Grant Agreement Number: 257528

KHRESMOI

www.khresmoi.eu

D2.6: Report on and Prototype of final Image Retrieval and Analysis Framework

Deliverable number	*D2.6*
Dissemination level	*Public*
Delivery date	*28.02.2014*
Status	*Final*
Authors	*Dimitrios Markonis, René Donner, Ljiljana Dolamic, Roger Schaer, Georg Langs, Célia Boyer, Henning Müller*

This project is supported by the European Commission under the Information and Communication Technologies (ICT) Theme of the 7th Framework Programme for Research and Technological Development.

Executive Summary

In this deliverable we describe the KHRESMOI prototype framework for image retrieval and analysis. The framework serves as the basis for both 2D and 3D image retrieval. Indexed data ranges from images on web-sites to figures in literature, and finally to imaging data occurring in a clinical environment. After giving an overview of the retrieval systems developed on the course of the project we detail the three primary phases of the framework that are shared across imaging modalities and use–case scenarios.

First, features are extracted from imaging data. They are the basis for establishing similarity across images, or even regions within different images. For 2D images we use local features and global descriptors that capture the appearance of images, and are sufficiently specific to not only find similar images but also to perform certain classification tasks, such as to constrain searches within certain types or modalities. For clinical imaging data, we follow the paradigm that the primary retrieval matches regions with imaging data, since typically only a small portion of a volume is affected by disease, and correspondingly only regions are matched across cases.

Secondly, based on these features we build indices that allow for fast search within the imaging data bases. The ParaDISE indexer indexes 2D images, and allows for similarity search across images and figures in medical literature. For 3D clinical imaging data, the framework builds multiple indices that are valid within certain anatomical regions.

Lastly, the actual retrieval phase relies on indices to generate a ranked list of retrieval results that match a query cases. In case of 2D images, this is typically an image, while in 3D clinical imaging data, the query consists of a region of interest that contains a pathological observation for which the user aims to find similar cases.

The retrieval outlines the involved methods, and gives an overview of the resulting framework. The details of the methods are explained in published work and previous KHRESMOI deliverables. Here we aim at providing the reader with a compact overview of the framework to allow judging and understanding of individual components in the context of the overall system.

Table of Contents

List of Figures

Notation

$\mathbf{I}_i$	Image or volume with index i. If it is 2D or 3D data will be clear from the context.
$\mathbf{I}_i \in \mathbb{R}^2$	2D data such as images.
$\mathbf{I}_i \in \mathbb{R}^3$	3D data such as volumes.
$\mathbf{I}_i(x)$	Value of image of volume at position x
$\mathbf{f}(x)$	Feature (vector) extracted at position x
$\mathbf{d}(\mathbf{I})$	Image descriptor (vector) for an entire image/volume

Abbreviations

ANN	Approximate Nearest Neighbour
API	Application Programming Interface
BoC	Bags of Colors
BoVW	Bags of Visual Words
CBIR	Content–based image retrieval
CEDD	Color and Edge Directivity Descriptor
CouchDB	Cluster of unreliable commodity hardware DataBase
CSV	Comma–Separated Values
CT	Computed Tomography
DICOM	Digital Imaging and Communications in Medicine
DoG	Difference of Gaussians
E2LSH	Euclidean Locally Sensitive Hashing
EMEA	European Medicines Agency
Europarl	Europarl: A Parallel Corpus for Statistical Machine Translation
ezDL	easy Digital Libraries
FCTH	Fuzzy Color and Texture Histogram
GLCM	Gray Level Coocurrence Matrix
GUI	Graphical User Interface
HES–SO	University of Applied sciences, Western Switzerland
HoG	Histogram of Gradients
HTTP	Hypertext Transfer Protocol
ImageCLEF	Image Retrieval in the Cross Language Evaluation Forum
JSON	JavaScript Object Notation
K4E	Khresmoi for Everyone
LBP	Local Binary Patterns
MAP	Mean Average Precision
MeSH	Medical Subject Headings
MRI	Magnetic Resonance (Imaging)
MUW	Medical University of Vienna
MySQL	My Structured Query Language
PACS	Picture archiving and communication system
ParaDISE	Parallel Distributed Image Search Engine
REST	Representational state transfer
ROI	Region of interest
SCA	Service Component Architecture
SIFT	Scale-Invariant Feature Transform
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TF–IDF	Term Frequency – Inverse Document Frequency
URL	Uniform Resource Locator

1 Introduction

The KHRESMOI project aims at providing effective retrieval of medical information to the general public, medical professionals, and clinical radiologists. Each of these user groups has different information needs. While the general public typically searches web pages, medical professionals need efficient access to relevant medical literature, and clinical radiologists need to relate different cases based on visual and textual information.

In this deliverable we describe the framework that allows for the retrieval based on image information. It is part of the various prototypes in KHRESMOI that serve different user groups and usage scenarios. While the context changes, the underlying structure of the framework in these scenarios is similar. Image information is captured by means of local features and image descriptors that capture relevant appearance information. This quantitative information is indexed to allow for fast retrieval of visually similar cases. Lastly the actual retrieval is performed either on the level of entire images, or based on local regions of interest that contain query patterns encountered during clinical routine.

There are several challenges when performing retrieval in medical imaging data. Typically, the similarity that is considered relevant during retrieval is of a high–level type, such as 'similar disease', or even 'disease relevant for differential diagnosis'. The task of the image retrieval and analysis framework is to identify those features and provide manners of indexing data that allow for serving this information need based on imaging information.

In the following we describe the feature extraction, indexing and retrieval steps in the 2D, and 3D contexts. We outline the corresponding information need and how it is met by means of learning algorithms, feature extractors, localizers and the structuring of indices following modalities or anatomical structures.

2 Full system overview

In this section we provide system overviews and describe the usage of 2D image retrieval, 3D clinical imaging data retrieval and text based image retrieval.

2.1 2D article image retrieval

In this section the full system for retrieving 2D images contained in biomedical articles is presented. First, a walkthrough of the system usage is presented describing the interface tools (Section 2.1.1). Then the engine responsible for indexing and retrieving the images by their visual content is described (Section 2.1.2). This includes the description of the backend and the frontend design concepts together with the general architecture. Finally the integration of this system with the KHRESMOI user interface is described (Section 2.1.3).

2.1.1 Usage of the KHRESMOI articles and 2D Image search prototype

The user interface of article and 2D image retrieval is based on ezDL [5]. A more detailed description can be found in [6]. A screenshot of the main 2D image search interface is given in Figure 1. The basic elements are the Query View, the Results View and the Detail View. The

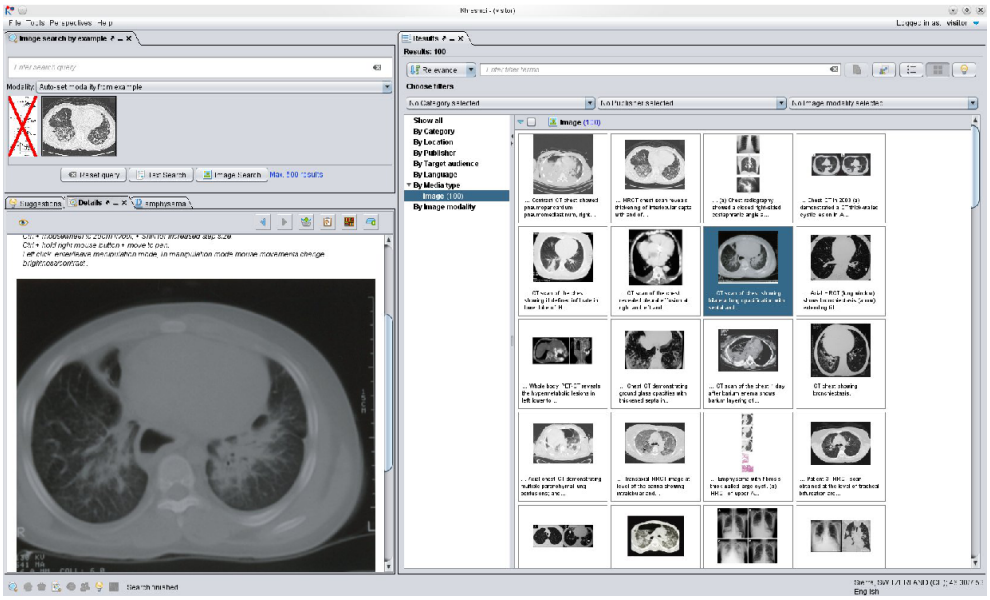

Figure 1: The 2D image retrieval interface.

user can use the Query view to add text or positive and negative image examples and initiate a search. Restricting the search with a specific image modality (or a group of modalities) is also supported.

Once a search has been initiated, the results are presented in the Result View in either ranked list or grid format. Results found in this list can be added in the query to initiate a new search iteration through relevance feedback (RF). Filtering the results by modalities and media type is also supported.

By selecting a result, its associated information appears in the Details view. For articles this means the full title, the abstract and the images included. Search for similar images can be initiated from this view. For image results this means the full size image, the caption and link to the corresponding article. Basic image manipulation is available to allow for better image content inspection.

More tools, such as the Personal library and collaborative tools are available and described in more details in [6]. The 2D image retrieval prototype is a part of KHRESMOI professional prototype [1]. It is also used in the Khresmoi Radiology prototype [2] for extending the search within external resources.

2.1.2 The Parallel Distributed Image Search Engine(ParaDISE)

The development of the 2D image retrieval system of KHRESMOI including building a Content–Based Image Retrieval (CBIR) engine. The outcome was named Parallel Distributed Image Search Engine (ParaDISE). Also taking into consideration the use–case requirements [41], the design of ParaDISE was based on the following concepts: flexibility, expandability and scala-

[1] http://professional.khresmoi.eu/index.html
[2] http://radiology.khresmoi.eu/index.html

bility. The development of the different parts of ParaDISE with regard to these concepts will be discussed after the overview of the architecture of the ParaDISE backend and frontend.

- **Backend**
  The ParaDISE backend follows an object–oriented architecture and consists of basic elements, called Components. Each Component is associated with a Manager object. The Manager is responsible for selecting one out of the supported instances of the Component. The behavior of the selected instance is controlled by a Parameters object that contains the tunable values of the method implemented in the instance. The Components are the Extractor, the Descriptor, the Storer and the Fusor:

  - **The Extractor** undertakes the extraction of local descriptors. More information on the local feature extractors supported in the Extractor can be found in Section 3.1.1.
  - **The Descriptor** is responsible for the mid–level features aggregating the local descriptors extracted by the Extractor. It also contains global descriptors, for which no local features extraction is needed. More information on the global descriptors and mid–level features supported in the Descriptor exists in Section 3.1.2.
  - **The Storer** is used to store the image representation vectors produced by the Descriptor during the indexing process. It is also responsible for accessing the index during online search. The storing methods supported in the Storer are described in Section 4.1.1.
  - **The Fusor** undertakes the fusion of retrieved results lists. These can be either lists retrieved by multiple image queries or results retrieved using different features, indices and even other image retrieval systems. The fusion rules supported in the Fusor are described in Section 5.1.1.

  The Components are combined to perform the two main operations for CBIR, offline indexing of the database images and online search using a set of image examples (Figure 2). These two processes are implemented in complex ParaDISE elements, called CompositeComponents: the Indexer and the Seeker. Again, a Manager is used to select an available Indexer or Seeker and a CompositeParameters object is used to control its behavior. The indexing and search processes are described in more detail in Sections 4.1.1 and 5.1.1 respectively.

  The Components count was kept as low as possible to cover most CBIR approach pipelines without making the system architecture too complex. However, the addition of new Components (e.g. a Preprocessor or a Classifier) is relatively simple due to the component–based architecture. JAVA was chosen as the main programming language for the implementation of the ParaDISE backend.

- **Frontend**
  The ParaDISE frontend, namely the service layer, consists of multiple web services which use a REpresentational State Transfer (REST)–style architecture. Standard Hyper Text Transfer Protocol (HTTP) GET and POST requests are used to communicate with the web services. However, an offline version of ParaDISE frontend exists in the form of a JAVA library. This facilitates easy installation and usage of the engine for single–server applications, personal databases and small–scale research experiments.

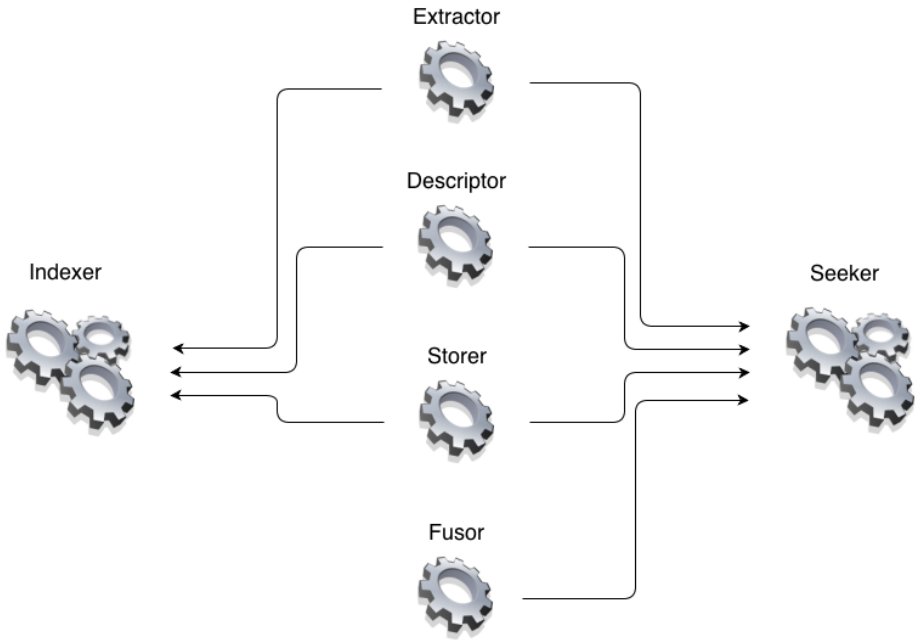

Figure 2: An overview of the ParaDISE backend. The four basic elements are combined to perform the indexing and search processes.

As mentioned before, the design concepts of ParaDISE were flexibility, expandability and scalability.

Flexibility for such a system is crucial, in order to be usable for both research purposes and as an application. Evaluating image representations is really important in CBIR as different features perform better for different databases, depending on the content and the task. Moreover, state–of–the–art CBIR techniques usually include several steps and require a lot of parameter tuning [62]. The choice of component–based architecture for ParaDISE allows for combining local and mid–level features and the evaluation of single steps in the indexing and retrieval pipeline. The use of editable parameters of the ParaDISE components facilitates tuning parameters and experiment with different configurations of methods. Scientific software packages, such as MatLab can cope with most research tasks. They are, however, rarely used in practical applications due to their lack of efficiency. ParaDISE is programmed in JAVA and uses JSON as a data transfer protocol to enable interoperability and realistic application development. The frontend webservice–based architecture allows for the integration of ParaDISE into larger systems and a flexible hardware topology. The use of REST and HTTP request simplifies interaction between the system and various client applications (Web–based or desktop applications that can be written in any language capable of making HTTP requests).

With CBIR being an active research field, novel techniques emerge achieving faster and more precise performance. Thus, expandability is important to be able to add new components for specific steps or new algorithms for the existing components. The object–oriented and plugin–like architecture of ParaDISE allows for such expansions (e.g. 3D features, a Classifier component etc.). The late fusion techniques of the Fusor component can be used to expand the engine by combining it with other retrieval systems (e.g. text–based retrieval engines, such as

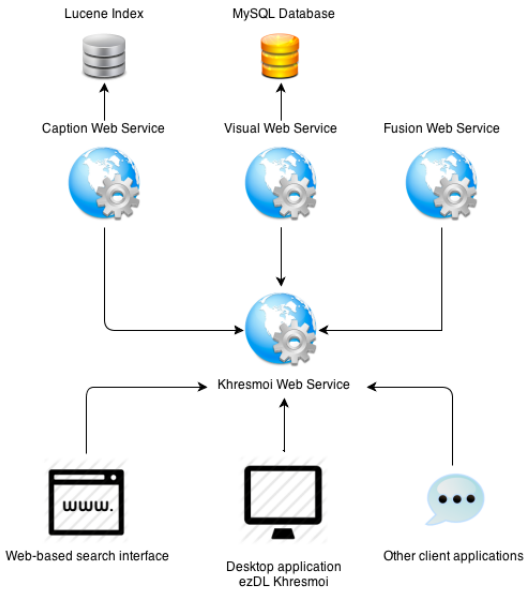

Figure 3: An overview of the ParaDISE interface.

Lucene).

Last but not least, scalability is a critical issue for many real–life applications and an active research field in CBIR [1, 49, 28, 53]. Indexing large image collections and storing the indices can be troublesome and resource–demanding. Updating such indices in regular time intervals should be taken into consideration when designing the indexing pipelines. Moreover, exhaustive search time in large indices is prohibitive in CBIR applications, since CBIR search constitutes of computing distances of image descriptor vectors. In ParaDISE, parallel indexing is supported using the MapReduce framework [14] (see Section 4.1.1). Efficient indexing methods to facilitate fast online search and binary descriptors to reduce memory storage are also supported (Sections 3.1, 4.1.1, and 5.1). The component–based architecture is dealing with scalability by allowing the use of distributed resources and expand when the amount of data and computations grows.

2.1.3 Integration in KHRESMOI system

ParaDISE is exposed to the KHRESMOI user interface using the service layer and through the KHRESMOI SCA integration layer [42]. An overview of the achitecture is shown in Figure 3. The Khresmoi web service combines the caption and visual search services using the Fusion web service. The visual search service is the frontend of ParaDISE, while the caption search service interfaces a text search engine based on Lucene[3]. The Fusion web service is the frontend of ParaDISE Fusor Component. As depicted in Figure 3, it is also used to interface these web services with potential Graphical user Interfaces (GUIs).

[3]http://lucene.apache.org/

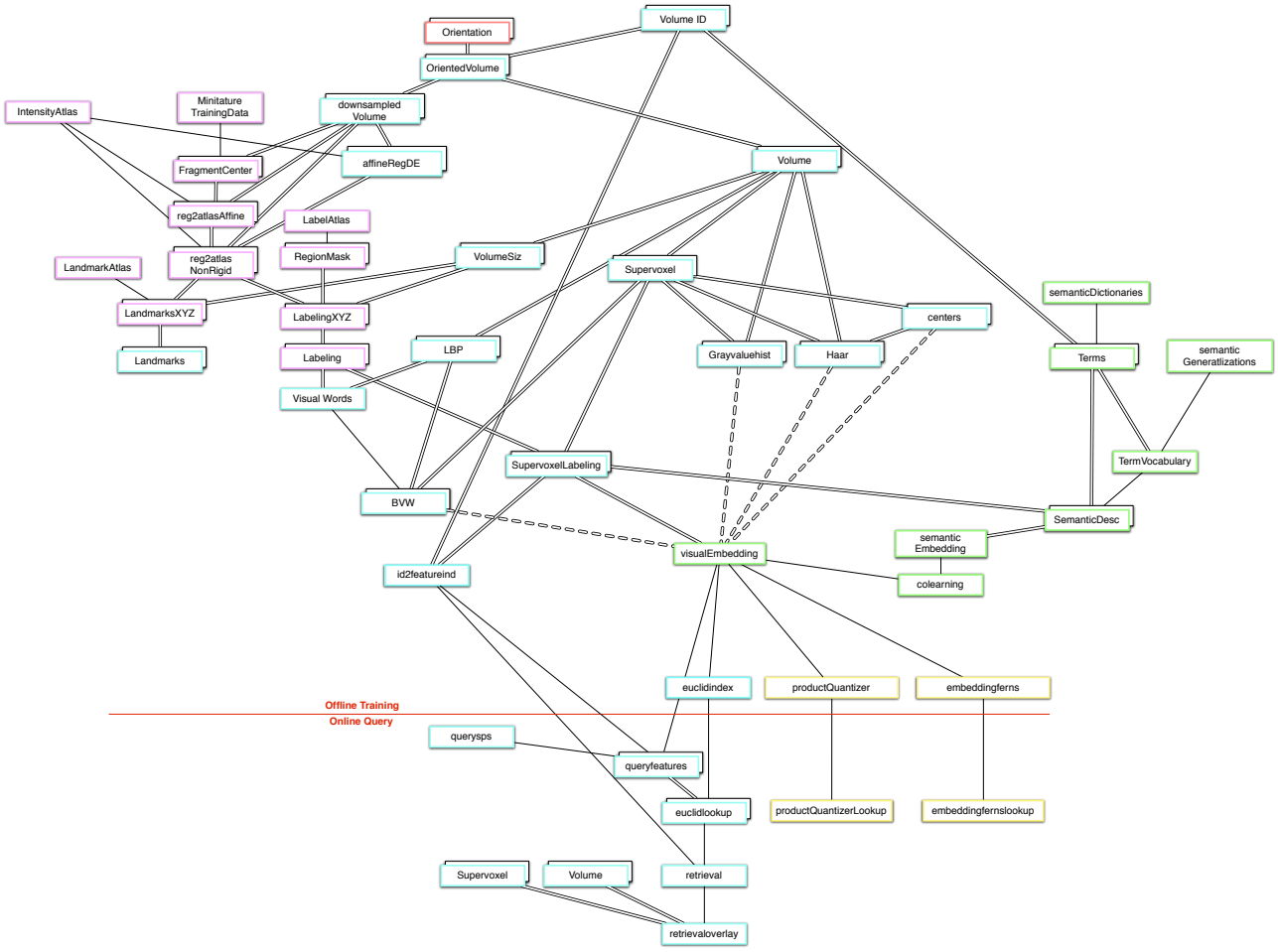

Figure 4: Final MUW 3D retrieval MapReduce graph. Note how the graph describes both the long-running off-line training as well as the interactive, on–line retrieval backend powering the user interface.

2.2 3D clinical image retrieval

The 3D retrieval system developed within this project is a comprehensive system which includes aspects of segmentation, registration, visual feature extraction as well as semantic analysis and indexing to provide a clinically relevant tool to diagnosing radiologists.

The complex interaction between the individual components of the system is captured by the dependency graph in Fig. 4. This dependency graph also drives the actual computations via a map reduce framework.

In the following we will give an overview of the entire system structure as well as the individual modules and refer to subsequent sections for detailed explanations.

2.2.1 3D prototype usage description

The final prototype as depicted in Fig. 5 encompasses all the features planned for. After the ability to choose a case from the indexed cases, complete with 3D preview (not pictured), the

Figure 5: Screenshot of the 3D prototype

user is presented with the pictured query view. It allows, on the left hand side, to select the region of interest, for which similar cases shall be found. On the right hand side the resulting cases are listed, in the form of a 2D preview trough the area of highest similary within each volume, and a textual summary of the semantic terms in the report are listed.

Clicking on one of these results shows the corresponding volume in the center of the screen. The regions (supervoxels) are colorcoded according to their similarity with the region of interest. This rendering can be toggled on and off, of course. At the bottom we see the diagnostic report belonging to this volume, with the anatomical and pathological terms highlighted.

From all the results on the right, by aggregating the semantic anathomy and pathology terms, a so-called consensus term is extracted, shown just above the result list. This consensus is used to kick off the text–only and the 2D + text searches in the background, accessible through the tabs at the top right of the screen.

Requirements Driven by the requirements of the radiological use case the system developed in the work package is responsible for finding visually similar areas in radiological image data, given a query region of interest. This results in two distinct phases of the system where in the first phase the visual information has to be indexed so that in the second phase, i.e. the end–user interaction, this indexed information can be rapidly accessed.

The first phase is responsible for

- importing the data from the hospital PACS system
- labeling individual volumes with the anatomically relevant regions
- partitioning the volumes into coherent regions (super voxels)
- computing visual characteristics for each of these super voxels

- indexing the super voxels according to these visual characteristics
- extracting the semantic information contained in the diagnostic reports for each case

The second phase is responsible for

- finding the anatomic region corresponding to the region of interest user indicated
- computing the visual characteristics of the region of interest
- retrieving the visually similar supervoxels from the index
- aggregating and ranking these retrieval results to yield the list of most relevant volumes
- computing a consensus of the semantic information contained in the report corresponding to these volumes

Computational framework We developed a powerful MapReduce framework to be able to automatically structure the computations necessary for the complex interaction between the required tasks. The interaction between the tasks is defined by their mutual dependencies. The resulting dependency graph capturing the data flow within this project is depicted in Fig. 4. It can be coarsely structured according to the two phases mentioned above. The first phase is the off–line training phase. It takes the volumetric image data and the diagnostic reports as input and yields the index and the semantic information as output. The second online phase is run during user interaction. It takes the index and semantic information as well as the region of interest, defined by the user, as input and yields the list of the most relevant volumes together with the probabilistic overlays and the semantic consensus as output.

The off–line phase can furthermore be subdivided into five sub-graphs: volume pre–processing, localization and registration, visual feature computation, semantic extraction, and indexing.

Volume pre–processing The volume pre–processing is responsible for taking raw image data (DICOMs), aligning them according to their header information and resampling them to a uniform, isotropic resolution. The nodes in the dependency graph belonging to the volume preprocessing can be seen at the top of Fig. 4.

The entire processing of the off–line phase is driven by individual volumes defined by the volume identification (*volume ID*). Given this volume identification the corresponding metadata can be retrieved from the database which also includes the actual filenames of the slices composing the volume. The node *orientation* is responsible for checking the axis alignment of the volume and for providing information about their correction if necessary. The node *orientedvolume* is then responsible for orienting the volume according to this information as defined by the DICOM standard. The final two nodes of the volume pre–processing are *volume* and *downsampled volume*. Both of these are isotropic volumes, i.e. the resolution is the same in all three dimensions. The volume node has a resolution of .7 mm which retains the fine visual characteristics required to compute the visual features. The node *downsampled volume* has a resolution of 2 mm as this is sufficient for the alignment step and furthermore speeds up the computation.

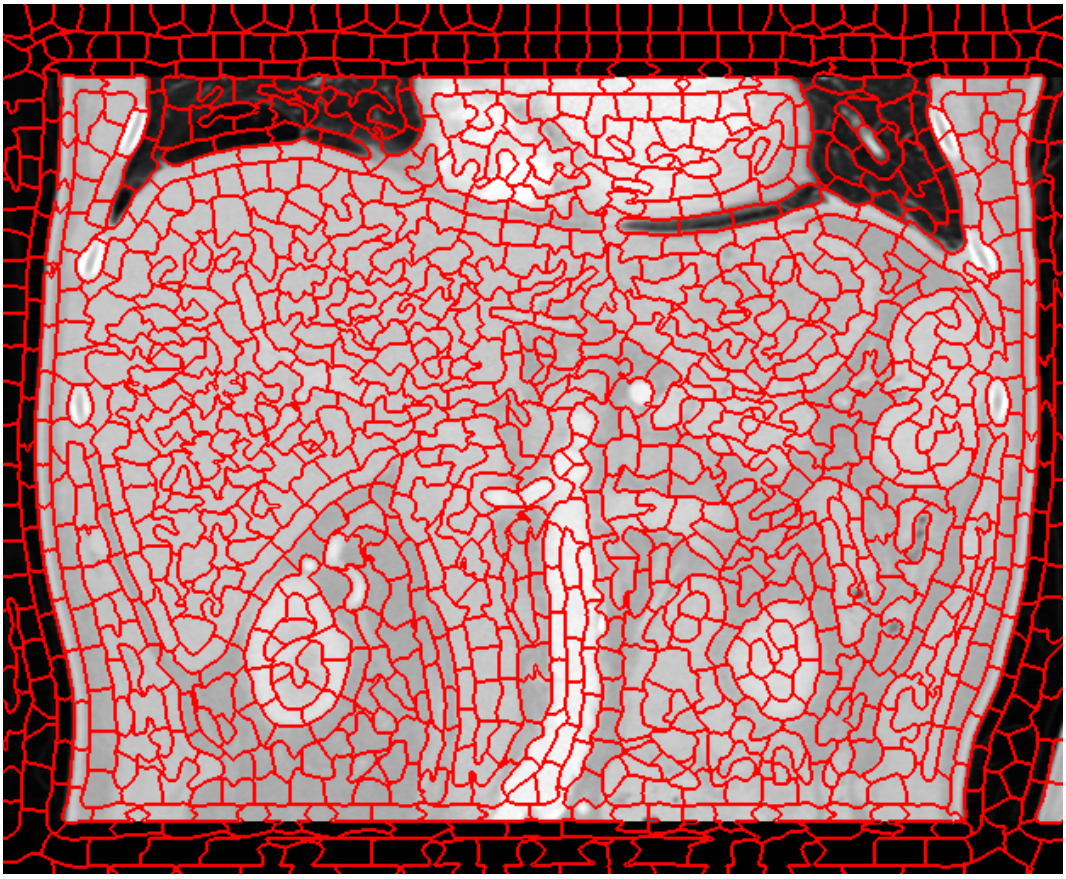

Figure 6: Example for 3D supervoxel computation on a CT volume.

Super voxels The concept of super voxels is central to the entire processing of the 3D part of this work package. The term super pixels or super voxels refers to a concept of partitioning an image into individual segments. This is also called over–segmentation. While several methods exist for computing such a partition the motivation behind these approaches is always similar: They strive to obtain regions which are very homogeneous while still retaining the edge information contained in your original image.

During the course of this project we developed a novel algorithm to compute three-dimensional supervoxels. An example of this can be seen in Fig. 6.

Volume alignment and anatomical labeling The sub–graph on the top left of Fig. 4 shows the nodes responsible for registering the volume to the atlas resulting in the labeling of each super voxel according to anatomical structures.

The intensity atlas is a reference whole body CT on which the anatomical structures have been delineated. The aim of the nodes in this section is to align the volume with this intensity atlas as accurately as possible. To this end we first roughly align the volume with the atlas by finding the volumes center in the atlas, then computing an affine registration and then a nonrigid registration.

Given the deformation information obtained from this registration we can propagate the anatomical labels from the label atlas to the volume resulting in the output of the node *labeling*.

Visual features The visual information for each of the super voxels is computed directly from the high–resolution volume. During the course of this project several different visual features have been studied and can be combined to yield optimal results.

Among the studied features are gray value histograms and Haar–like wavelets. These allow to capture the local and regional gray value distributions and contrast changes in the volume.

Local binary patters are very efficient binary representations of the local great value information and combining them into visual words yields very compact yet descriptive representations of the visual information. Haralick features on the other end first compute a gray label co–occurrence matrix and derive statistical measures from it. All of these features can be combined and form the input for the indexing step.

Semantic extraction The semantic extraction step analyzes the radiological reports and yields their representation in a semantic ontology (Radlex). This allows to use the textual information in statistical and machine learning frameworks.

Based on existing and extended dictionaries the term extraction step maps individual words to individual terms and through generalizations we can obtain a task specific term vocabulary. Combining the raw term information and these vocabularies yields a set of semantic descriptors for each volume.

2.3 Text-based web image retrieval

This section describes the full system of text based image retrieval implemented within the K4E[4]. As described in [50] results of text based image retrieval are related to advanced search mode in the Khresmoi For Everyone (K4E) prototype. With this mode active, with each page of results four images are displayed (Figure 7).

Figure 7: K4E results corresponding to search query “knee”

The image results list is expandable to 12 items (Figure 8) and is pageable independently of the textual results.

Clicking on the image in the result list shows an enlarged image, containing the additional information concerning the image as well as a link to the image original context (Figure 9).

[4] http://everyone.khresmoi.eu/hon-search/

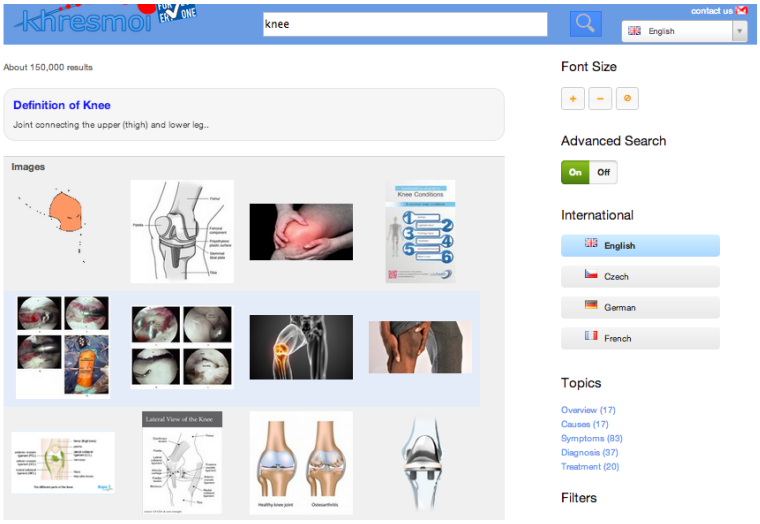

Figure 8: K4E expanded results

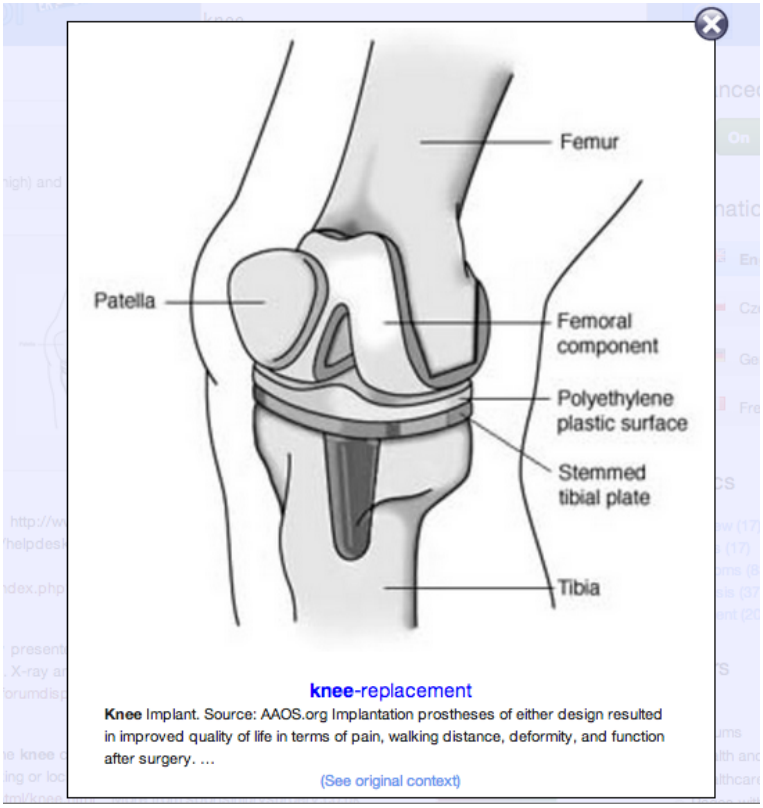

Figure 9: K4E enlarged image result

3 Feature extraction

In this section we provide an overview of the feature extraction algorithms in the framework, completing the work performed in [32]. They range from local features that capture the appearance of small areas in images or volumes to global image descriptors that reflect the overall

content of images for similarity assessment and retrieval. Textual features are also described for text–based retrieval of web images.

3.1 2D article images

In order to represent the 2D images included in the biomedical literature, a large bank of visual feature extractors has been built into the ParaDISE system. This bank was used for finding the features or combination of features that better model the visual information. These features are split into two categories, local features and global descriptors, and are presented in Sections 3.1.1 and 3.1.2

3.1.1 Low–level local features

Local features have been used in CBIR for more than a decade [37], demonstrating state–of–the–art performance in many applications [43, 8]. They represent low–level visual characteristics of regions of the image, such as color, shape and texture. The local feature extraction takes place in the Extractor component of ParaDISE (see Section 2.1). The following local features are supported in the current version of ParaDISE (see also [32]):

- **Scale Invariant Feature Transform (SIFT)** [37]
  SIFT is one of the most commonly used local features, using gradient orientation histograms to model shape. The method of extracting the SIFT features is divided into two main parts: the detection of the interest points in different scales and the description of the neighbourhoods of these points in the appropriate scale. The first part uses the result of a Difference of Gaussians (DoG) applied in scale-space to a series of smoothed and resampled images to detect minima and maxima. The Difference of Gaussians operator can be seen as an approximation of the Laplacian operator:

$$\mathbf{DoG}(x,y;s) = \mathbf{L}(x,y;s+\Delta s) - \mathbf{L}(x,y;s) \approx \frac{\Delta s}{2}\nabla^2 L(x,y;s) \tag{1}$$

  These minima and maxima are candidates as interest points (keypoints). Low contrast candidate points and edge response points along an edge are discarded for robustness to noise. Finally, dominant orientations are assigned to localized keypoints to provide rotation–invariance.

  In the second part, a keypoint descriptor is created by first computing the gradient magnitude and orientation at each sample point in a region around the keypoint location. These are weighted by a Gaussian window to give less emphasis to gradients that are far from the center of the descriptor, as these are most affected by misregistration errors. These samples are then accumulated into 8-binned orientation histograms $\mathbf{H}_i$ with $i = 1 \ldots 16$ summarizing the contents over 4×4 subregions, taken by a 16×16 array around the keypoint. This results in an $4 \times 4 \times 8 = 128 - d$ vector for each keypoint:

$$\mathbf{f}(\mathbf{x}) = [\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_{16}] \tag{2}$$

  The implementation of the SIFT feature in the Fiji image processing package[5] was used.

[5] http://fiji.sc/

- **Speeded Up Robust Feature (SURF)** [4]
  This local feature is similar to SIFT, with the main difference found in the way of detecting interest points. SURF creates a stack without downsampling higher levels in the pyramid, resulting in images of the same resolution. Due to the use of integral images, SURF filters the stack using a box–filter approximation of second-order Gaussian partial derivatives The implementation of the SURF feature in the Fiji image processing package was used.

- **RootSIFT** [3]
  A local feature that is based on SIFT and introduced in [3]. The idea behind this local descriptor is to use the Hellinger kernel to alieviate the effect of large bin values "dominating" the descriptor.

- **Lab local features** [60]
  These local features are used in the approach Bag–of–Colors (BoC) [60]. They produce a 3D feature vector in the CIELab space for the most frequent color of a region.

3.1.2 Mid–level features and global descriptors

While local features perform well in object recognition, image classification and CBIR, they are inefficient for large scale tasks. For this reason statistical image representations have been used, also called mid–level features, with Bag–of–Visual–Words (BoVW) [54] being the most commonly used. Moreover, since there is no one–solution–fits–all in image retrieval applications, other global descriptors have been included in the feature bank. The following mid–level features and global descriptors are supported in the Descriptor component of ParaDISE (see Section 2.1 and [32]):

- **Bag–of–Visual–Words (BoVW)** [54]
  One of the most common approaches for image description using local features in large datasets is the BoVW representation. A training set of images is chosen and local descriptors are extracted from interest points of each image of this set. The descriptors are then clustered using a clustering method into k clusters and the centroids of the clusters are used as visual words. The visual vocabulary $\mathbf{V}$ represents all cluster centers

$$\mathbf{V} = \{v_1, \ldots, v_k\}, \;\; v_i \in \mathbb{R}, \;\; i = 1, \ldots, k \tag{3}$$

  Then, local features are also extracted from all other images in the database and mapped to the cluster centers to create for each image a histogram of visual words. Images are thus indexed as histograms of the visual words (bag–of–visual–words) by assigning the nearest visual word to each feature vector.

  The final image descriptor of image $\mathbf{I}$, called Bag–of–visual–words, is defined as a vector $\mathbf{F}(\mathbf{x}) = \{\bar{v}_1, \ldots, \bar{v}_k\}$ such that, for each local feature vector $\mathbf{f}(\mathbf{x})$ extracted from the image $\mathbf{I}$:

$$\bar{v}_i = \sum_{l=1}^{n_f} \sum_{j=1}^{k} g_j(\mathbf{f}(\mathbf{x}_l)), \;\; \forall i = 1, \ldots, k$$

where n_f is the number of local features extracted from the image and

$$g_j(\mathbf{f}(\mathbf{x})) = \begin{cases} 1 \; if \; d_\varepsilon(\mathbf{f}(\mathbf{x}), v_j) \leq d_\varepsilon(\mathbf{f}(\mathbf{x}), v_l) \; \forall v_l \in \mathbf{V} \\ 0 \; otherwise \end{cases} \tag{4}$$

This mid–level feature, as well as its variants, can be used in combination with any of the low–level local features described in section 3.1.1. The following variants of BoVW are available:

- **Binary BoVW** [54]
  A compact binary representation of BoVW for large scale CBIR. In this descriptor, the elements of the final image descriptor are given by:

$$\bar{v}_j = \begin{cases} 1 \; if \exists i \in [1...n_f] \; such \; that \; d_\varepsilon(\mathbf{f}(\mathbf{x}_i), v_j) \leq d_\varepsilon(\mathbf{f}(\mathbf{x}_i), v_l) \; \forall v_l \in \mathbf{V} \\ 0 \; otherwise \end{cases} \tag{5}$$

  where n_f is the number of local features extracted from the image.
- **Grid BoVW**
  A mid–level representation that splits the image into an $n \times n$ grid of subsections consisting of the concatenated BoVWs of these subregions, to include spatial information.
- **Spatial Pyramid Matching BoVW** [36]
  This mid–level feature includes spatial information of the visual words, by employing pyramid matching. The final image descriptor is a weighted concatenation of BoVWs of the subregions produced by splitting the image into various grid sizes. The weights are inversely proportional to the cell width of the grid.

- **Vector of Locally Agreggated Descriptors (VLAD)** [29]
  VLAD is a recently introduced mid–level feature that has been shown to outperform the state–of–the–art BoVW representation in several computer vision tasks. The image descriptor $\mathbf{F}(\mathbf{x}) = \{\bar{v}_1, \ldots, \bar{v}_k\}$ is a concatenation of vectors $\bar{v}_j$ with elements $\bar{v}_{i,j}$ defined as:

$$\bar{v}_{i,j} = \sum_{j=1}^{d} \mathbf{f}(\mathbf{x}_j) - v_{i,j}, \;\; \forall \mathbf{f}(\mathbf{x}) \; such \; that \; NN(\mathbf{f}(\mathbf{x})) = v_i \tag{6}$$

  where,
  d the dimentionality of the feature space,
  $\mathbf{f}(\mathbf{x})$ a local feature vector extracted from the image $\mathbf{I}$,
  $v_i \in \mathbf{V}$ the visual word,
  $v_{i,j}$ the jth element of v_i,
  $NN(x)$ the nearest visual word to x.

- **GIST** [47]
  The GIST descriptor is a global feature originally created to model the shape of scenes using spatial envelope properties which were estimated using spectral and coarsely localized information. The implementation provided in [47] was used.

- **Riesz miniature**
  This descriptor represents the image as a single Riesz transform vector [15], which is a multidimensional extension of the Hilbert transform. A first step is the downsampling of the image to reduce the dimensionality of the descriptor. It uses a linear combination of N–th order Riesz templates at multiple scales. The weights of the linear combination are derived from one–versus–all support vector machines. Steerability and multiscale properties of Riesz wavelets allow for scale and rotation covariance of the descriptor. Orientations are normalized by locally aligning the Riesz templates, which is carried out analytically. This approach has been used to model texture, shown to outperform state–of–the–art texture attributes in lung classification [15]. An adapted version of the implementation provided in [15] was used.

- **Histograms of Gradients (HoG) miniature** [16]
  This feature mainly represents the image as a single SIFT descriptor of a downsized version of the image. A version of this descriptor on the RGB space is also implemented to combine color information.

- **Gabor Filters**
  Gabor filters have been known to be used in CBIR, modeling texture [55]. Usually, a set of several Gabor filters of different orientations and scales is applied in blocks over the image and histograms of mean filter outputs are used to represent the texture characteristics of the image.

- **Tamura** [56]
  This texture feature was created with regard to human visual perception. Six basic textural properties (i.e., coarseness, contrast, directionality, line–likeness, regularity, and roughness) are used to model texture. For the implementation of this feature, the Lucene Image Retrieval library (LIRe) was used [38].

- **Color and Edge Directivity Descriptor (CEDD)** [10]
  This global descriptor extracts the color information from regions of the image using a set of fuzzy rules and resulting in a HSV color space histogram. It includes texture information using the proposed MPEG-7 [9] Edge Histogram Descriptor rules. Finally, it uses Gustafson Kessel fuzzy classifier [24] to binarize the histogram. For the implementation of this feature, LIRe was used.

- **Fuzzy Color and Texture Histogram (FCTH)** [11]
  Very similar to the CEDD feature, FCTH mainly differs in using Haar Wavelet transform to model texture information. For the implementation of this feature, LIRe was used.

- **Color Layout** [30]
  The extraction for this descriptor consists of four stages: image partitioning, dominant color selection, Discrete Cosine Transform (DCT), and non-linear quantization of the zigzag-scanned DCT coefficients. In the first stage, an input picture is partitioned into 64 blocks. The size of the each block is $W/8 \times H/8$, where W and H denote the width and height of the image, respectively. In the second stage, the dominant color (e.g. the average color) is selected in each block to build an image of size 8×8. In the third stage,

each of the three components (Y, Cb and Cr) is transformed by 8×8 DCT, and three sets of DCT coefficients are obtained. A few low frequency coefficients are then extracted using zigzag scanning and quantized. This descriptor is part of the MPEG-7 standard and the LIRe implementation was used.

- **Fuzzy Color Histogram** [25]
  The Fuzzy Color Histogram is defined as $\mathbf{F}(\mathbf{x})$

$$\mathbf{F}(\mathbf{x}) = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_N] \tag{7}$$

  where $\mathbf{f}_i = \frac{1}{N}\sum_1^N \mu_{ij}, \forall i = 1...N$ where μ_{ij} is the membership of the jth pixel in the ith color bin. The fuzzy C means algorithm is used for the computation of the memberships. For the implementation of this feature, LIRe was used.

- **HSV Color Histogram**
  A simple color Histogram in the HSV color space. For the implementation of this feature, LIRe was used.

- **Singular Value Decomposition (SVD)**
  SVD has been used as a feature in image retrieval and also medical CBIR [13, 22]. Top k singular values are used to create an image descriptor.

3.2 3D clinical images

The information need for retrieving 3D clinical imaging data is typically characterized by diseases that affect local structures in the imaging data as opposed to the entire body. Therefore retrieval is typically based on local regions of interest marked by the user. Furthermore, the appearance across the human body exhibits a very high variability. In order to capture those visual differences that are relevant for finding similar deviations from normal anatomy, we have to learn features. To account for the relatively large appearance differences across anatomical structures, compared to rather subtle changes correlating to disease, we perform learning independently in different anatomical structures. In the following we give an overview of the most important features and image descriptors (see also [35] and [32]).

3.2.1 Gray–level histogram features

As part of the features computed for each supervoxel which can be used in the indexing stage simple gray-value histograms are employed. They are particularly useful in modalities where the image intensity values correspond to measurements on a physically meaningful scale, like the Hounsfield units in computed tomography volumes.

The bins used for the histograms are equidistantly spaced in the value range of 0 to 2000. Each bin covers 100 values, i. e. there are 20 bins. Counting in each bin all voxels of the supervoxel within the corresponding range and normalizing the histogram to add up to one yields the resulting gray–value histogram.

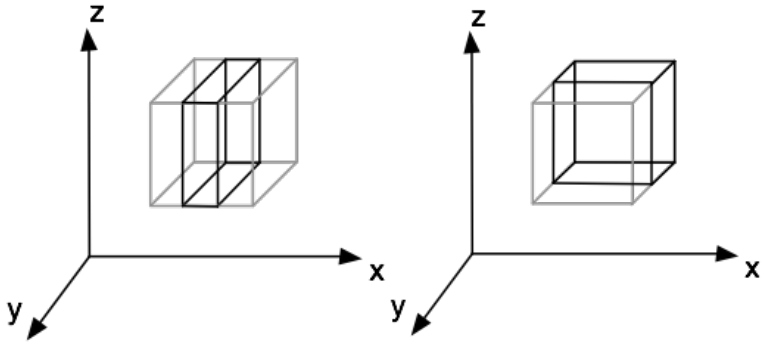

Figure 10: Haal–like wavelets employed in the feature computation stage of the 3D retrieval prototype

3.2.2 Haar–like Wavelets / Long–range context

For describing the local appearance around the model landmarks we employ a set of 3D features computed using a basis of filters similar to Haar wavelets. These features [58] can be computed using integral volumes [31] in a highly efficient manner. An integral volume $\mathbf{J}$ transforms the information content of the original volume $\mathbf{I}$ such that

$$\mathbf{J}(x,y,z) = \sum_{i=1\ldots x} \sum_{j=1\ldots y} \sum_{k=1\ldots z} \mathbf{I}(i,j,k) \tag{8}$$

This allows to compute the sum s within a cuboid given by the coordinates $(x_1,y_1,z_1,x_2,y_2,z_2)$ by

$$\begin{aligned} s \quad = \quad & \mathbf{J}(x_2,y_2,z_2) - \mathbf{J}(x_2,y_1,z_2) - \mathbf{J}(x_1,y_2,z_2) \\ & + \mathbf{J}(x_1,y_1,z_1) - \mathbf{J}(x_2,y_2,z_1) + \mathbf{J}(x_2,y_1,z_1) \\ & + \mathbf{J}(x_1,y_2,z_1) - \mathbf{J}(x_1,y_1,z_1). \end{aligned} \tag{9}$$

Computing a Haar–like feature then consists of forming 2 such sums with opposing signs. To capture the local to medium–range visual information around a supervoxel Haar–like features were computing on 3 different scales. At each scale a block was computed at the center of the supervoxel, forming the *positive* block. Around the supervoxel, 10 other *negative* blocks were computed, positioned according to a fixed Gaussian distribution of with a standard deviation of $\sigma = 10 \cdot scale$ pixels. Fig. 10 shows a schematic view of Haar–like wavelets where the positive and negative blocks are adjacent.

3.2.3 Local binary patterns

As another base feature extractor we use a three-dimensional adaptation of the Local Binary Pattern operator (LBP) described by Ojala et al. [45, 46]. It responds well to microscopic structure [39], is computationally cheap, and invariant to the gray–scale range [27]. The LBP operator computes the local structure at a given pixel i by comparing the values of its eight neighborhood pixels with the value of i. Various extensions of the original operator have been published in recent years. As the local contrast is an important property in the medical domain,

our method is based on LBP/C [46], which combines the base LBP operator $\mathbf{D}_{LBP}$, with a local contrast measure $\mathbf{D}_C$. In some medical imaging modalities, such as CT, intensities of local regions are an important decision instrument for physicians. Since the LBP operator is by definition gray-scale invariant, we furthermore supplement a local average intensity measure $\mathbf{D}_I$ of the 3x3x3 LBP cube to the feature vector if absolute intensity is relevant [7].

3.2.4 Texture bags

Texture bags [7] learn features that represent a given set of training data well by establishing a vocabulary of features. These features are based on e.g., local binary patterns, and are optimal for describing the variability present in a specific region. The features extracted from training data are quantized by performing clustering in the feature space. The resulting k clusters constitute the texture vocabulary or texture words W_k. Each voxel is represented with its closest texture word $\mathbf{W}_k^s$, i.e., with the index of the closest cluster center. Image or volume regions such as super voxels are represented by the histograms of texture words occurring in the region.

3.3 Text

Distributed crawling processes gather web pages which are stored locally and pushed through a textual feature extraction workflow build as a suite of Perl modules. Finally, the information extracted from webpages is stored into a NoSql database (CouchDB [21]) for indexation. Two types of features are extracted in this workflow from each web page.

HTML features URL, title, host etc. HTML feature extraction is out of the scope of this deliverable. This process is described in [26] and [51].

Image features for each image detected within the web page the following information is extracted:

- width
- alt
- title
- precedingText
- followingText
- url
- cleanedFilename
- imageType
- language
- largeFollowingText

- largePrecedingText

CleanedFilename represents the transformed file name e.g. “breastSelfExam” ⇒ “breast self exam”. Preceding text (“precedingText”) and following text (“followingText”) limit extracted text to a window of 200 and 300 bytes relative to the “img” tag, respectively. Large preceding text (“largePrecedingText”) and large following text (“largeFollowingText”) captures the window within 3 HTML tags from the “img” tag. The sizes of the various windows listed above are configurable through the configuration file. Additionally, large texts are included only if the lexical distance between them and title and alt fields is smaller than the configurable threshold. This threshold is situated between 0 (no terms in common) and 1 (all terms in common).

Even though the evaluation of the system described in the deliverable [35] has given promising results, the user feedback has indicated the problem of low relevance of the retrieved images to the query term.

3.3.1 Identified problems

HON has proceeded with two types of evaluation and problem detection.

The HON Image Rater application was created that allows users to rate images returned from K4E and evaluate them as ”relevant” or ”irrelevant.” This web application can be accessed at `http://khresmoi.honservices.org/image-rater/`. Figure 11 shows the images returned by the K4E for the query “asthma”, by selecting the “Yep!” check box a user confirms that the image is relevant to the query.

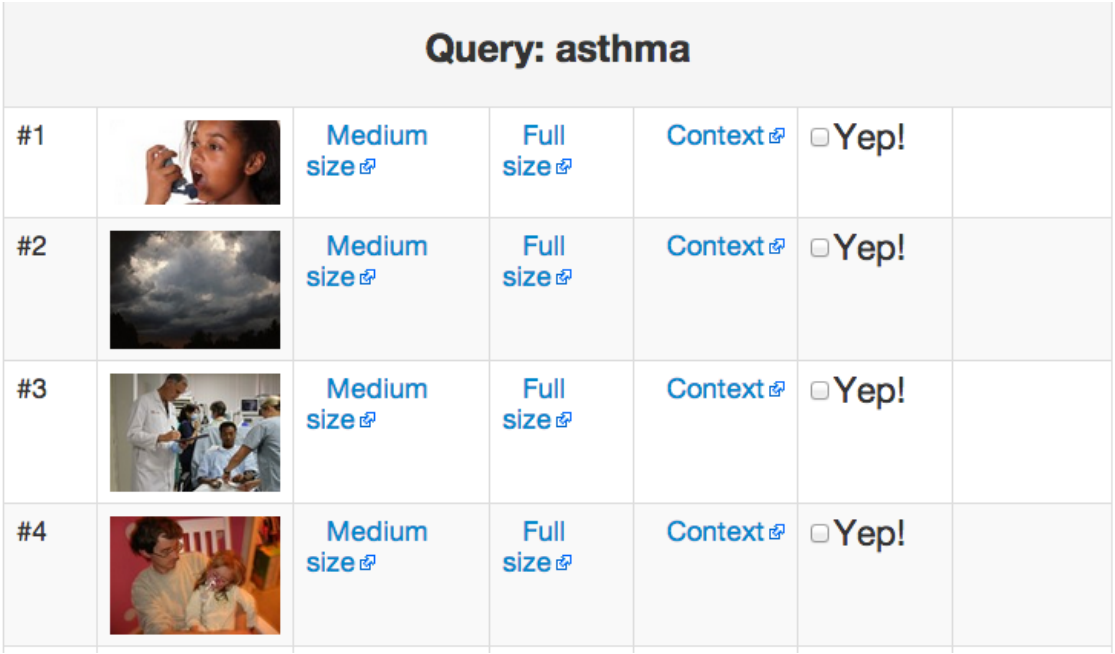

Figure 11: HON Image Rater

Images returned by K4E for a selected set of queries are displayed within the application. These images were evaluated by Khresmoi project partners. The results of this evaluation were

then posted to the above mentioned CouchDB and used by HON to determine problems concerning image retrieval precision. The following issues concerning image extraction were detected:

Slide show images: Large amounts of images having unrelated context, were extracted from slide shows. All images extracted from the same slide show have the same, general caption which most of the time is not related to any of them.

Insufficient image-text similarity: Similarity calculation attributed easily text to the image, especially in the case of very short alt or title length. Additionally, omitting linguistic treatment prior to similarity calculation, has led to a large number of false positives.

Omitted images from good sources: Draconian restriction such as rejections of images smaller than $300 * 300$ or those having a different host than the page, have resulted in large numbers of images being rejected by the system (e.g the image in Figure 12).

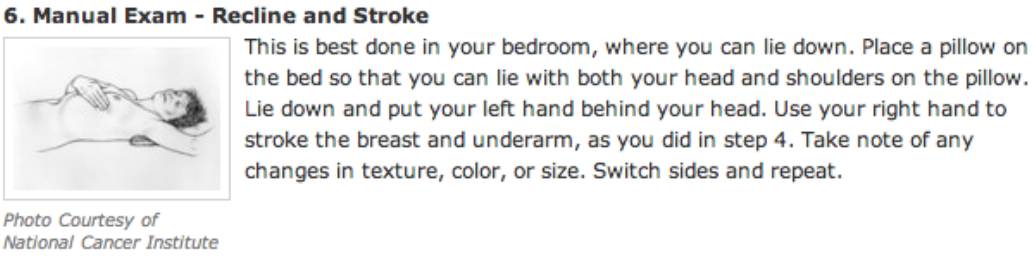

Figure 12: K4E: rejected image

During previous image retrieval evaluations described in [35] HON has not encountered these problems. The Wikipedia pages used in that evaluation do not share some common characteristics with pages crawled by our system. The textual information surrounding images on Wikipedia is related to the image itself, slide shows are not present and the images have the same host as the webpage. Thus, the wikipedia gold reference for web based image retrieval is not adapted and does not reflect the real web, or the real type of images we found on the majority of websites.

Crowd sourcing: In order to determine the exact precision of the current system, the testing collection of 2902 images was created. These images were then sent for crowd source evaluation (CrowdFlower[6]). This collection was used for TREC like evaluation of the retrieval effectiveness of the system. Using various parameters setups, the obtained results indicate that the current system is unable of achieving MAP higher than 0.3388. This precision being judged insufficient has led to changes in the way features are extracted described in the following section. Details of this evaluation will be given within the evaluation deliverable D2.7 (planned for June 2014).

[6] http://crowdflower.com/

3.3.2 Feature extraction redesign

With a goal of dealing with previously detected issues the image features extraction was changed as follows:

- Slide show images omittance – if the images detected within a web page belong to a slide show, they are not taken into account during image extraction.
- Linguistic treatment before lexical distance calculation. Incorporating this step avoids text to be attributed to an image based on stop word (eg. “the”, “or”)
- Removing a certain number of original restrictions, such as size or image source, resulted in extraction of relevant images, such as those found at How To Do a Breast Self Exam (BSE)[7], shown in Figure 12 previously rejected due to a different image host.
- Web page title – linked to the image as an additional field (pageTitle), if the lexical distance to title and alt respect threshold limitations.

4 Indexing

This section presents the pipelines for visual and textual indexing of the accessed image data sets. Components included in these pipelines and efficient index structures used are described.

4.1 2D article images

KHRESMOI uses ParaDISE for indexing the visual content of the 2D images contained in the medical literature, while the caption and article information is indexed using Lucene. The ParaDISE Indexer is presented in Section 4.1.1 and the full KHRESMOI indexing pipeline for 2D images is described in Section 4.1.2.

4.1.1 ParaDISE Indexer

The indexing of the visual content of the image collection is an offline operation. As mentioned in Section 2.1.2, the Indexer Composite Component is responsible for this task in ParaDISE. Apart from serial indexing, parallel indexing is also supported using the MapReduce framework. Below follows the description of the two currently supported methods:

- **Serial Indexer**
  The serial indexing pipeline uses the basic ParaDISE Components (see Figure 13). First, the local features of each image are extracted by the Extractor, if needed. Then, the image descriptor is created by the Descriptor, either using the local feature vectors or a global descriptor. The Storer inserts that image descriptor vector into the index. The direction in the decision nodes is decided by the values of the Indexer Parameters. Four different Storers are currently supported in ParaDISE:

[7] http://breastcancer.about.com/od/risk/tp/bse_illustrated.htm

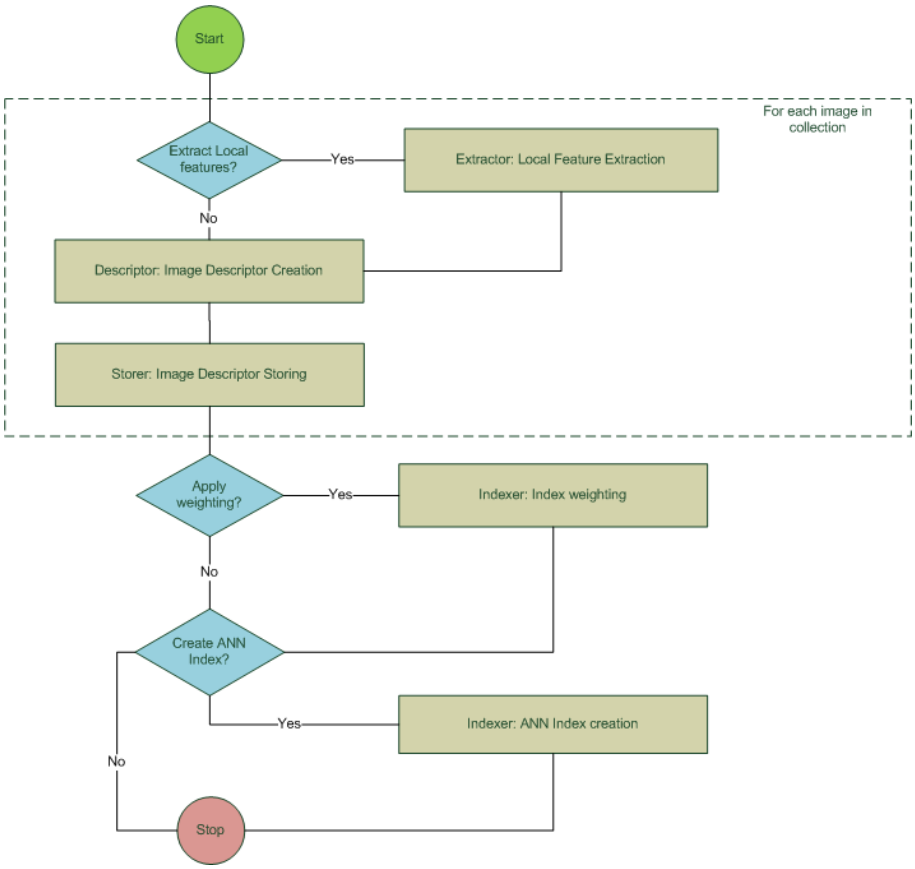

Figure 13: The indexing pipeline of ParaDISE Indexer.

- **CSV Storer**
  This Storer uses a Comma–Separated Values (CSV) file to store the index. It is mostly suitable for research evaluations and small image collections, as it is very inefficient for real applications.
- **SQL Storer**
  The SQL storer stores the image descriptor vectors in a table in a MySQL database. It can be used for application use cases and can handle large datasets as well as image vectors of small dimensionality.
- **CouchDB Storer**
  A noSQL alternative of SQL storer for image vectors of high dimensionality, such as concatenated feature vectors or BoVW models with large vocabularies.
- **Cassandra Storer**
  Cassandra Storer stores the index in a column family of a Cassandra[8] keyspace. Cassandra allows to have a parallel database with millions of columns. This makes it suitable for very large datasets and image vectors of very high dimensionality.

After the index has been created, a weighting can be applied to the index. The following weighting methods are supported:

- **Term Frequency – Inverse Document Frequency (TF–IDF)**
  The TF–IDF weighting is widely used in text–based information retrieval. The rationale behind this weighting is that words that are found often in a document contain more information. At the same time, words that are found often in the document collection are not that informative. The mathematical expression of TF–IDF is the following:
  $$tfidf = \frac{n_{id}}{n_d} \log \frac{N}{n_i} \tag{10}$$
  where n_{id} is the number of occurrences of word i in document d,
  n_d is the total number of words in the document d,
  n_i is the number of occurrences of word i in the whole database
  and N is the number of documents in the whole database.
  It can be used in CBIR in combination with BoVW approaches. For documents containing no visual words ($n_d = 0$) then tfidf $= 0\ \forall i$. For visual words not present in the database ($n_i = 0$), the tfidf of the term i is not relevant and the word can be exluded from the vector comparison. However, for combatibility reasons the maximum tfidf weight is given:
  $$tfidf = \frac{n_{id}}{n_d} \log N$$
- **Frequent Item Selection [59]**
  This weighting uses only the top k tfidf values per image, called frequent items, to provide a compact image representation. The images are then ranked according to the number of common shared frequent items with the query image.

[8] http://cassandra.apache.org/

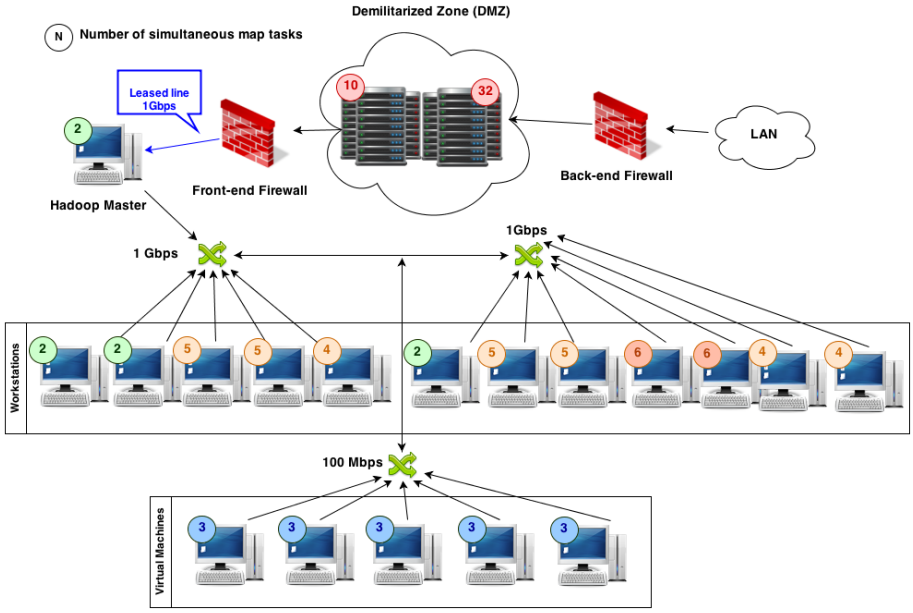

Figure 14: An overview of the HES–SO in–house cluster.

Finally, the indexer can create an Approximate Nearest Neighbour (ANN) index structure to facilitate fast retrieval. Currently, serial and parallel versions of Euclidean Locally Sensitive Hashing (E2LSH) [2] ANN method are supported. This algorithm uses families of hashing functions to partition the index feature space and thus limit the search into the subspace that a query falls into.

- **Hadoop Indexer**
  The Hadoop [61] implementation of MapReduce was used for the parallization of the indexing, since it is an easily parallelizable task. The pipeline is identical to the one shown in Figure 13 except for the fact that the blocks in the frame are executed in parallel. This is achieved by splitting the image collection into small groups of images. Each group is indexed by a different map task.

  Either an in–house or a cloud Hadoop cluster can be used for this indexing method, since the implementation is fully parametrizable. An in–house cluster was created in HES–SO for the needs of the prototype, consisting of 13 workstations, 2 servers and 5 virtual machines. This results in a 20 node cluster with a computational capability of 99 concurrent map tasks (Figure 14).

Once the index is stored, the index parameters are saved in JSON format in a configuration file. This way, the ParaDISE Seeker can use the same configuration for extracting the visual features of the query images when searching within the specific index.

4.1.2 KHRESMOI full 2D image indexing pipeline

In the previous section the visual indexing capabilities of ParaDISE were presented. However, visual indexing is only a part of KHRESMOI indexing pipeline. In Figure 15 the full pipeline

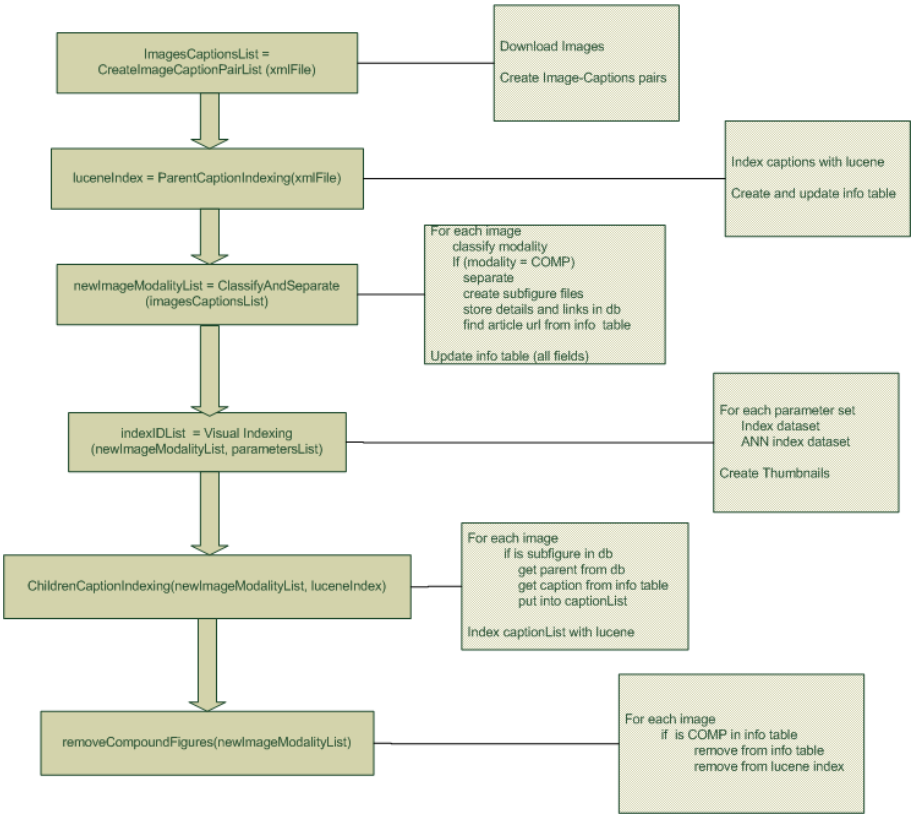

Figure 15: The KHRESMOI indexing pipeline of 2D images.

of 2D image indexing is presented.

In the beginning, the images are downloaded to the server for faster access and caption–images pairs are created. Lucene is used to index the captions of the images. An info table with the various image information, such as the corresponding article URL, the image URL and the caption of the image is created during that step.

The next step is to classify the images according to their image modality. The compound figures are separated and their subfigures are saved as new images and are reclassified. The info table is then updated, including the modality information and the subfigure URLs. The method presented in [23] was used for the modality classification. The method proposed in [12] was used for the compound figure separation. Hadoop was used for the parallelization of this task.

After a new image list is created with the inclusion of subfigures, ParaDISE undertakes the task of visual indexing. For the visual indexing, BoVW and BoC representations were used as shape and color features of the images. E2LSH was used as an ANN indexing method.

A new round of caption indexing is performed, this time on the subfigures captions. The compound figures are then removed from the indices.

A dataset of 1.2 million images from 500.000 articles of PubMed Central has been indexed using this pipeline, resulting in 1.7 million images after subfigure indexation.

4.2 3D clinical images

Parts of this section are also present in KHRESMOI Deliverables D2.5 [34] and D9.4.1 [40].

Indexing is performed in two stages. First, the volume or image is segmented into anatomical regions for which individual feature indices are generated. This is based on detecting landmarks, and localizing anatomical structures in the imaging data. We adopt a global approach and register the entire image to a reference space. Based on this registration region labels are mapped to the image. Then, for each region an index is constructed from the imaging features observed in the corresponding areas across the entire data. Regions can be either anatomical structures (e.g., lung, liver) or spatial neighbourhoods (e.g., a 1 cm^3 grid in the reference space).

4.2.1 Localizing the global position of a volume

This algorithm consists of a training component (`MiniatureTrainingData`) that is executed during indexing , and a localization component that is executed during retrieval (`FragmentCenter`). In this first step the indexing framework identifies the coarse position of a medical imaging volume in a whole body reference space [16]. During training, the center positions for a corpus of several thousands of volumes are annotated by experts. For these volumes miniatures are constructed that form a sampling of the possible appearances sufficiently dense for retrieval. This is similar to [57], but due to the anatomical domain, a significantly lower number of examples is sufficient to sample the appearance space. The training algorithm builds a kd–tree [48] from the miniature descriptors and during localization the center position of a volume is estimated by means of k–nearest neighbour regression based on the annotated examples. A detailed description of the algorithm and a corresponding evaluation can be found in [16, 34] and Deliverable D2.3 [35].

4.2.2 Accurately localizing landmarks

In addition to whole organ mapping, we have developed components that can localize individual anatomical landmarks in medical imaging data. The algorithm that detects landmark configurations follows a two step approach and has been published in [18]. Its learning and localization methodology is described in detail in KHRESMOI Deliverable D2.4 [33]. Given an input volume the algorithm first generates a set of hypotheses locations for each landmark. These hypotheses are based on a global classification of image appearance based on local Hough Forests. In the second step the possible configurations of landmarks formed by all hypotheses (i.e., landmark candidates) are disambiguated based on a statistical shape model learned during training on a small annotated sample set.

A Markov Random Field (MRF) represents the relationships among the landmarks, and the match between the appearance of each hypothesis and the corresponding prototype. The optimal labeling of the MRF yields a reliable estimate for all landmark locations that are part of the configuration. The algorithm performs global search in a volume without the need for initialization. Furthermore, the individual label weights assigned to the MRF by the observed data can serve as a means to detect outliers, or missing landmarks. Typically the number of landmark candidates is low. This is a critical property that allows the algorithm to scale well to high–resolution 3D imaging data.

Similar to the above method, an alternative landmark localization approach focuses on improving speed during localization. The approach, first published in [17], is divided into a training phase and a localization phase. During training the algorithm creates a multi–scale codebook of image patches and landmark positions. It represents local appearance that is specific to landmarks. During localization this codebook is traversed starting from coarse scale image representation to increasingly higher resolution image patches. During this iterative process that starts with a representation of the image similar to the miniature resolution the landmark estimate becomes increasingly accurate. At each step landmark estimates are regularized by a linear statistical shape model that represents the variability in the training set population based on their covariance structures. The resulting algorithm is extremely fast, while achieving high accuracy and reliability in landmark localization.

4.2.3 Dense region label mapping

The location information for a volume can comprise the overall position of the volume in a reference space, the positions of individual landmarks (defined in the reference space) in the volume, and further refinement of a dense correspondence field across the entire volume obtained by non–rigid registration [19].

4.2.4 Indexing of features within anatomical structures

Given all the visual features (and potentially semantic features) for all the volumes, they are analyzed in the nodes *visualEmbedding*, *semanticEmbedding* and *colearning* (Fig 4). The main task of these nodes is the application of state of the art methods in the field of manifold learning. Several approaches (linear embeddings, graph–Laplacian based manifold learning techniques, stochastic embedding techniques) can be employed in this stage and are the focus of current research.

Once the manifold structure has been learned and the features for each supervoxel have been embedded into the low–dimensional, euclidean space this embedding can be indexed. A simple euclidean index (*euclidIndex*) can be used as a baseline, but for efficient retrieval (both in lookup time and spacial representation), sophisticated methods like *product Quantization* are investigated.

4.3 Text

Textual information related to the image, extracted from the web page is indexed into a Solr index [44]. For certain number of extracted fields language information (result of the language detection described in [26]) is incorporated. As a result of this step, language related filters are incorporated in the field analyzer creation [20]. The following fields extracted are indexed taking the web page language into account: alt, title, pageTitle, precedingText, followingText, largePrecedingText, largeFollowingText resulting in “fieldName_lang” index field (e.g. title_en) as illustrated in Figure 16.

```
▼<doc>
   <str name="alt_en">Breast Self Exam: Recline Stroke 6</str>
   <str name="cleanedFilename">bse06</str>
   <str name="contentMD5">BreastSelfExamMD5</str>
   <date name="date">1970-01-01T00:00:00Z</date>
   <str name="docType">image</str>
   <str name="domain">tqn.com</str>
   <str name="followingText_en">Photo Courtesy of National Cancer Institute</str>
  ▼<str name="id">
     http%3A%2F%2Fbreastcancer.about.com%2Fod%2Frisk%2Ftp%2Fbse_illustrated.htm_http%3A%2F%2F0.tq
     %2F-%2Fbse06.jpeg
   </str>
   <str name="imageType">JPEG</str>
   <str name="language">en</str>
  ▼<str name="largeFollowingText_en">
     Photo Courtesy of National Cancer Institute This is best done in your bedroom, where you can
     lie down. Place a pillow on the bed so that you can lie with both your head and shoulders on
     the pillow. Lie down and put your left hand behind your head. Use your right hand to stroke
     the breast and underarm, as you did in step 4. Take note of any changes in texture, color,
     or size. Switch sides and repeat.
   </str>
   <str name="largePrecedingText_en">6. Manual Exam - Recline and Stroke</str>
  ▼<str name="pageTitle_en">
     How To Do Your Breast Self-Exam BSE - Breast Self Exam
   </str>
  ▼<str name="parentUrl">
     http://breastcancer.about.com/od/risk/tp/bse_illustrated.htm
   </str>
   <str name="precedingText_en">6. Manual Exam - Recline and Stroke</str>
   <str name="site">0.tqn.com</str>
  ▼<str name="url">
     http://0.tqn.com/d/breastcancer/1/6/U/1/-/-/bse06.jpeg
   </str>
 </doc>
```


Figure 16: K4E image in the Solr index

5 Retrieval

This section outlines the backend pipelines for on–line search and retrieval. Components included in these pipelines are described. Supported similarity methods and fusion rules are presented.

5.1 2D article images

KHRESMOI uses ParaDISE for similarity search within the visual indices of 2D images and Lucene for text–based search. Section 5.1.1 presents the ParaDISE Seeker, while Section 5.1.2 describes the full KHRESMOI search pipeline for 2D images.

5.1.1 ParaDISE Seeker

As mentioned in Section 2.1.2, the Seeker Composite Component is responsible for CBIR search in ParaDISE. As required by CBIR, the ParaDISE Seeker allows similarity search using image examples as queries. Multiple query images and negative examples are also supported. From the user side, this allows for iterating the search using relevance feedback (RF) [52]. The relevance feedback can be handled in various ways. In ParaDISE Seeker the following algorithms are supported for handling RF:

- **Rocchio Seeker**
  This Seeker uses the Rocchio algorithm [52] to handle multiple images of positive or

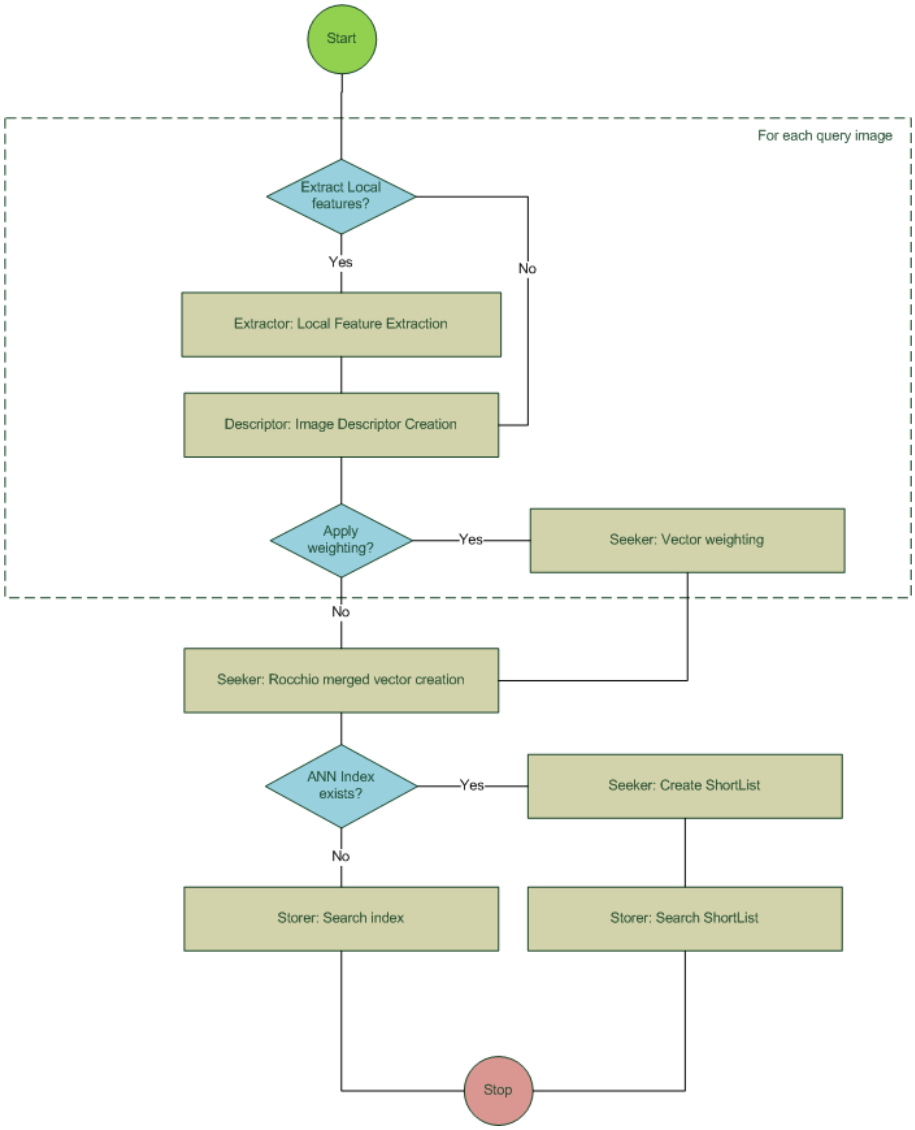

Figure 17: The search pipeline of the Rocchio Seeker.

negative relevance. The Rocchio formula is given by:

$$\vec{q}_m = \alpha\vec{q}_o + \beta\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \gamma\frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}}\vec{d}_j \tag{11}$$

where α, β and γ are weights,
$\vec{q}_m$ is the modified query,
$\vec{q}_o$ is the original query,
D_r is the set of relevant images and
D_{nr} is the set of non–relevant images.

The search pipeline of this method is shown in Figure 17. The Seeker reads the index Parameters from the configuration file of the index it tries to access (see Section 4.1.1). According to these Parameters, it transforms the images to the appropriate vector representations. The Rocchio formula is then executed, producing a single merged vector. If an ANN index exists for the accessed visual index then a shortlist of the vectors existing in the same subspace as the merged vector is returned. In this case, Storer searches within the returned shortlist otherwise the whole index is searched. The similarity search uses a distance metric or a similarity measure to rank the images. The following distances/similarities are supported in ParaDISE:

– **Euclidean Distance (L2 norm)**

$$d(\vec{p},\vec{q}) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \tag{12}$$

– **Manhattan distance (L1 norm)**

$$d(\vec{p},\vec{q}) = \sum_{i=1}^{n}|p_i - q_i| \tag{13}$$

– **Canberra distance**

$$d(\vec{p},\vec{q}) = \sum_{i=1}^{n}\frac{|p_i - q_i|}{|p_i| + |q_i|} \tag{14}$$

– **χ^2 distance**

$$d(\vec{p},\vec{q}) = \frac{1}{2}\sum_{i=1}^{n}\frac{(p_i - q_i)^2}{p_i + q_i} \tag{15}$$

– **Jeffrey divergence**

$$d(\vec{p},\vec{q}) = \sum_{i=1}^{n}(\log\frac{2p_i}{p_i + q_i} + \log\frac{2q_i}{p_i + q_i}) \tag{16}$$

– **Histogram Intersection**

$$s(\vec{p},\vec{q}) = \sum_{i=1}^{n} \min(p_i, q_i) \tag{17}$$

– **Cosine similarity**

$$s(\vec{p},\vec{q}) = \frac{\sum_{i=1}^{n}(p_i \times q_i)}{||\vec{p}|| \times ||\vec{q}_i||} \tag{18}$$

where $\vec{p},\vec{q} \in R^n$. Also special similarity measures are supported for specific approaches:

– **Hamming Distance**
For binary vectors $\vec{p},\vec{q}$, the hamming distance $d(\vec{p},\vec{q})$ is defined as the number of ones of $p \oplus q$. It can be used for comparing binary representations, such as binary BoVW.

– **Frequent Item Selection Distance**
This similarity is used in combination with the Frequent Selection weighting (see Section 4.1.1). The similarity score is equal to the number of common shared frequent items.

- **LateFusion Seeker**
The pipeline of this Seeker is demonstrated in Figure 18. It is similar to Rocchio Seeker pipeline but instead of producing a single merged query vector it initiates a different search for each positive query image. In the end the Fusor ParaDISE Component is used to fuse the retrieved lists. Negative query image examples are ignored. The fusion rules supported in Fusor are:

– **CombSUM**

$$S_{\texttt{combSUM}}(i) = \sum_{k=1}^{N_k} S_k(i) \tag{19}$$

where $S_k(i)$ is the score assigned to image i in retrieved list k.

– **CombMNZ**

$$S_{\texttt{combMNZ}}(i) = F(i) * S_{\texttt{combSUM}}(i) \tag{20}$$

where $F(i)$ is the number of times an image i is present in retrieved lists with a non–zero score, and $S(i)$ is the score assigned to image i.

– **CombMAX**

$$S_{\texttt{combMax}}(i) = \max_k S_k(i) \tag{21}$$

where $S_k(i)$ is the score assigned to image i in retrieved list k.

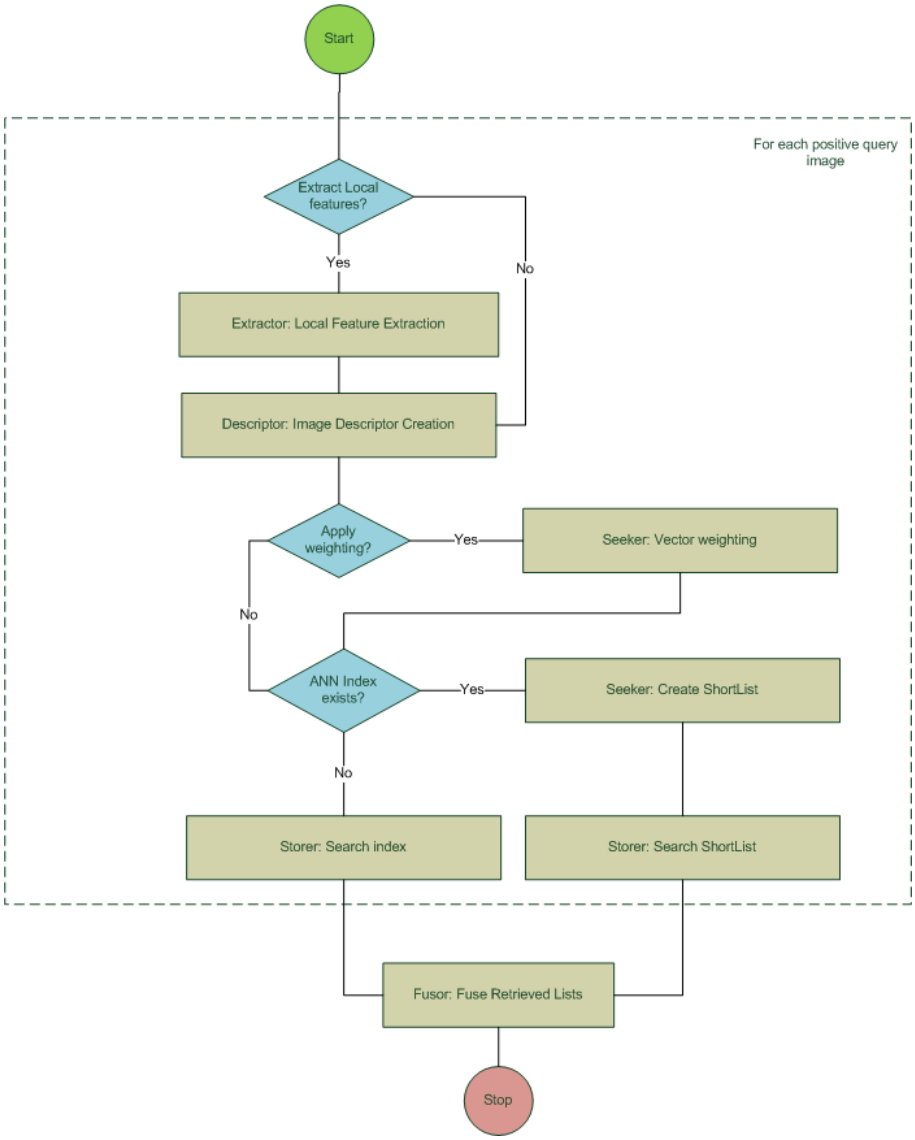

Figure 18: The search pipeline of the LateFusion Seeker.

– **CombMIN**

$$S_{\texttt{combMin}}(i) = \min_k S_k(i) \tag{22}$$

where $S_k(i)$ is the score assigned to image i in retrieved list k.

– **Linear Weighting**

$$S_{\texttt{linear}}(i) = \sum_{k=1}^{N_k} w_k S_k(i) \tag{23}$$

with $w_k \in [0,1]$ and $\sum_{k=1}^{N_k} w_k = 1$.

– **Borda Count**

$$S_{\texttt{Borda}}(i) = \sum_{k=1}^{N_k} \frac{1}{R_k(i)} \tag{24}$$

where $R_k(i)$ the rank of the image in retrieved list k

– **Reciprocal Rank**

$$S_{\texttt{RRF}}(i) = \sum_{k=1}^{N_k} \frac{1}{c + R_k(i)} \tag{25}$$

where c a constant and $R_k(i)$ the rank of the image in retrieved list k.

5.1.2 KHRESMOI full 2D image search pipeline

In the previous section the visual similarity search options of ParaDISE were presented. However, the development of a new Seeker was dictated by the requirements for the KHRESMOI system, such as search by modality. The object–oriented implementation of ParaDISE facilitated this and ModalityFilter Seeker was created. This Seeker extends the Rocchio Seeker and accepts as inputs a list of modalities with wich to filter the results. The weights used for the Rocchio algorithm are $\beta = 0.6$ and $\gamma = 0.4$. Query and relevant vectors were considered as the same set of vectors.

The backend 2D image search pipeline is presented in Figure 19. Once the web service is called, the call arguments dictate the behavior of the work flow. Query Images can be automatically classified to produce a list of target modalities or specific target modalities can be passed as arguments. If text is included in the query then the text search pipeline is enabled (in the left frame). Image captions can also be used in relevance feedback iterations. RadLex terms can be extracted by the captions of the query images using the ONTOtext disambiguation service [42] and be added to the query string. Captions of negative query image examples have their terms (the ones not present in positive ones) added using the NOT boolean operator.

The next step is the visual similarity search. For each visual index needed to be accessed there is a concurrent search using modality filtering. The Histogram Intersection similarity measure is used. If there is no text included in the query, the ANN index is used to build the shortlist to be searched. Otherwise, the top results returned by the text query constitute the

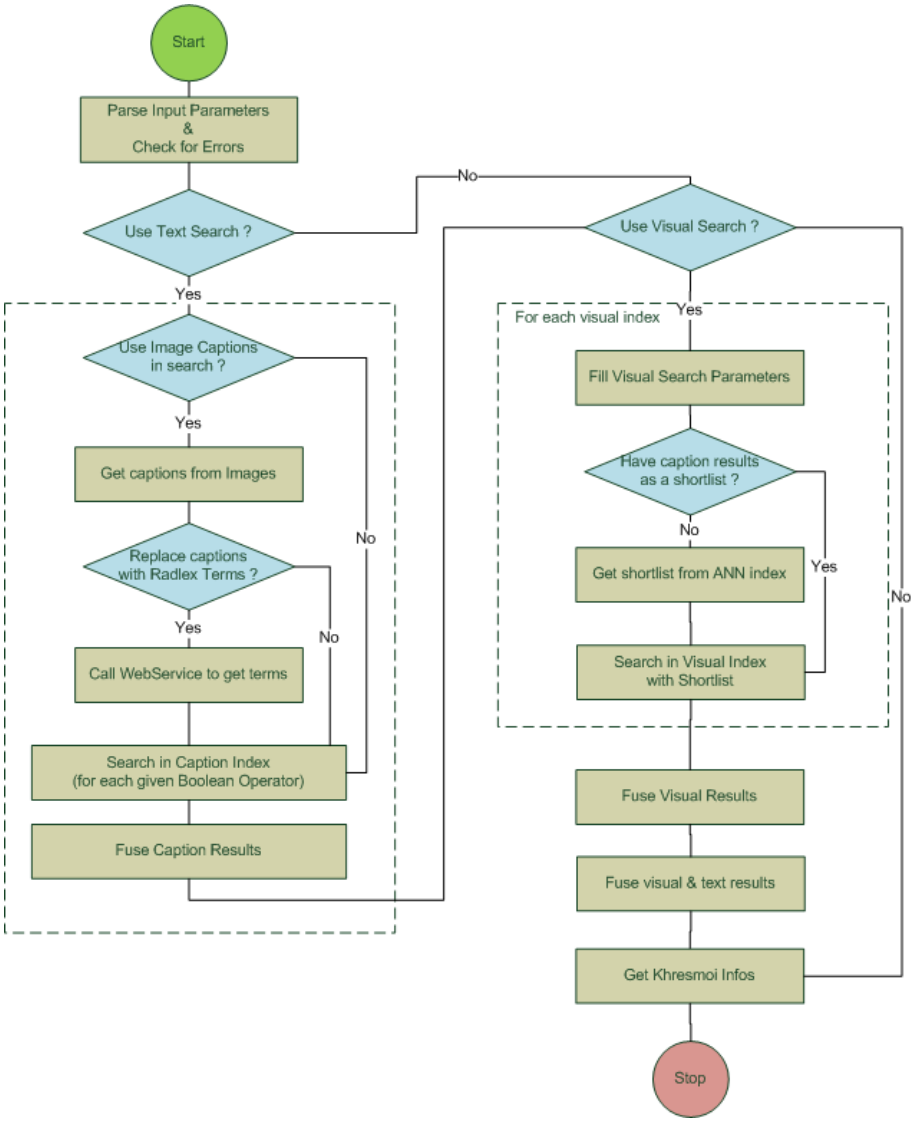

Figure 19: The KHRESMOI search pipeline for 2D image retrieval.

shortlist for the visual search. The ParaDISE Fusor then is used to fuse the retrieved lists from the visual indices. The CombMNZ rule is used for this fusion.

The next step consists of the fusion of the text and visual search results, using the Fusor and Reciprocal Rank Fusion rule. Finally, image information existing in the info table (see Section 4.1.2) is added to the results.

5.2 3D clinical images

Parts of this section are also present in KHRESMOI Deliverable D9.4.1 [40].

5.2.1 Identifying location

The nodes in the data flow graph in Fig. 4 are run at training time and represent potentially long–running computations. The MapReduce framework also allows to be used in time–sensitive areas, as the computation overhead is en–par with less general approaches. In the case of the retrieval GUI, a query has to be received from the HTTP Application Programming Interface (API) provided by MUW and the corresponding retrieval results have to be returned.

This first involves the mapping of the volume ID and user–indicated region of interest (ROI) to the indices of the supervoxels (*querysps* and *queryfeatures*). As all the volumes are mapped to the common reference frame this also yields information about which body region's index should used.

5.2.2 Search in the feature index

As each of the supervoxels has a unique, implicit ID within the entire system the lookup step is very straightforward, as the lookup nodes (e.g. *euclidlookup* or *productQuantizerLookup*) are provided with embedded feature vectors as query elements, and are thus operating in a euclidean space for both training and query, without any knowledge of or constraints imposed by the 3D retrieval system.

The result of the lookup is a sorted list, together with similarity values, for several hundred of the supervoxels that are closest to the query supervoxels in feature space. The node *retrieval* combines this information about the individual supervoxels into a ranked list of most similar cases / volumes, which together with meta–data, is sent to the GUI client.

For the visual representation of the results and especially to indicate the region of similarity within the result volume, the gray–level volume is overlaid with a semi–transparent rendering of the similarity scores in *retrievaloverlay*. Upon request, this rendering is returned to the client for display in the GUI.

5.3 Text

The text based image retrieval is performed using full–text search in the following Solr/Lucene index fields: alt_lang, title_lang, pageTitle_lang, precedingText_lang, followingText_lang, largePrecedingText_lang, largeFollowingText_lang where "lang" represents the language of the interface chosen by the user, as illustrated in Figure 20.

6 Conclusion

In this deliverable we describe the image retrieval and analysis framework that serves the retrieval prototypes in KHRESMOI. While the image retrieval framework serves different retrieval scenarios, and user groups, the underlying structure across this uses is similar. Images are described by extracting local features, and contructing image descriptors that allow for effective comparison of images, and the calculation of similarity scores. Indices are constructed from these features, in order to allow for fast retrieval. Finally retrieval is based either on entire query images, or on query regions of interests, and results in a ranked list of cases, most similar to the query.

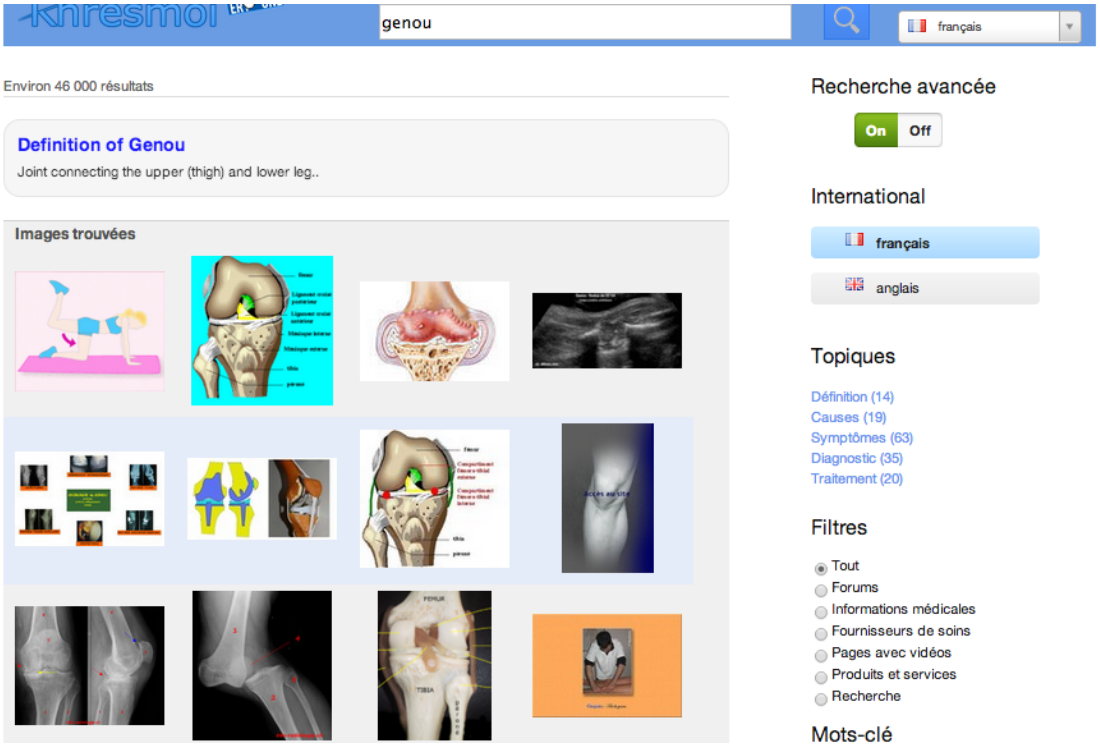

Figure 20: K4E image results, interface in French

References

[1] Mohamed Aly, Mario Munich, and Pietro Perona. Indexing in large scale image collections: Scaling properties and benchmark. In *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, pages 418–425. IEEE, 2011.

[2] Alexandr Andony and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *47th Annual IEEE Symposium on Foundations of Computer Science, 2006. FOCS '06*, pages 459–468, 2006.

[3] Relja Arandjelovic and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2911–2918. IEEE, 2012.

[4] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006*, pages 404–417. Springer, 2006.

[5] T. Beckers, S. Dungs, N. Fuhr, M. Jordan, S. Kriewel, and V.T. Tran. An interactive search and evaluation system. *Open Source Information Retrieval*, 9, 2012.

[6] Thomas Beckers, Sebastian Dungs, Norbert Fuhr, Matthias Jordan, and Sascha Kriewel. Final flexible user interface framework and documentation. Deliverable d3.6 of the khresmoi project, University of Duisburg, February 2014.

[7] Andreas Burner, René Donner, Marius Mayerhoefer, Markus Holzer, Franz Kainberger, and Georg Langs. Texture bags: anomaly retrieval in medical images based on local 3d-texture similarity. In *Medical Content-Based Retrieval for Clinical Decision Support*, pages 116–127. Springer, 2012.

[8] Gustavo Carneiro and Allan D Jepson. The distinctiveness, detectability, and robustness of local image features. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 296–301. IEEE, 2005.

[9] Shih-Fu Chang, Thomas Sikora, and Atul Purl. Overview of the mpeg-7 standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 11(6):688–695, 2001.

[10] Savvas A. Chatzichristofis and Yiannis S. Boutalis. CEDD: Color and edge directivity descriptor: A compact descriptor for image indexing and retrieval. In *Lecture notes in Computer Sciences*, volume 5008, pages 312–322, 2008.

[11] Savvas A. Chatzichristofis and Yiannis S. Boutalis. FCTH: Fuzzy color and texture histogram: A low level feature for accurate image retrieval. In *Proceedings of the 9th International Workshop on Image Analysis for Multimedia Interactive Service*, pages 191–196, 2008.

[12] Ajad Chhatkuli, Dimitrios Markonis, Antonio Foncubierta-Rodríguez, Fabrice Meriaudeau, and Henning Müller. Separating compound figures in journal articles to allow for subfigure classification. In *SPIE Medical Imaging*, 2013.

[13] Júlia Epischina Engrácia de Oliveira, Arnaldo de Albuquerque Araújo, and Thomas M Deserno. Content-based image retrieval applied to bi-rads tissue classification in screening mammography. *World journal of radiology*, 3(1):24, 2011.

[14] Jeffrey Dean and Sanjay Ghemawat. MapReduce: simplified data processing on large clusters. *Communications of the ACM — 50th anniversary issue*, 51:107–113, January 2008.

[15] Adrien Depeursinge, Antonio Foncubierta-Rodríguez, Dimitri Van De Ville, and Henning Müller. Multiscale lung texture signature learning using the Riesz transform. In *Medical Image Computing and Computer–Assisted Intervention MICCAI 2012*, volume 7512 of *Lecture Notes in Computer Science*, pages 517–524. Springer Berlin / Heidelberg, October 2012.

[16] Rene Donner, Sebastian Haas, Andreas Burner, Markus Holzer, Horst Bischof, and Georg Langs. Evaluation of fast 2D and 3D medical image retrieval approaches based on image miniatures. In Hayit Greenspan, Henning Müller, and Tanveer Syeda-Mahmood, editors, *Medical Content-based Retrieval for Clinical Decision Support*, volume 7075 of *MCBR-CDS 2011*. Lecture Notes in Computer Sciences (LNCS), September 2011.

[17] René Donner, Björn Menze, Horst Bischof, and Georg Langs. Fast Anatomical Structure Localization Using Top-down Image Patch Regression. In *Proc. MICCAI Workshop on Medical Computer Vision*, 2012.

[18] René Donner, Björn Menze, Horst Bischof, and Georg Langs. Global Localization of 3D Anatomical Structures by Pre-filtered Hough Forests and Discrete Optimization. *Medical Image Analysis, submitted*, 2013.

[19] Matthias Dorfer, René Donner, and Georg Langs. Constructing an un-biased whole body atlas from clinical imaging data by fragment bundling. In *Proc. MICCAI'13*, 2013.

[20] Apache Software Foundation. Lucene. `http://lucene.apache.org`, 2011.

[21] Apache Software Foundation. Couchdb. `http://couchdb.apache.org/`, 2012.

[22] Alba García Seco de Herrera, Antonio Foncubierta-Rodríguez, Emanuele Schiavi, and Henning Müller. 2D–based 3D volume retrieval using singular value docomposition of detected regions. In *MICCAI workshop on Medical Computer Vision*, Lecture Notes in Computer Science. Springer, 2013.

[23] Alba García Seco de Herrera, Dimitrios Markonis, and Henning Müller. Bag of colors for biomedical document image classification. In Hayit Greenspan and Henning Müller, editors, *Medical Content–based Retrieval for Clinical Decision Support*, MCBR–CDS 2012, pages 110–121. Lecture Notes in Computer Sciences (LNCS), October 2013.

[24] Donald E Gustafson and William C Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *Decision and Control including the 17th Symposium on Adaptive Processes, 1978 IEEE Conference on*, volume 17, pages 761–766. IEEE, 1978.

[25] Ju Han and Kai-Kuang Ma. Fuzzy color histogram and its use in color image retrieval. *IEEE Transactions on Image Processing*, 11(8):944–952, 2002.

[26] Allan Hanbury, William Belle, Nolan Lawson, Ljiljana Dolamic, Natalia Pletneva, Matthias Samwald, and Célia. D8.3: Prototype of a rst search system for intensive tests. *Khresmoi project public deliverable*, 2012.

[27] Zujun Hou. A Review on MR Image Intensity Inhomogeneity Correction. *International Journal of Biomedical Imaging*, 2006.

[28] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *Computer Vision–ECCV 2008*, pages 304–317. Springer, 2008.

[29] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Aggregating local descriptors into a compact image representation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3304 – 3311, June 2010.

[30] Eiji Kasutani and Akio Yamada. The MPEG–7 color layout descriptor: a compact image feature description for high-speed image/video segment retrieval. In *Proceedings of the International Conference on Image Processing*, ICIP'2001, pages 674–677, 2001.

[31] Yan Ke, Rahul Sukthankar, and Martial Hebert. Efficient visual event detection using volumetric features. In *Proc. ICCV*, 2005.

[32] Georg Langs, Andreas Burner, Joachim Ofner, Rene Donner, Henning Müller, Adrien Depeursinge, Dimitrios Markonis, Alexandre Masselot, and Nolan Lawson. Report on and prototype for feature extraction and image description. Deliverable d2.2 of the khresmoi project, Medical University of Vienna, 2012.

[33] Georg Langs, Rene Donner, Dimitrios Markonis, Matthias Dorfer, and Henning Müller. Report on the robust learning from incoplete and spurious data. Deliverable d2.4 of the khresmoi project, Medical University of Vienna, 2013.

[34] Georg Langs, Rene Donner, and Dorfer Matthias. Report on the anatomical structure identification and localization. Deliverable d2.5 of the khresmoi project, Medical University of Vienna, 2013.

[35] Georg Langs, Joachim Ofner, Andreas Burner, Rene Donner, Henning Müller, Adrien Depeursinge, Dimitrios Markonis, Celia Boyer, Alexandre Masselot, and Nolan Lawson. Report on results of the wp2 first evaluation phase. Deliverable d2.3 of the khresmoi project, Medical University of Vienna, 2012.

[36] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the 2006 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR, pages 2169–2178, Washington, DC, USA, 2006. IEEE Computer Society.

[37] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[38] Mathias Lux and Savvas A. Chatzichristofis. Lire: lucene image retrieval: an extensible java CBIR library. In *Proceedings of the 16th ACM international conference on Multimedia*, pages 1085–1088, October 2008.

[39] T. Mäenpää. University of Oulu, Aug 2003.

[40] Dimitrios Markonis, Rene Donner, Sebastian Dungs, Roger Schaer, Markus Holzer, Matthias Jordan, Sascha Kriewel, Georg Langs, and Henning Müller. Intermediate version of the final prototype. Deliverable d9.4.1 of the khresmoi project, University of Applied Sciences Western Switzerland, 2013.

[41] Dimitrios Markonis, Markus Holzer, Sebastian Dungs, Alejandro Vargas, Georg Langs, Sascha Kriewel, and Henning Müller. A survey on visual information search behavior and requirements of radiologists. *Methods of Information in Medicine*, 51(6):539–548, 2012.

[42] Ivan Martinez and Miguel Angel Tinte. Prototype and evaluation of the full software architecture. Deliverable d6.3.3 of the khresmoi project, ATOS, 2013.

[43] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 27(10):1615–1630, 2005.

[44] Mark Miller. Lucene and solr development have merged. http://www.lucidimagination.com/blog/2010/03/26/lucene-and-solr-development-have-merged/, 2010.

[45] T Ojala, M Pietikainen, and D Harwood. *Performance evaluation of texture measures with classification based on Kullback discrimination of distributions*, volume 1. IEEE, 1994.

[46] T Ojala, M Pietikainen, and D Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59, 1996.

[47] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175, 2001.

[48] Beng Chin Ooi, Ken J McDonell, and Ron Sacks-Davis. Spatial kd-tree: An indexing mechanism for spatial databases. In *In Proc. IEEE COMPSAC Conf*, pages 433–438, 1987.

[49] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[50] Natalia Pletneva, Sascha Kriewel, and Marlene Kritz. D8.5.2: Prototype of a second search system based on feedback. *Khresmoi project public deliverable*, 2013.

[51] Angus Roberts, Johann Petrak, Célia Boyer, Ljiljana Dolamic, Allan Hanbury, Michael Dittenbach, and Julien Gobeill. D1.7: Prototype and report on semantic indexing and annotation for information retrieval. *Khresmoi project public deliverable*, 2014.

[52] J. J. Rocchio. Relevance feedback in information retrieval. In *The SMART Retrieval System, Experiments in Automatic Document Processing*, pages 313–323. Prentice Hall, Englewood Cliffs, New Jersey, USA, 1971.

[53] Denis Shestakov, Diana Moise, Gylfi Thór Gudmundsson, Laurent Amsaleg, et al. Scalable high-dimensional indexing with hadoop. In *CBMI—International Workshop on Content-Based Multimedia Indexing*, 2013.

[54] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pages 1470–1477, Washington, DC, USA, 2003. IEEE Computer Society.

[55] David McG. Squire, Wolfgang Müller, Henning Müller, and Jilali Raki. content–based query of image databases, inspirations from text retrieval: inverted files, frequency–based weights and relevance feedback. pages 143–149.

[56] Hideyuki Tamura, Shunji Mori, and Takashi Yamawaki. Textural features corresponding to visual perception. *IEEE Transactions on Systems, Man and Cybernetics*, 8(6):460–473, June 1978.

[57] A Torralba, R Fergus, and WT Freeman. 80 Million Tiny Images: A large Data Set for Nonparametric Object and Scene Recognition. *TPAMI*, 2008.

[58] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. pages 511–518, 2001.

[59] Winn Voravuthikunchai, Bruno Crémilleux, Frédéric Jurie, et al. Finding groups of duplicate images in very large dataset. In *Proceedings of the British Machine Vision Conference (BMVC 2012)*, 2012.

[60] Christian Wengert, Matthijs Douze, and Hervé Jégou. Bag–of–colors for improved image search. In *Proceedings of the 19th ACM international conference on Multimedia*, MM '11, pages 1437–1440, New York, NY, USA, 2011. ACM.

[61] Tom White. *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 2010.

[62] Jun Yang, Yu-Gang Jiang, Alexander G Hauptmann, and Chong-Wah Ngo. Evaluating bag–of–visual–words representations in scene classification. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, pages 197–206. ACM, 2007.