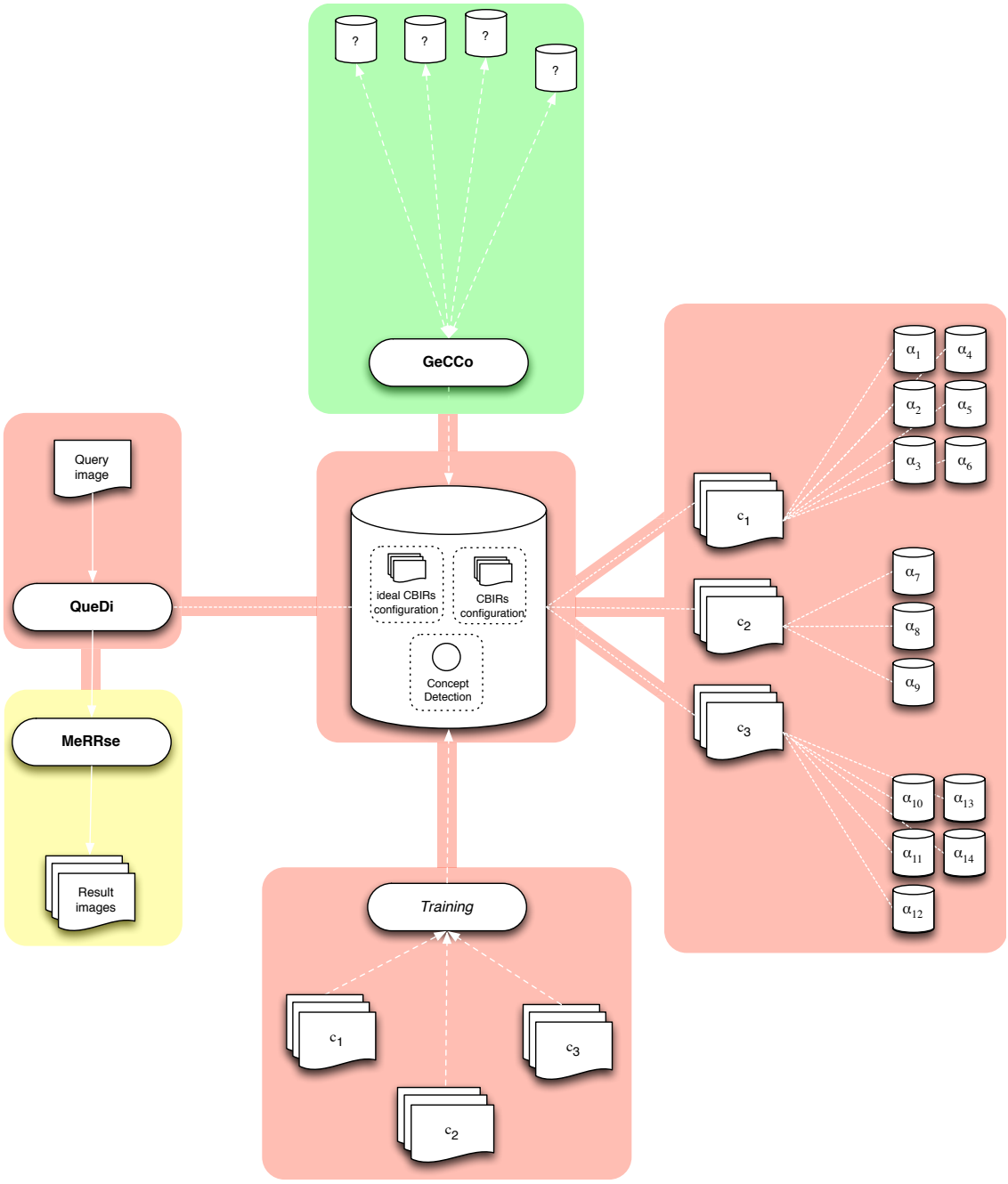


Fig. 5.9.: The Comprehensive Thesis Approach

Figures 5.10 and 5.11 show the evenly addressed interfaces between ***QueDi*** and ***MeRRSe*** as well as between ***GeCCo*** and ***MeRRSe***. Figure 5.10 visualizes the handing over of the returned result sets that were gathered by ***QueDi*** to ***MeRRSe***. There was a small modification applied to the approach that was explained in chapter 5. In chapter 5 the approach returns one single merged result list. Hereby a simple merging algorithm result was used that merges the image results by taking into consideration its scores, not the fact that the configurations of the source CBIRs might differ.

Figure 5.11 visualizes the moment that the external CBIRs is analyzed and the calculated configuration about the external CBIRs. The ***GeCCo*** approach sends the information to ***QueDi*** which again registers the information to the CBIRs. The registered configuration can later be used for the query forwarding.

Figure 5.9 visualizes the interaction of the different approaches of this thesis using color coding. It shows the green GeCCo approach, as well as the yellow ***MeRRSe*** approach, and the red ***QueDi*** approach. Furthermore the previously discussed interfaces are integrated. In this modified implementation the ***QueDi*** approach returns a set of result sets without merging them. As a result the ***MeRRSe*** approach can do what it was built for. This approach merges the returned result sets and returns them to the user.

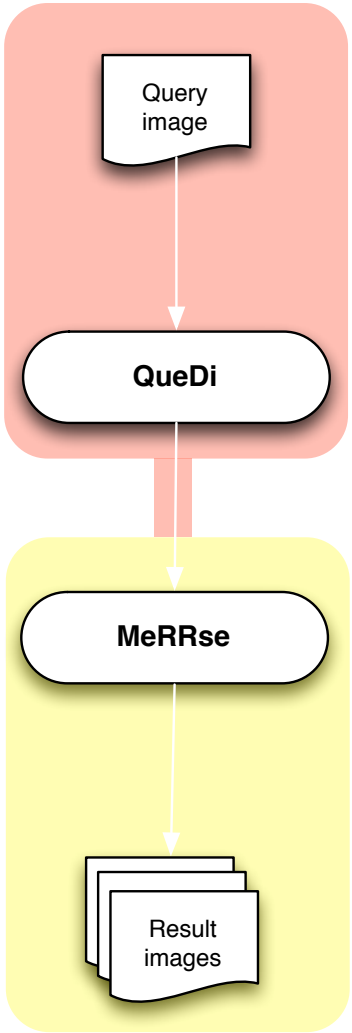


Fig. 5.10.: Interface MeRRSe QueDi

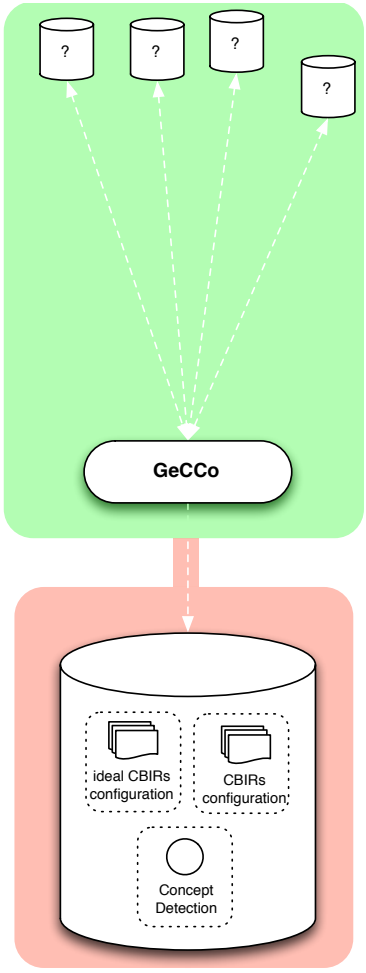


Fig. 5.11.: Interface GeCCo QueDi

The query image enters the approach lefthand side of figure 5.9 and enters ***QueDi***. By subsequently detecting the semantic concept, identifying the ideal configuration for the semantic concept and matching it to the registered CBIR systems' configuration the query image gets distributed to selected CBIRs. The queried CBIRs that the query image is forwarded to send their responses back to the central unit. From there on the gathering of the returned images is forwarded to ***MeRRSe***. Eventually those images are received from the ***MeRRSe*** and rearranged in order return an optimally ranked list

of result images to the user.

One further important adaption had to be done in the test environment generation unit. The code had to be rewritten to ensure from now on the overlap of 30 % that is necessary for ***MeRRSe*** in order to perform properly as already discussed in section 3.4.2. In the beginning of the test environment generation from now on initially this overlap of each CBIRs to at least one CBIRs is defined, afterwards the vacant image slots in the CBIRs are filled. As this overlap was not necessary for the preceding evaluation the overlap was not integrated in the previous implementation.

5.5.2. Evaluation

Experiments were conducted to evaluate the performance of this thesis' approach as a whole. As visualized in figure 5.12, the path that an input query has to take is very similar to the one in chapter 5. However, there are two main differences in the evaluation of chapter 5 and section 5.5. The first difference is that a percentage of 5 % of the CBIRs were analyzed using ***GeCCo***. This percentage was intentionally defined that small as in general the semantic CBIRn was designed to operate with self instantiated CBIRs. Adding external CBIRs was merely made possible for adding of specialized CBIRs occasionally. These 5 % were therefore added in order to simulate these specialized external CBIRs that wish to join the network.

Due to the fact that in this first tests of the system only CBIRs whose features were known to ***GeCCo*** were integrated this did not have an influence on the evaluation results. This limitation was imposed as CBIRs whose features were unknown to ***GeCCo*** could not have been analyzed. The second difference is that the returned results are processed by ***MeRRSe***. This adjustment increased the efficiency of the comprehensive thesis approach in contrast to the ***QueDi***-approach. To make both approaches imperatively comparable, the evaluation of this approach was performed using the same parameters as were used in the evaluation of chapter 5.

The retrieved test results are visualized in Figure 5.13. Three distinct test constellations were tested. 'Comprehensive Thesis Approach' is the constellation that was explained in this chapter. 'Semantic CBIRn' is the constellation that was explained in chapter 5. Lastly, 'CBIRn' means that the instantiated CBIRs did not apply the ideal configurations, no concept detection takes place for the input image, and the input queries are merely broadcasted to every CBIRs. Here weightings of all the features of the CBIR systems were randomly set. As the importance and the performance of the concept detection algorithm was already discussed in section 5.4 a further presentation and discussion is not given in this section. Therefore figure 5.13 shows only the retrieval results for the cases that the correct semantic concept of the input image was detected.

When the comprehensive thesis approach was applied, which is visualized in the left side of the graph, it is clearly visible that it had the best performance. In 100 % of the queries it returned 90 % to 100 % of images of the same semantic class as the query image. When the semantic CBIRn was applied, which is visualized in the middle of the graph, it is visible that the semantic CBIRn had a very good performance as well. In 75 % of the queries it returned 90 % to 100 % of images of the same semantic class as the query image. In only a few cases, 20 % to 30 % of noise images were returned. The 'CBIRn' contrasts strongly with this result. In the queries it returned only 15 % to 20 % of images of the same semantic class as the query

image. As in the precedent section this very good performance is due to the fact that all the CBIR systems in the 'CBIRn' are queried for the query image and merely the best 100 merged images were examined for this evaluation.

5.6. Conclusion

Prior work has documented the feasibility of content based image retrieval networks: In [6], for example, a peer-to-peer CBIRs network is designed that was inspired by the Metric Social Network (MSN) and therefore uses the principles of social networking to support the self-organization and self-adaptability of the network. However, these studies have not paid attention to the heterogeneity of the associated CBIRs or did not consider the semantic concepts

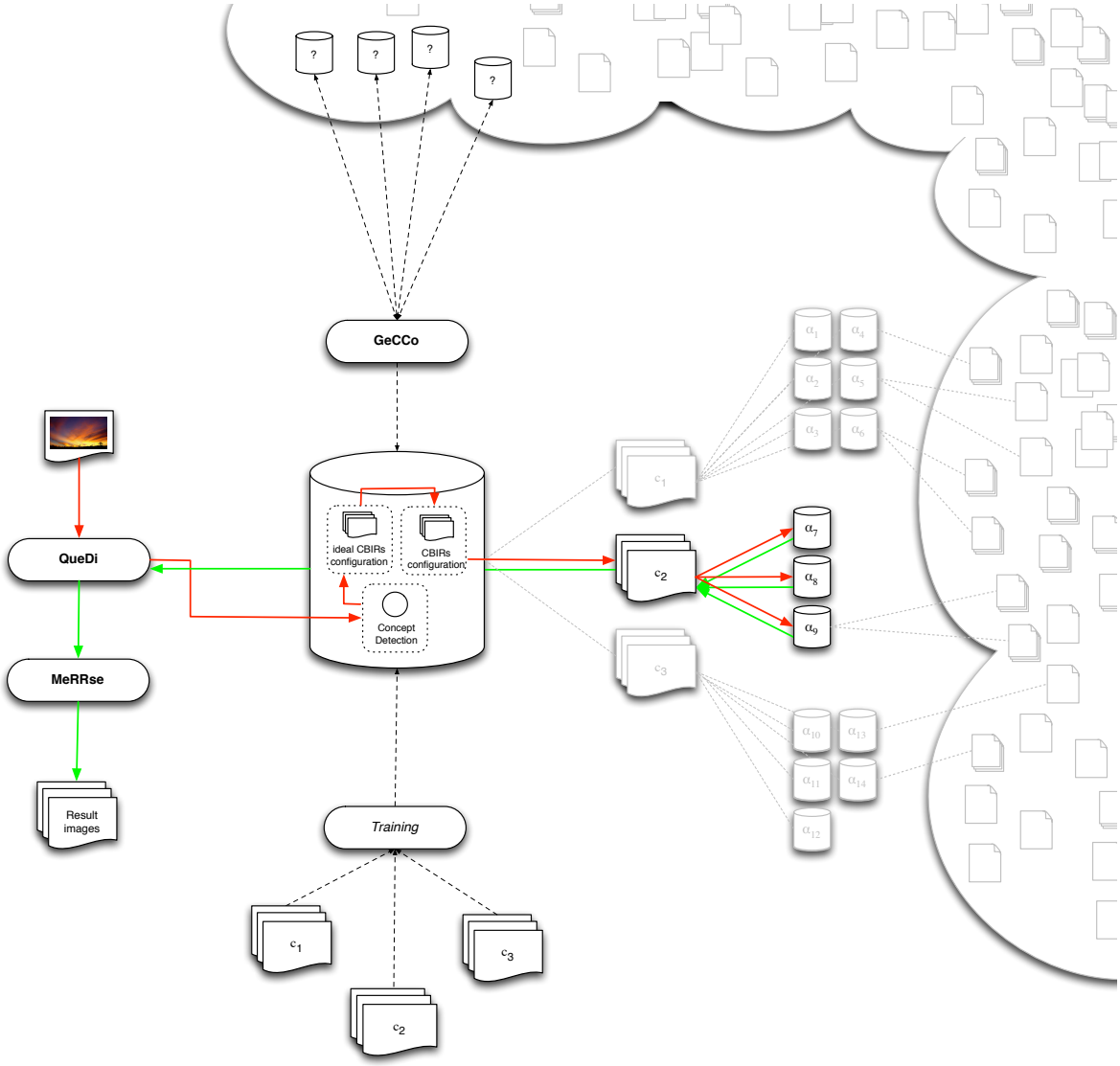


Fig. 5.12.: The path of a query in the Comprehensive Thesis Approach

involved. In this chapter, an approach that makes a step towards the automatic distribution of CBIR queries in a semantic CBIR network was presented, which took advantage of the heterogeneity of the associated CBIRs and the semantic concepts involved.

Chapter 5 has documented the feasibility of ***QueDi*** with all its' key elements as a content based image retrieval network that takes into consideration the heterogeneity of the associated CBIRs and the semantic concepts involved, called semantic CBIRn. This semantic CBIR network is furthermore open to external CBIRs and merges the returned results in an advanced way. It was furthermore shown that a notable benefit can be obtained from this construction. When the input image's concept is detected, by the concept detection unit, the query can be forwarded, using a matching of the ideal CBIRs configuration to the registered CBIRs configuration, to specialized CBIRs and therefore images that would not be of any use for the user can be omitted beforehand. Even a possible contamination of the content of the specialized CBIRs - i.e. images whose semantic concept do not fit to the semantic concept the CBIRs is specialized for - can be coped with by ***QueDi***.

However implementational details, like the extraction of multiple semantic concepts, have to be improved before the prototypical implementation (see GUI in annex B) of this comprehensive thesis approach could be made publicly available and a practical use could be derived for the final user.

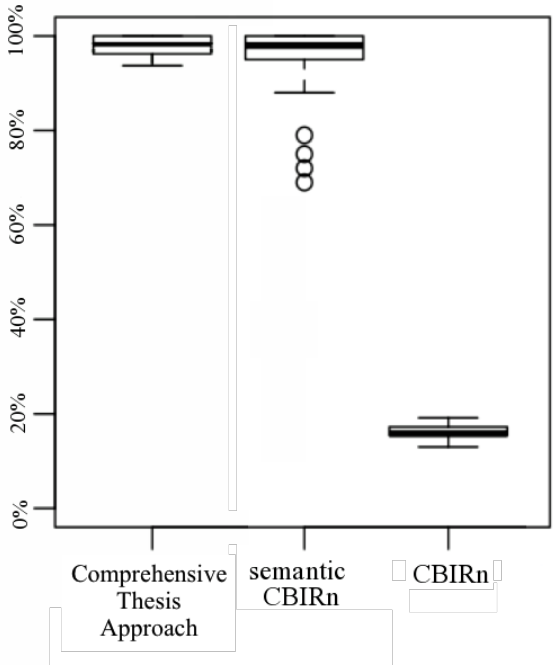


Fig. 5.13.: Retrieval Results
Showing the number of returned relevant images to a query using the Comprehensive Thesis Approach (left), semantic CBIRn of chapter 5 (middle) and CBIRn (right)

6. Conclusion & Future Work

This thesis was organized as follows. Chapter 2 focused on the state of the art condition of the subject. Subsequently, chapters 3 to 5, described the approach that was developed in this thesis. This approach provides solutions for the following research questions, which were established in chapter 1:

- **How can a semantic analysis and modeling of a CBIRn be implemented?** - The semantic CBIRn that was specified in this thesis is a CBIRn that is in fact internally subdivided into semantic concepts. Currently, these semantic concepts are manually predefined as current machine learning algorithms are not yet capable of judging which classes of images are seen by humans as a semantic concept. In turn, these semantic concepts are attached themselves to CBIRs.

  This specification has resulted in a clear structure of the network which becomes of major importance when answering the question of the query forwarding. However, it is not only the structure of the semantic CBIRn that is of major importance. The information about the configuration of each individual CBIRs, as well as the ideal configurations of a CBIRs for semantic concepts which were calculated in the training process are important information to select the CBIRs that matches best to reply to a query image.

- **How can the configuration of an unknown CBIRs be detected?** - In the, so called, ***GeCCo*** approach that was developed in this thesis, a CBIRs whose configuration was to be detected was queried by a set of predefined images. The CBIRs in question had to reply to every single query image with a ranked list of result images and their associated scores. Subsequently, a configuration was approximated that generated scores that were as similar as possible to the result sets' scores. Finally, similarity score rating values were assigned to every feature to enable the filtering of irrelevant features.

  Five different classes of cases have been highlighted depending on the algorithm's knowledge about the features that were used by the CBIRs that has to be analyzed. ***GeCCo*** has the ability to compute the applied features and their weights in two of the classes of cases. In those cases the correct features have been identified with an average precision of 98 % and a recall of 91 % over all the applied algorithms. In the three remaining classes of cases ***GeCCo*** can merely detect and inform that it is not able to determine the CBIRs configuration. In those cases in average over all the applied algorithms in 93 % of those cases it was correctly detected that it is not able to determine the CBIRs configuration. It was moreover demonstrated that the problem can be traced back to an optimization problem. Eventually in the evaluation section ***GeCCo*** showed a high accuracy for feature-weight detection and the capability of the system to identify cases that cannot be dealt with a good precision. Hereby it was shown that Cuckoo Search can overall be considered as the best algorithm in the applied evaluational set up as it was slightly more accurate than the other algorithms.

- **How can a query be exclusively forwarded to relevant CBIRs?** - In the context of the

semantic CBIRn that was defined in this thesis, the, so called, ***QueDi*** approach was implemented to forward queries exclusively to relevant CBIRs. Using a concept detector, that is trained with the used semantic concepts and is deployed in the central unit of the semantic CBIRn, each input images' concept is detected. By making use of the information about the detected semantic concept of the query image the query is solely forwarded to suitable specialized CBIRs. By applying this strategy, images that are dissimilar to the query image's semantic concept are omitted beforehand.

The feasibility of ***QueDi*** with all it's key elements as a Content Based Image Retrieval network that takes into consideration the heterogeneity of the associated CBIRs and the semantic concepts involved has been documented. It has been furthermore shown that a real benefit can be derived from this construction. Hereby, an augmentation of the number of returned images that are of the same semantic concept as the input images is achieved by a factor of 4.0. When the input image's concept is detected, by the concept detection unit, the query can be forwarded, using a matching of the ideal CBIRs configuration to the registered CBIRs configuration, to specialized CBIRs and therefore images that would not be of any use for the user can be omitted beforehand. Even a possible contamination of the content of the specialized CBIRs can be coped with by ***QueDi***.

- **How can the results of individual CBIRs be optimally merged?** - In the, so called, ***MeRRSe*** approach that was developed in this thesis, several CBIRs reply individually with a result-set of images they consider to be the best fit to the input image from the total, of all the images in their possession. These result sets' items are furnished with an identifier, a score, and a rank. Each of these characteristics naturally relies on the comparison to the input image and its own image.

  These result sets are subsequently forwarded to the next stage and processed using a linear regression analysis. The regression analysis takes hereby advantage of existing duplicates in these result sets. These duplicates are mandatory for this approach. The regression analysis process subsequently compares the scores of the duplicates in two of the used CBIRs and can hereby calculates a factor that makes the result item's scores comparable. Using this factor, not only are the duplicate scores matched, but every score of every result item in the result set is corrected. In the end of this linear regression step, result sets are finally generated that are comparable. All the items are then merged and a pre-assigned number of result items is skimmed off. This amount of images is now recalculated using an internal CBIRs. Finally a result set is built using these best result items. An improvement of up to 12 % the results was demonstrated compared to a solely application of CombMAX.

This thesis tried to answer each of the previously listed research questions and realized a prototypical implementation. This was rendered possible by bringing a number of heterogeneous specialized CBIRs to work together in a semantic CBIRn. The current approach of the semantic CBIRn - as mentioned in the beginning of this thesis - does not intend to solve the problem of the semantic gap, but allows users to more efficiently search the mass of images that are available in the network. This is possible thanks to the integration and generation of CBIRs that are each specialized on one specific semantic concept. Even integration of external CBIRs is made possible as their configuration is analyzed and stored for the query process. This integration step is necessary as the configuration of a CBIRs, normally consisting of a selection of features, metrics, and weights, is different for each CBIRs to match the challenge of ranking

domain specific images. Inter alia, thanks to the additional information obtained from the integration step, it is subsequently possible to forward the queries inside the semantic CBIRn in an intelligent way. Hence discrepancies of the returned result list that are caused by the different configurations of the involved CBIRs, are compensated for and their returned results are rendered comparable.

If now, as was the case in chapter 1, a user is involved in a process that requires images and s/he has no adequate images to hand - e.g. the student that needs a superior image of a chromosome to decorate the cover page of an assignment or the editor-in-chief that is not provided adequate images to illustrate her/-is articles perfectly - none of them would take advantage of a search engine that operates on keywords in order to acquire their image.

Which means for the student s/he would not launch Google's[1], Bing's[2] or Baidu's[3] keyword-based image search and browse for a license free replacement for the chromosome image that s/he wanted to use. The editor-in-chief would not log in to Fotolia[4] and scan for fitting images. As both are in possession of images that nearly meet their requirements, both of them would search using content-based criteria in a semantic CBIRn.

The semantic CBIRn developed thus far will in future work be revised in the following aspects:

- **Multiple semantic concepts in an Image** - A strategy has to be developed to treat multiple semantic concepts in an image. In this way images could be attached to multiple semantic concepts and their reference would be registered in multiple CBIRs for different semantic concepts. This fact should result in an increased overlap between the images of the semantic concepts, which is required by MeRRSe.

- **Improvement of the Enrolling Process** - The component in the comprehensive thesis approach which looks after the configurations of externally integrated CBIRs was discussed in chapter 4. Five different cases of feature constellations that could happen when analyzing a CBIRs were explained. Two of these cases resulted, thanks to a successful analysis, in a possible integration of the CBIRs in the semantic CBIRn. Three of these cases resulted in a rejection of the CBIRs. A new methodology has to be developed for these three cases in order to successfully integrate these CBIRs to the semantic CBIRn. Future research could therefore consider statistical analysis in these three cases according to the correlation of specific features [34] in order to analyze the CBIRs in practice. Hereby features could be classified in feature classes and subsequently compered instead of a mere comparison of identical features.

- **Improving the Merging by Using the Knowledge about the CBIRs' Configuraiton** By taking advantage of the knowledge about the CBIRs' configuration, the merging of the result items returned by the queried CBIRs is expected to be able to be improved. Rating or scoring of result items could be increased or reduced, depending on whether the CBIRs' configuration is adapted to the image concepts.

[1] `http://www.google.com`
[2] `http://www.bing.com`
[3] `http://www.baidu.com`
[4] `http://www.fotolia.com`

Bibliography

[1] Proceedings of the International Conference on Scientific Information, 1958.

[2] Dublin core metadata element set, version 1.1. webpage, Oct. 2010.

[3] C. C. Aggarwal. Towards systematic design of distance functions for data mining applications. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 9–18, New York, NY, USA, 2003. ACM.

[4] F. Anklesaria and M. McCahill. The Internet Gopher. In A. Heck and F. Murtagh, editors, *Intelligent Information Retrieval: The Case of Astronomy and Related Space Sciences*, volume 182 of *Astrophysics and Space Science Library*, pages 119–125. Springer Netherlands, 1993.

[5] P. R. Bagley. *Electronic Digital Machines for High-Speed Information Searching*. PhD thesis, 1951.

[6] S. Barton, V. Dohnal, J. Sedmidubsky, and P. Zezula. Building self-organized image retrieval network. In *Proceeding of the 6th ACM workshop on Large-Scale distributed systems for information retrieval (LSDS-IR)*, pages 51–58, New York City, New York, United States of America, 2008. ACM.

[7] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural computation*, 12(10):2385–2404, 2000.

[8] J. Becker and R. M. Hayes. *Information storage and retrieval: tools, elements, theories*. Information sciences series. Wiley, 1967.

[9] C. Beecks, M. S. Uysal, and T. Seidl. Signature quadratic form distances for content-based similarity. In *ACM Multimedia*, pages 697–700, 2009.

[10] S. Berretti, A. D. Bimbo, and P. Pala. Merging results of distributed image libraries, 2003.

[11] S. Berretti, A. Del Bimbo, and P. Pala. Merging results for distributed content based image retrieval. *Multimedia Tools and Applications*, 24(3):215–232, 2004.

[12] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022, Mar. 2003.

[13] S. Boag, D. Chamberlin, M. F. Fernández, D. Florescu, J. Robie, and J. Siméon. XQuery 1.0: An XML Query Language, 2007.

[14] D. Borth and A. Ulges. Automatic concept-to-query mapping for web-based concept detector training. In *Proceedings of the 19th ACM international conference on Multimedia (ACM MM)*, pages 1453–1456, Scottsdale, Arizona, United States of America, 2011. ACM.

[15] K. Bosch. *Formelsammlung Statistik*. Oldenbourg, 2003.

[16] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2559–2566, June 2010.

[17] V. Bush. As we may think. *interactions*, 3(2):35–46, 1945.

[18] T. Caesar, J. Gloger, and E. Mandler. Preprocessing and feature extraction for a handwriting recognition system. In *Document Analysis and Recognition, 1993., Proceedings of the Second International Conference on*, pages 408–411, Oct 1993.

[19] P. P. K. Chan, Z.-C. Huang, W. W. Y. Ng, and D. S. Yeung. Dynamic hierarchical semantic network based image retrieval using relevance feedback. In *Proceedings of the 5th International Conference on Machine Learning and Cybernetics (ICMLC)*, pages 1746–1751, Guilin, Republic of China, 2011. IEEE.

[20] S.-F. Chang, J. R. Smith, M. Beigi, and A. Benitez. Visual information retrieval from large distributed online repositories. *Commun. ACM*, 40:63–71, Dec. 1997.

[21] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *Proceedings of the 20th Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, Minneapolis, Minnesota, United States of America, 2007. IEEE.

[22] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences and Trends. *ACM Computing Surveys (CSUR)*, 40(2):1–60, 2008.

[23] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[24] J. Deng, A. Berg, K. Li, and L. Fei-Fei. What does classifying more than 10,000 image categories tell us? *Computer VisionECCV 2010*, pages 71–84, 2010.

[25] T. Deselaers, D. Keysers, and H. Ney. Features for image retrieval: an experimental comparison. *Information Retrieval (IR)*, 11(2):77–107, 2007.

[26] C. Ding and H. Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology*, 3(02):185–205, 2005.

[27] K. Doi, S. Katsuragawa, J. Morishita, and X.-W. Xu. Computer-aided method for automated image feature analysis and diagnosis of medical images, Aug. 4 1998. US Patent 5,790,690.

[28] M. Döller, F. Stegmaier, H. Kosch, R. Tous, and J. Delgado. Standardized interoperable image retrieval. In *Proceedings of the 25th ACM Symposium on Applied Computing (SAC)*, number January, pages 880–886, Sierre, Switzerland, 2010. ACM.

[29] M. Döller, R. Tous, M. Gruhne, K. Yoon, M. Sano, and I. S. Burnett. The MPEG Query Format: Unifying Access to Multimedia Retrieval Systems. *IEEE Multimedia*, 15(4):82–95, 2008.

[30] C. Dorai and S. Venkatesh. Bridging the semantic gap with computational media aesthetics. *IEEE multimedia*, 10(2):15–17, 2003.

[31] F. Dufaux, M. Ansorge, and T. Ebrahimi. Overview of JPSearch: A standard for image search and retrieval. In *Content-Based Multimedia Indexing, 2007. CBMI'07. International Workshop on*, pages 138–143. IEEE, 2007.

[32] R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Proceedings of the 6th International Symposium on Micro Machine and Human Science (MHS'95)*, pages 39–43, 1995, IEEE.

[33] M. Ehrig. *Ontology alignment: bridging the semantic gap*, volume 4. Springer, 2006.

[34] H. Eidenberger. Evaluation of content-based image descriptors by statistical methods. *Multimedia Tools and Applications (MTA)*, 35(3):241–258, Apr. 2007.

[35] R. El-Hajj, L. Likforman-Sulem, and C. Mokbel. Arabic handwriting recognition using baseline dependant features and hidden markov modeling. In *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, pages 893–897 Vol. 2, Aug 2005.

[36] A. Emtage and P. Deutsch. Archie: An electronic directory service for the internet. In *Proceedings of the Winter 1992 USENIX Conference*, pages 93–110, 1992.

[37] P. Enser and C. Sandom. Towards a comprehensive survey of the semantic gap in visual image retrieval. In E. Bakker, M. Lew, T. Huang, N. Sebe, and X. Zhou, editors, *Image and Video Retrieval*, volume 2728 of *Lecture Notes in Computer Science*, pages 291–299. Springer Berlin Heidelberg, 2003.

[38] P. Enser and C. Sandom. Towards a comprehensive survey of the semantic gap in visual image retrieval. In *Proceedings of the 2Nd International Conference on Image and Video Retrieval*, CIVR'03, pages 291–299, Berlin, Heidelberg, 2003. Springer-Verlag.

[39] C. Faloutsos, W. Equitz, M. Flickner, W. Niblack, D. Petkovic, and R. Barber. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3:231–262, 1994.

[40] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[41] Y. Feng and P. Shi. Face detection based on kernel fisher discriminant analysis. In *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, pages 381–384. IEEE, 2004.

[42] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the 16th Conference on Computer Vision and Pattern Recognition*, pages 264–271, Madison, Wisconsin, United States of America, 2003. IEEE.

[43] R. A. FISHER. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936.

[44] E. A. Fox and J. A. Shaw. Combination of Multiple Searches. In D. K. Harman, editor, *The 2nd Text Retrieval Conference TREC2 NIST SP 500215*, volume 500-215 of *NIST Special Publication*, pages 243–252. NIST, National Institute for Standards and Technology, NIST Special Publication 500215, 1994.

[45] G. Frederix, G. Caenen, and E. Pauwels. Panoramic, adaptive and reconfigurable interface for similarity search. In *Image Processing, 2000. Proceedings. 2000 International Conference on*, volume 3, pages 222 –225 vol.3, 2000.

[46] M. L. Giger, K. Doi, and H. MacMahon. Image feature analysis and computer-aided diagnosis in digital radiography. 3. automated detection of nodules in peripheral lung fields. *Medical Physics*, 15(2):158–166, 1988.

[47] G. G. Gordon. Face recognition based on depth and curvature features. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on*, pages 808–810. IEEE, 1992.

[48] P. Gupta and K. Johari. Implementation of Web Crawler. In *Emerging Trends in Engineering and Technology (ICETET), 2009 2nd International Conference on*, pages 838 –843, Dec. 2009.

[49] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations (SIGKDD)*, 11(1):10–18, 2009.

[50] J. S. Hare, P. H. Lewis, P. G. B. Enser, and C. J. Sandom. Mind the gap: Another look at the problem of the semantic gap in image retrieval. In *Proceedings of Multimedia Content Analysis, Management and Retrieval 2006 SPIE*, 2006.

[51] J. S. Hare, P. A. S. Sinclair, P. H. Lewis, K. Martinez, P. G. B. Enser, and C. J. S. Bridging the semantic gap in multimedia information retrieval: Top-down and bottom-up approaches. In *In 3rd European Semantic Web Conference (ESWC-06), LNCS 4011*. Springer Verlag, 2006.

[52] A. S. Hornby. *Oxford Advanced Learner's Dictionary*, volume 41. Oxford University Press, 2005.

[53] C.-M. Huang, K.-c. Lan, and C.-Z. Tsai. A Survey of Opportunistic Networks. *22nd International Conference on Advanced Information Networking and Applications - Workshops (aina workshops 2008)*, pages 1672–1677, 2008.

[54] Y.-P. Huang, S.-W. Luo, and E.-Y. Chen. An efficient iris recognition system. In *Machine Learning and Cybernetics, 2002. Proceedings. 2002 International Conference on*, volume 1, pages 450–454. IEEE, 2002.

[55] D. P. Huttenlocher, G. A. Klanderman, G. A. Kl, and W. J. Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:850–863, 1993.

[56] F. Jing, M. Li, H.-J. Zhang, and B. Zhang. A unified framework for image retrieval using keyword and visual features. *Image Processing, IEEE Transactions on*, 14(7):979–989, July 2005.

[57] D. Joshi, J. Z. Wang, and J. Li. The Story Picturing Engine—a system for automatic text illustration. *ACM Transactions on Multimedia Computing Communications and Applications*, 2(1):68–89, 2006.

[58] A. Kent. *Information analysis and retrieval*. Information sciences series. Becker and Hayes, 1971.

[59] A. Kent, M. M. Berry, F. U. Luehrs Jr., and J. W. Perry. Machine literature searching VIII. Operational criteria for designing information retrieval systems. *American Documentation*, 6(2):93–101, 1955.

[60] M. L. Kherfi, D. Ziou, and A. Bernardi. Image retrieval from the world wide web: Issues, techniques, and systems. *ACM Computing Surveys (CSUR)*, 36(1):35–67, 2004.

[61] C. A. Knoblock. Searching the world wide web. *IEEE Expert: Intelligent Systems and Their Applications*, 12(1):8–14, 1997.

[62] A. L. Koerich, R. Sabourin, and C. Y. Suen. Large vocabulary off-line handwriting recognition: A survey. *Pattern Analysis & Applications*, 6(2):97–121, 2003.

[63] V. H. Kondekar, V. S. Kolkure, and S. N. Kore. Image Retrieval Techniques based on Image Features, A State of Art approach for CBIR. *Journal of Computer Science*, 7(1):69–76, 2010.

[64] H. Kosch and P. Maier. Content-Based Image Retrieval Systems - Reviewing and Benchmarking. *Journal of Digital Information Management*, 8(1):54–64, 2010.

[65] T. Kurita and T. Taguchi. A modification of kernel-based fisher discriminant analysis for face detection. In *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*, pages 300–305. IEEE, 2002.

[66] D. Kuropka. *Modelle zur Repräsentation natürlichsprachlicher Dokumente: Ontologie-basiertes Information-Filtering-und-Retrieval mit relationalen Datenbanken*. Advances in information systems and management science. Logos-Verl., 2004.

[67] F. W. Lancaster and E. G. Fayen. *Information retrieval: on-line [by] F. W. Lancaster and E. G. Fayen*. Melville Pub. Co Los Angeles, 1973.

[68] J.-E. Lee, R. Jin, A. K. Jain, and W. Tong. Image Retrieval in Forensics: Tattoo Image Database Application. *Multimedia in Forensics Security and Intelligence (MiFor)*, 19(1):2–11, 2012.

[69] J. H. Lee. Analyses of multiple evidence combination. *Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval SIGIR 97*, 31(SI):267–276, 1997.

[70] Y. Li, S. Gong, and H. Liddell. Recognising trajectories of facial identities using kernel discriminant analysis. *Image and Vision Computing*, 21(13):1077–1086, 2003.

[71] J. C. R. Licklider, C. on Library Resources, Bolt Beranek, and inc Newman. *Libraries of the future*. M.I.T. Press, 1965.

[72] Y. Linde, A. Buzo, and R. M. Gray. An Algorithm for Vector Quantizer Design. *IEEE Transactions on Communications*, 1980.

[73] Q. Liu, R. Huang, H. Lu, and S. Ma. Face recognition using kernel-based fisher discriminant analysis. In *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*, pages 197–201. IEEE, 2002.

[74] Q. Liu, H. Lu, and S. Ma. Improving kernel fisher discriminant analysis for face recognition. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(1):42–49, 2004.

[75] D. Lu and Q. Weng. A survey of image classification methods and techniques for improving classification performance. *International journal of Remote sensing*, 28(5):823–870, 2007.

[76] H. P. Luhn. *Auto-encoding of Documents for Information Retrieval Systems*. IBM Research Center, 1958.

[77] M. Lux and S. A. Chatzichristofis. LIRe: Lucene Image Retrieval - An Extensible Java CBIR Library. In *Proceeding of the 16th ACM international*, pages 1085–1087, Vancouver, British Columbia, Canada, 2008. ACM.

[78] H. M˙ A review of content-based image retrieval systems in medical applicationsclinical benefits and future directions. *International Journal of Medical Informatics*, 73(1):1 – 23, 2004.

[79] L. Ma, T. Tan, Y. Wang, and D. Zhang. Efficient iris recognition by characterizing key local variations. *Image Processing, IEEE Transactions on*, 13(6):739–750, 2004.

[80] L. Ma, Y. Wang, and T. Tan. Iris recognition using circular symmetric filters. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 2, pages 414–417. IEEE, 2002.

[81] J. Maintz and M. A. Viergever. A survey of medical image registration. *Medical image analysis*, 2(1):1–36, 1998.

[82] B. Manjunath, R. Chellappa, and C. Von der Malsburg. A feature based approach to face recognition. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR '92., 1992 IEEE Computer Society Conference on*, pages 373–378, Jun 1992.

[83] M. E. Maron and J. L. Kuhns. On Relevance, Probabilistic Indexing and Information Retrieval. *Journal of the ACM*, 7(3):216–244, 1960.

[84] O. Marques and B. Furht. *Distributed multimedia databases: {T}echniques and applications*, chapter 3: Content, pages 37–57. Idea Group Inc (IGI), 1 edition, 2002.

[85] J. M. Martinez, R. Koenen, and F. Pereira. {MPEG-7}: {T}he generic multimedia content description standard, part 1. *Multimedia, IEEE*, 9(2):78–87, Apr. 2002.

[86] J. Melton and A. Eisenberg. SQL multimedia and application packages (SQL/MM). *Sigmod Record*, 30(4):97–102, 2001.

[87] J. Melton and A. Eisenberg. Sql multimedia and application packages (sql/mm). *ACM Sigmod Record*, 30(4):97–102, 2001.

[88] D. Metzler and W. Bruce Croft. Linear feature-based models for information retrieval. *Inf. Retr.*, 10(3):257–274, June 2007.

[89] D. E. Millard, N. M. Gibbins, D. T. Michaelides, and M. J. Weal. Mind the semantic gap. In *Proceedings of the Sixteenth ACM Conference on Hypertext and Hypermedia*, HYPERTEXT '05, pages 54–62, New York, NY, USA, 2005. ACM.

[90] M. Montague and J. A. Aslam. Relevance score normalization for metasearch. In *Proceedings of the Tenth International Conference on Information and Knowledge Management*, CIKM '01, pages 427–433, New York, NY, USA, 2001. ACM.

[91] C. N. Mooers. Information retrieval viewed as temporal signaling. In *Proceedings of the International Congress of Mathematicians*, pages 572–573, 1950.

[92] H. Müller and P. Clough. The ImageCLEF benchmark on Multimodal, Multilingual Visual Image Retrieval. *IAPR Newsletter*, 28(2):13–17, 2006.

[93] H. Müller, N. Michoux, D. Bandon, and A. Geissbuhler. A review of content-based image retrieval systems in medical applicationsclinical benefits and future directions. *International Journal of Medical Informatics*, 73(1):1 – 23, 2004.

[94] M. Nabti and A. Bouridane. An effective and fast iris recognition system based on a combined multiscale feature extraction technique. *Pattern Recognition*, 41(3):868–879, 2008.

[95] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7(4):308–313, 1965.

[96] A. Neumaier. Solving ill-conditioned and singular linear systems: A tutorial on regularization. *Siam Review*, 40(3):636–666, 1998.

[97] T. OReilly. What is Web 2.0: Design patterns and business models for the next generation of software. *Communications & strategies*, (1):17, 2007.

[98] C. F. J. Overhage, R. J. Harman, and M. I. of Technology. *Intrex: report of a Planning Conference on Information Transfer Experiments, September 3, 1965*. M.I.T. Press, 1965.

[99] G. P. Penney, J. Weese, J. A. Little, P. Desmedt, D. L. Hill, and D. J. Hawkes. A comparison of similarity measures for use in 2-d-3-d medical image registration. *Medical Imaging, IEEE Transactions on*, 17(4):586–595, 1998.

[100] A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*, pages 84–91, Jun 1994.

[101] D. L. Pham, C. Xu, and J. L. Prince. Current methods in medical image segmentation 1. *Annual review of biomedical engineering*, 2(1):315–337, 2000.

[102] P. J. Phillips, P. J. Flynn, T. Scruggs, K. W. Bowyer, and W. Worek. Preliminary face recognition grand challenge results. In *Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition (FGR)*, pages 15–24, Southampton, United Kingdom. 2006. IEEE.

[103] D. Picard and M. Cord. Performances of Mobile-Agents for Interactive Image Retrieval. In *Proceedings of ACM International Conference on Web Intelligence (WI)*, pages 581–586, Hong Kong, Hong Kong. 2006. IEEE.

[104] J. Platt et al. Sequential minimal optimization: A fast algorithm for training support vector machines. 1998.

[105] K. Polat, S. Sahan, H. Kodaz, and S. Günes. A new classification method for breast cancer diagnosis: feature selection artificial immune recognition system (fs-airs). In *Advances in Natural Computation*, pages 830–838. Springer, 2005.

[106] A. C. R. Project and C. W. Cleverdon. *Report on the testing and analysis of an investigation into the comparative efficiency of indexing systems*. College of Aeronautics, 1962.

[107] E. Riloff and L. Hollaar. Text databases and information retrieval. *ACM Comput. Surv.*, 28(1):133–135, Mar. 1996.

[108] S. E. Robertson and K. Sparck Jones. Document retrieval systems. chapter Relevance Weighting of Search Terms, pages 143–160. Taylor Graham Publishing, London, UK, UK, 1988.

[109] F. B. Rogers. The Development of MEDLARS. *Bulletin of the Medical Library Association*, 52:150–151, 1964.

[110] Y. Rubner, J. Puzicha, C. Tomasi, and J. M. Buhmann. Empirical evaluation of dissimilarity measures for color and texture. *Computer Vision and Image Understanding*, 84(1):25–43, 2001.

[111] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

[112] Y. Rui, T. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: A power tool for interactive content-based image retrieval. *Circuits and Systems for Video Technology, IEEE Transactions on*, 8(5):644 –655, Sept. 1998.

[113] H. Said, T. Tan, and K. Baker. Personal identification based on handwriting. *Pattern Recognition*, 33(1):149 – 160, 2000.

[114] G. Salton. The SMART System – Retrieval Results and Future Plans, 1966.

[115] G. Salton. Automatic Information Organization and Retrieval. *Journal Of Experimental Psychology. Learning Memory And Cognition*, 9(3):430–439, 1968.

[116] G. Salton. The smart document retrieval project. In *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '91, pages 356–358, New York, NY, USA, 1991. ACM.

[117] G. Salton, E. A. Fox, and H. Wu. Extended boolean information retrieval. *Commun. ACM*, 26(11):1022–1036, Nov. 1983.

[118] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, Nov. 1975.

[119] J. W. Sammon. A Nonlinear Mapping for Data Structure Analysis. *IEEE Transactions on Computers*, C-18(5):401–409, 1969.

[120] J. W. Sammon and N. Y. United States. Air Development Centre Rome. *Some Mathematics of Information Storage and Retrieval*. United States. Air Development Center, Rome, N.Y. TEchnical report. Clearinghouse for Federal Scientific and Technical Information, 1968.

[121] B. Scholkopft and K.-R. Mullert. Fisher discriminant analysis with kernels. 1999.

[122] J. Sedmidubsky, V. Dohnal, S. Barton, and P. Zezula. A Self-Organized System for Content-Based Search in Multimedia. In *Proceedings of the 10th International Symposium on Multimedia (ISM)*, pages 322–327, Berkeley, California, United States of America, 2008. IEEE.

[123] T. Seymour, D. Frantsvog, and S. Kumar. History of search engines. *International Journal of Management & Information Systems*, 15(4), 2011.

[124] A. Singhal. Modern Information Retrieval : A Brief Overview. *IEEE Data Engineering Bulletin*, 24(4):1–9, 2001.

[125] S. K. Sinha, K. kumar Pandey, and D. Kushwah. A quick survey on latest content based image retrieval systems (cbirs). *International Journal of Advanced Research in Computer Science and Electronics Engineering (IJARCSEE)*, 2(9):pp–642, 2013.

[126] A. Smeulders and M. Kersten. Crossing the divide between computer vision and data bases in search of image databases. In *Proceeedings of the 4th Working Conference on Visual Database Systems*, pages 223–239, L'Aquila, Italy, 1998. Chapman & Hall.

[127] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000.

[128] F. Stegmaier, M. Döller, H. Kosch, A. Hutter, and T. Riegel. Air: Architecture for interoperable retrieval on distributed and heterogeneous multimedia repositories. In *Analysis, Retrieval and Delivery of Multimedia Content*, pages 149–163. Springer, 2013.

[129] F. Stegmaier, M. Döller, H. Kosch, A. Hutterer, and T. Riegel. AIR: Architecture for interoperable Retrieval on distributed and heterogeneous multimedia repositories. In *Proceedings of the 11th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMS)*, pages 1–4, Desenzano del Garda, Italy, 2010. IEEE.

[130] G. Sun and J. Li. Results processing in a heterogeneous word. In *ITCC*, pages 212–217. IEEE Computer Society, 2002.

[131] M. Swain. Searching for multimedia on the World Wide Web. In *Multimedia Computing and Systems, 1999. IEEE International Conference on*, volume 1, pages 32 –37 vol.1, July 1999.

[132] M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[133] C. Tappert, C. Suen, and T. Wakahara. The state of the art in online handwriting recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(8):787–808, 1990.

[134] The Moving Picture Experts Group (MPEG). MPEG-7. ISO/IEC Standard ISO/IEC 15938, ISO, Geneva (CH), 2002.

[135] M. Turk and A. Pentland. Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR '91., IEEE Computer Society Conference on*, pages 586–591, Jun 1991.

[136] G. Valentini, M. Muselli, and F. Ruffino. Cancer recognition with bagged ensembles of support vector machines. *Neurocomputing*, 56:461–466, 2004.

[137] C. J. van Rijsbergen. Readings in information retrieval. chapter A Non-classical Logic for Information Retrieval, pages 268–272. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.

[138] M. Vatsa, R. Singh, and A. Noore. Reducing the false rejection rate of iris recognition using textural and topological features. *International Journal of Signal Processing*, 2(2), 2006.

[139] R. C. Veltkamp and M. Tanase. A survey of content-based image retrieval systems.

[140] R. C. Veltkamp and M. Tanase. Content-based image retrieval systems: A survey. Technical report, 2000.

[141] C. Vilsmaier, D. Coquil, F. Stegmaier, M. Döller, L. Brunie, and H. Kosch. Query Result Aggregation in Distributed Multimedia Databases. In G. Tsihrintzis, E. Damiani, M. Virvou, R. Howlett, and L. Jain, editors, *Intelligent Interactive Multimedia Systems and Services*, pages 299–308. Springer, 2010.

[142] C. Vilsmaier, J. Jurgovsky, M. Döller, H. Kosch, and L. Brunie. Query forwarding in a semantic CBIR network. 2012.

[143] C. Vilsmaier, R. Karp, M. Döller, H. Kosch, and L. Brunie. Towards Automatic Detection of CBIRs Configuration. In *Proceeding MMM'12 Proceedings of the 18th international conference on Advances in Multimedia Modeling*, pages 334–345, Klagenfurt, Austria, 2012.

[144] Viola and Jones. Rapid object detection using boosted cascade of simple features. In *Proceedings of the Computer Society Conference on Computer Vision and Pattern Recognition*, pages 511–518, Kauai, Hawai'i, United States of America, 2001. IEEE.

[145] L. Von Ahn and L. Dabbish. Labeling images with a computer game. *Proceedings of the 2004 conference on Human factors in computing systems CHI 04*, 6(1):319–326, 2004.

[146] E. Voorhees, N. K. Gupta, and B. Johnson-Laird. The collection fusion problem. *NIST SPECIAL PUBLICATION SP*, pages 95–95, 1995.

[147] E. Voutsakis, E. Petrakis, and E. Milios. {IntelliSearch}: {I}ntelligent Search for Images and Text on the {W}eb. In A. Campilho and M. Kamel, editors, *Image Analysis and Recognition*, volume 4141 of *Lecture Notes in Computer Science*, pages 697–708. Springer Berlin / Heidelberg, 2006.

[148] C. Wang, L. Zhang, and H.-J. Zhang. Learning to reduce the semantic gap in web image retrieval and annotation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '08, pages 355–362, New York, NY, USA, 2008. ACM.

[149] Y. Wang and Q. Ruan. Kernel fisher discriminant analysis for palmprint recognition. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 4, pages 457–460. IEEE, 2006.

[150] I. Watson. *The Universal Machine: From the Dawn of Computing to Digital Consciousness*. Springer, 2012.

[151] S. K. M. Wong and Y. Yao. A generalized binary probabilistic independence model. *Journal of the American Society for Information Science*, 41(5):324–329, 1990.

[152] S. K. M. Wong, W. Ziarko, and P. C. N. Wong. Generalized vector spaces model in information retrieval. In *Proceedings of the 8th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '85, pages 18–25, New York, NY, USA, 1985. ACM.

[153] S. Wu and F. Crestani. Data fusion with estimated weights. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, CIKM '02, pages 648–651, New York, NY, USA, 2002. ACM.

[154] S. Wu and F. Crestani. Shadow Document Methods of Results Merging. *Information Sciences*, pages 1067–1072, 2004.

[155] Z. Wu, Q. Ke, J. Sun, and H.-Y. Shum. Scalable face image retrieval with identity-based quantization and multireference reranking. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(10):1991–2001, Oct. 2011.

[156] J. Yang, A. F. Frangi, J.-y. Yang, D. Zhang, and Z. Jin. Kpca plus lda: a complete kernel fisher discriminant framework for feature extraction and recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(2):230–244, 2005.

[157] X.-S. Yang and S. Deb. Engineering Optimisation by Cuckoo Search. *International Journal of Mathematical Modelling and Numerical Optimisation (IJMMNO)*, 1(4):330–343, 2010.

[158] J. Yoon. A network-aware semantics-sensitive image retrieval system. 2003.

[159] R. Zhao and W. I. Grosky. Bridging the semantic gap in image retrieval. *Distributed multimedia databases: Techniques and applications*, pages 14–36, 2002.

A. Introduction to MPEG-7 Image Features

- **Color Histogram & Scalable Color Descriptor** - Color distribution similarity has been one of the first choices of application, as it was the most obvious method of how humans distinguish images, as well as being the easiest one to implement. For this Low Level Feature (LLF), the number of colors was divided into even spaced regions, from now on called buckets. An additional normalizing of the histogram at that moment leads to a scale invariance of the algorithm. There are several different implementations available for this LLF using different histogram types and color spaces.

  Independently from the implementation, the empirical probability for one data point falling into one of these buckets is determined by counting. Commonly, but not necessarily, a normalization of these buckets takes place. To measure the distance between two of the determined vectors, various metrics such as the Manhattan Distance, Euclidian Distance, Jefrey Divergence, and more, are implemented and analyzed. The problem of restricting color similarity to color exclusively results in a further issue.

  This problem is visualized in figure A.1. Paining 1 and Paining 2 are of the same size and are each filled with the same amount of pixels of the same colors. Looking at those two paintings from a human point of view they obviously differ. Taking the point of view of an algorithm that applies the Color Histogram, these images are identical. Under these special circumstances, this perspective can straightforwardly be simulated by looking at the Color Histograms of both paintings in the middle of figure A.1. Both of them are obviously identical, independent of the applied comparison algorithm. Thus this example shows that the Color Histogram feature analyzes the existing amount of pixels for every color without taking into account their positioning.

  This is a very basic and widely used feature that represents the color distribution of the input image [39]. The MPEG-7 generic Scalable Color Descriptor is such a color histogram, encoded by a Haar transformation. It uses the HSV colors space, uniformly quantized to 255 bins. To arrive at a compact representation the histogram bin values are non-uniformly quantized in a range, from 16 bits/histogram for a rough representation of color distribution up to 1000 bits/histogram for high-quality applications. Matching between Scalable Color Discriminant realizations can be performed by matching Haar coefficients or histogram bin values with the employment of an L1 norm.

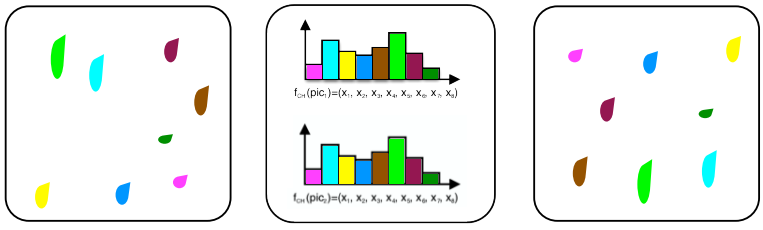


Fig. A.1.: Paining 1 - Color Histograms of both Painings - Paining 2

- **Color Layout Descriptor** Descriptor - The MPEG-7 Color Layout Descriptor is designed to capture the spatial distribution of color in an image, in addition to the color distribution. The feature extraction process consists of two parts; grid based representative color selection and discrete cosine transform with quantization. As already discussed in the precedent section, the description color feature is the most basic quality of the visual contents, therefore it is possible to use colors to describe and represent an image.

  The MPEG-7 standard has tested the most efficient procedure to describe the color and has selected those that have provided more satisfactory results. It proposes different methods to obtain these descriptors, and one tool aiming to describe the color, that permits the description of the color relation between sequences or group of images, is the Color Layout Descriptor. It captures the spatial layout of the representative colors on a grid superimposed on a region or image and is a very compact and resolution-invariant representation of color for high-speed image retrieval. It has been designed to efficiently represent the spatial distribution of colors.

  This feature can be used for a wide variety of similarity-based retrieval methods, content filtering, and visualizations. The extraction process of this color descriptor consists of four stages: Image partitioning, Representative color selection, DCT transformation, and Zigzag scanning. The standard MPEG-7 recommends use of the YCbCr color space for the CLD. In the image partitioning stage, the input picture (on an RGB color space) is divided into 64 blocks to guarantee the invariance to resolution or scale. The inputs and outputs of this step are shown in the first step of figure A.2. After the image partitioning stage, a single representative color is selected from each block.

  Any method to select the representative color can be applied, but the standard recommends the use of the average of the pixel colors in a block as the corresponding representative color, since it is simpler and the description accuracy is sufficient in general. This selection results in a tiny image icon of size 8x8. The inputs and outputs of this step are shown in the second step of figure A.2. In the image of the figure, the size of the original image has been maintained only in order to facilitate its representation. Once the tiny image icon is obtained, the color space conversion between RGB and YCbCr is applied. In the third step, the luminance (Y) and the blue and red chrominance (Cb and Cr) are transformed through the 8x8 discrete cosine function, so three sets of 64 discrete cosine function coefficients are obtained. A zigzag scanning is performed with these three sets of 64 discrete cosine function coefficients, following the schema presented in the last step of figure A.2. The purpose of the zigzag scan is to group the low frequency coefficients of the 8x8 matrix. A subsequent matching process which is performed by a CBIRs helps to evaluate if two elements are equal by comparing both elements and calculating the distance between them.

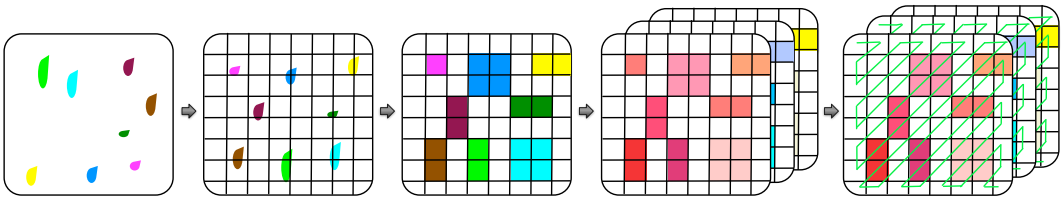


Fig. A.2.: Processing for the Color Layout Descriptor

- **Edge Histogram Descriptor** - The Edge Histogram Descriptor basically represents the distribution of 5 types of edges in each local area (called a sub-image). As shown in

the first step of figure A.3, the sub-image is defined by dividing the image space into 4x4 non-overlapping blocks. Thus, the image partition always yields 16 equal-sized sub-images, regardless of the size of the original image. To characterize the sub-image, a histogram of edge distribution is generated for each instance. Edges in the sub-images are categorized into 5 types: vertical, horizontal, 45-degree diagonal, 135-degree diagonal, and non-directional edges. Thus, the histogram for each sub-image represents the relative frequency of occurrence for the 5 types of edges in the corresponding sub-image. As a result, as shown as the output of step two in figure A.3, each local histogram contains 5 bins. Each bin corresponds to one of the 5 edge types. Since there are 16 sub-images in the image, a total of 516=80 histogram bins is required (See Fig. 4).

Each of the 80-histogram bins has its own semantics in terms of location and edge type. For example, the bin for the horizontal-type edge in the sub-image located at (0,0) in figure A.3 carries the information of the relative population of the horizontal edges in the top-left local region of the image. The semantics of the 1-D histogram bins form the normative part of the MPEG-7 standard descriptor. Specifically, starting from the sub-image at (0,0) and ending at (3,3), 16 sub-images are visited in the raster scan order and corresponding local histogram bins are arranged accordingly. Within each sub-image, the edge types are arranged in the following order: vertical, horizontal, 45-degree diagonal, 135-degree diagonal, and non-directional. Of course, each histogram bin value should be normalized and quantized.

For normalization, the number of edge occurrences for each bin is divided by the total number of image-blocks in the sub-image. The image-block is a basic unit for extracting the information regarding the edge. That is, for each image-block, it is determined whether there is at least an edge and which edge is predominant. When an edge exists, the predominant edge type among the 5 edge categories is also determined. Following this, the histogram value of the corresponding edge bin increases by one. Otherwise, for the monotone region in the image, the image-block contains normalized 80 bin values, that are non-linearly quantized and fixed-length coded with 3bits/bin.

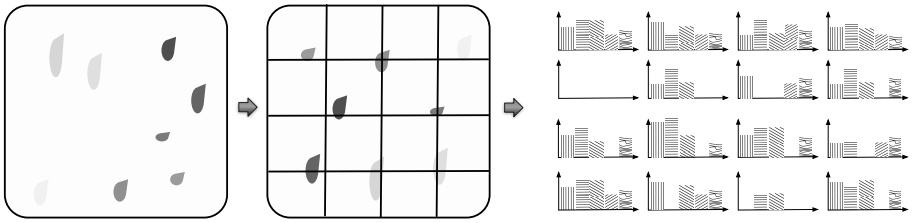


Fig. A.3.: Processing for the Edge Histogram Descriptor

B. Graphical User Interface

The implementation of this thesis approach bases on an early *A.I.R.* [129] implementation. The *QUASI:A*-called implementation was extended in order to be able to support the thesis approach and obtain a basis for the necessary evaluations. Figure B.1 visualizes hereby the user interface. Figure B.2 shows the query by media interface.

QUASI:A was extended so that the training process (discussed in chapter 5) can be launched by clicking on the training symbol on the left side of figure B.1. Similar adaptions were made to visualize the semantic CBIRn (see B.4) and to perform the tests. Minor adaptions in the form of icons were made to the result interface (see figure B.3) in order to make the distinction of the images' sources more intuitive.

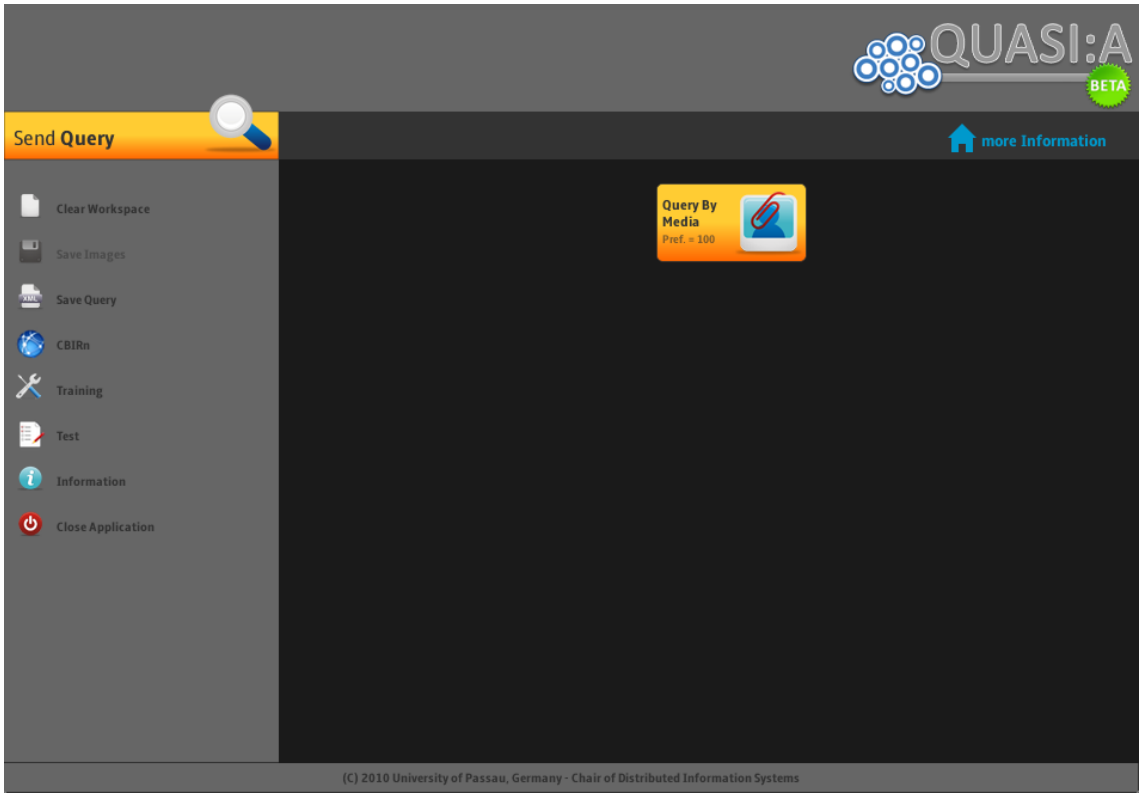


Fig. B.1.: QUASI:A BETA

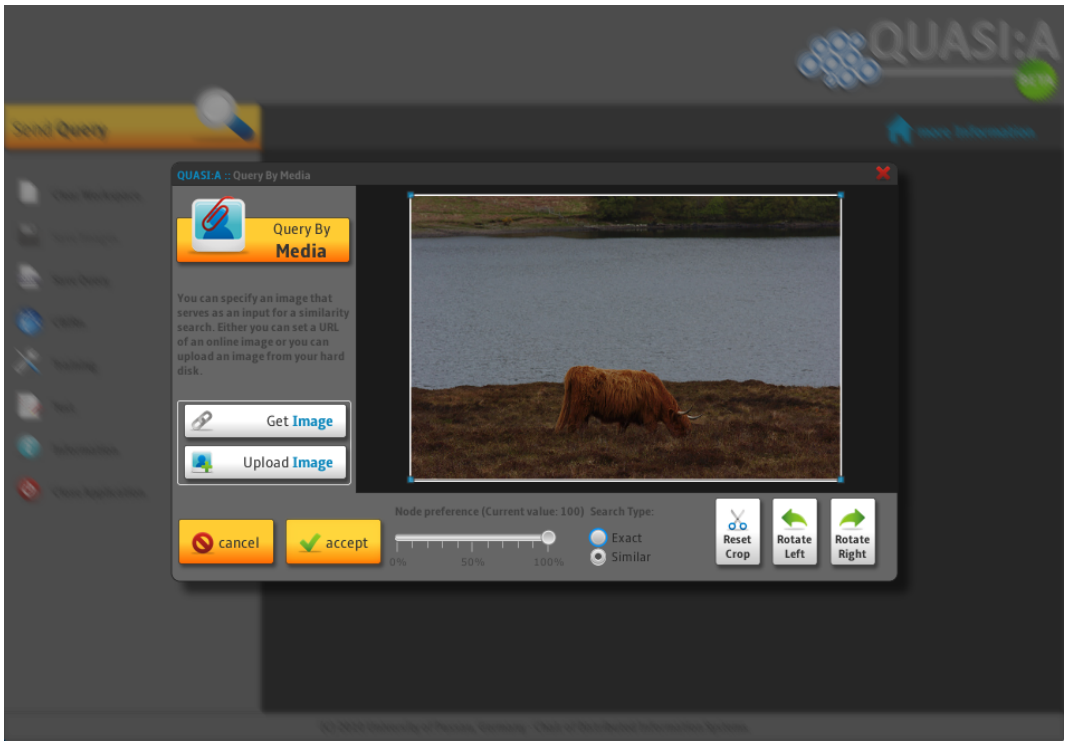


Fig. B.2.: QUASI:A BETA - Interface Query by Media

Fig. B.3.: QUASI:A BETA - Result Interface

Fig. B.4.: QUASI:A BETA - Interface CBIRn