Grant Agreement Number: 257528

KHRESMOI

www.khresmoi.eu

D2.2: Report on and prototype for feature extraction and image description

Deliverable number	*D2.2*
Dissemination level	*Public*
Delivery data	*31.5.2012*
Status	*Final*
Authors	*Georg Langs, Andreas Burner, Joachim Ofner, René Donner, Henning Mueller, Adrien Depeursinge, Dimitrios Markonis, Célia Boyer, Alexandre Masselot, Nolan Lawson*

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

This deliverable encompasses the prototype and description of the feature extraction, and image description for content based medical image retrieval in the KHRESMOI project. Image content is a central source of information during the retrieval of relevant cases during clinical routine, research, and teaching activities. A prerequisite for efficient and meaningful image retrieval is a rich description of image content. We describe image content by means of three feature types. First, primary features capture local appearance properties, based on simple a priori defined feature extractors. Secondary features collect primary features across local neighborhoods, to obtain a statistical characterization of the local appearance. Image descriptors aggregate information over an entire image, or volume. An important observation is that to provide for effective retrieval of medical images or cases, that e.g., exhibit similar pathologies, feature learning has to be performed. That is, in order to achieve sufficiently high specificity, feature extractors have to be adapted to the imaging data, or even to specific anatomical regions.

The present document describes the methods explored in the course of KHRESMOI, and documents the resulting modular prototype framework that serves as basis for image description in the learning- and retrieval framework.

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

LBP	Local Binary Patterns
PACS	Picture archiving and communication system
ImageCLEF	Image Retrieval in the Cross Language Evaluation Forum
GLCM	Gray Level Coocurrence Matrix
SIFT	Scale-Invariant Feature Transform
DoG	Difference of Gaussians
EMEA	European Medicines Agencey
Europarl	Europarl: A Parallel Corpus for Statistical Machine Translation
MeSH	Medical Subject Headings
BoC	Bags of Colors
BoVW	Bags of Visual Words
DICOM	Digital Imaging and Communications in Medicine
SVM	Support Vector Machine
CT	Computed Tomography
MRI	Magnetic Resonance (Imaging)

1 Introduction

The KHRESMOI project aims at developing a content based medical information retrieval framework that makes relevant information accessible during clinical practice and during research [22]. A key to identifying relevant data is image information provided by a diverse and growing number of medical imaging modalities used in a clinical environment.

Consequently, a core focus of KHRESMOI is the use of image content itself for retrieval. To this end, image data has to be characterized in a manner that allows for computational processing and analysis of the encoded information. We summarize this effort as *feature extraction and image description*. The present deliverable describes the work performed in the course of the project, and the resulting prototype, a flexible framework for feature extraction and image description.

To facilitate reading, we will call all imaging data *image*, regardless of whether it is a 2D image, or a 3D volume. In cases where this differentiation is important we will be specific regarding which data is described in the respective section.

1.1 Requirements and further use of features and descriptors.

Within the project objectives, the features extracted from images serve the following purposes:

- **Comparison and similarity among images**: Quantitative comparison of anatomical structures and anomalies observed in medical imaging data. The comparison establishes relationships among observations by means of *similarity measures*.
- **Classification of observations**: Observations such as modality, anatomical structure, or pathology are classified to obtain more relevant search results, or to support steps such as training of structure specific vocabularies, and retrieval. The features are a basis for classifier learning, and application during retrieval.
- **Identification and localization of anatomical structures**: Anatomical structures, and groups of structures have to be identified both during retrieval, and during training, when unsupervised methods capture structure in the data set. The learning of this structure (e.g., similar anatomical regions across cases) is necessary to support an efficient retrieval during application.

For each of the points, specificity and sensitivity is highly relevant. Features have to be both stable with regard to variability of acquisition, or normal variability across the population, while at the same time they have to be sufficiently specific to allow for the recognition of anatomy, and pathologies. This requires feature adaptation approaches that learn anatomy-, modality-, or even disease specific feature extractors based on training data.

1.2 Features and descriptors

To cope with the requirements, we use three main categories of feature computation, that are also reflected in the prototype.

1. **Primary features** describe the appearance at individual points such as voxel- or pixel-location in the data, and are extracted locally. Examples are local wavelet features.

2. **Secondary features** describe regions, such as *super-voxels* or *super-pixels* and are typically feature vectors that summarize the statistical properties of primary features within a region.

3. **Image level features** integrate features across an entire image to obtain an *image descriptor*.

For these features both fixed feature extractors, as well as adaptive feature extractors that are learned based on a training data set are used.

1.3 Outline

In this document we will first outline the main exemplary tasks for which image features are needed (Sec. 2). Following this, we will describe primary- and secondary features that capture image content, or contextual information (Sec. 3.1). Following this we will describe image level features in detail (Sec. 3.3), together with an explanation of the adaptation methodology that tunes feature extractors with regard to specific regions, or example sets. Following this, the prototype framework will be described (Sec. 4). The conclusion of the document is formed by a discussion of the choices made during the prototype design (Sec. 5).

2 Tasks

Features and image descriptors are used for various tasks within KHRESMOI. In the following three main tasks will be outlined, to motivate the methods described in the subsequent sections. The three tasks illustrate that the various scenarios covered by KHRESMOI require different features, and that a modular retrieval system needs a flexible feature extraction framework that allows for the integration of a number of feature extractors, and the modular use during indexing, and retrieval. The three tasks are *Anatomy retrieval*, *Pathology retrieval*, and *Modality recognition*. They were chosen to highlight different requirements relevant for the overall system.

Anatomy retrieval calls for features that are able to identify anatomical regions depicted in an image, and are stable with regard to natural - and substantial - variability across the population, or due to image acquisition. They are aimed at ranking images according to their overall similarity, and are trained for specific image sets. Pathology retrieval necessitates features that are able to capture subtle variations due to different pathologies. To work, they need to be adapted to specific anatomical regions, and the corresponding candidate pathologies. This is necessary, since the overall variability in the human anatomy far outweighs most local deviations caused by disease. Finally the modality classification task typical for document retrieval was chosen, since it makes features necessary, that capture statistical properties and differences of among images, that are entirely different from those within anatomy observed via a single modality.

In the following we will describe the tasks in detail, and will refer to them throughout the description of the features, and descriptors.

2.1 Anatomy retrieval

Task: *Identify the anatomical region in the query image, and retrieval images that show this region.*

In a typical clinical setting radiologists base their diagnosis on the image data of the patient at hand, drawing on their expertise (i.e. acquired, implicit image models) or reference textbooks. Typically there is a wealth of information present in every hospital's PACS that is relevant to the case. A first step, in finding relevant images when performing retrieval, is to find images that show the same anatomical site, identify the anatomical regions depicted in the imaging data, and possibly locate the precise position of a region of interest in the query case. A similar requirement is present during training, when feature vocabularies have to be trained for specific anatomical structures. Fig. 1 shows an example result when performing image retrieval based on anatomical similarity.

The goal is to efficiently retrieve a subset of overall visually highly similar images for a given query image. The resulting set of images can then be either directly displayed or passed on to modality, region or pathology specific analysis stages. For example, this is relevant in the context of anatomical structure localization approaches such as [7], where separate models need to be employed for each anatomical region.

The characteristics of medical images, and radiological data in particular, warrant a closer look at holistic image representations as a means for this task. Medical imaging protocols are standardized, meaning the positioning of body parts in the image varies only within a certain

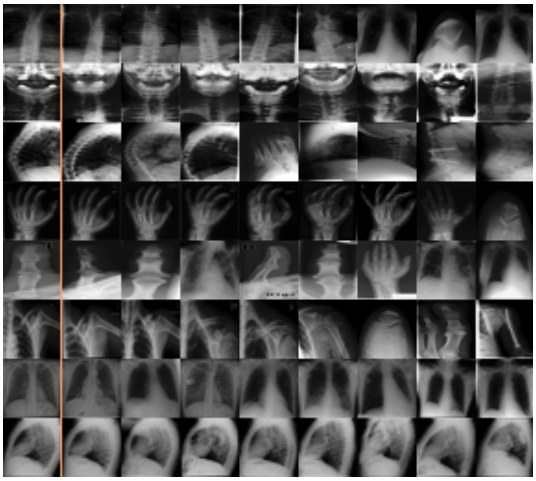


Figure 1: Content-based medical image retrieval focusing on the anatomical region: given a query image (first column), the most similar images from a given set are found, in this case the ImageCLEF 2009 data set. Results from the method published in [6].

range within the data set of images of the same body region. A large proportion of the acquired images are from a few body regions and view points, and overall imaging characteristics (i.e. post-processing kernels in lung CTs) are typically similar for similar diagnostic questions. On the other hand the features have to cope with substantial variability in size, overall shape, and appearance already in a healthy population. This variability is further increased, if patient data is included.

2.2 Pathology retrieval

Task: *Retrieve cases that show similar pathologies.*

Image retrieval in a clinical setting aims at providing the clinical radiologist with cases that have a pathology similar to the query case. This can be used either for exploring diagnoses of similar looking cases, or for differential diagnosis, where cases with similar appearance but alternative diagnosis are compared to the query case. Since pathologies can exhibit far more subtle features in radiological data compared to overall anatomical variability in a large population, pathology retrieval has to learn and adapt features for specific contexts. This occurs during a learning phase where example data is presented to the training algorithm, and features, or descriptors are learned on a region- or anatomy-specific basis [3].

In this scenario a physician typically marks a region of interest in a query image at hand and starts retrieval on the image repository. The system returns images with regions containing similar pathological structures. This enables the physician to compare the current case to past cases and gain knowledge by comparing the diagnosis, treatment, and progress of the disease.

2.3 Modality recognition

Task: *Identify the modality of an image, either during training of the retrieval index, or during retrieval.*

Several studies have shown that the imaging modality is an important aspect of the image for medical information retrieval. In user-studies, clinicians have indicated that modality is one of the most important filters that they would like to be able to limit their search by. Many image retrieval websites (Goldminer[1], Yottalook[2]) allow users to limit the search results to a particular modality.

However, this modality is typically extracted from the caption text and is often not correct or even present at all. Studies have shown that the modality can be extracted from the image itself using visual features. Additionally, using the modality classification, the search results can be improved significantly. For example, the images of the ImageCLEFmed 2011 collection which constitutes a realistic sample of medical literature, originate from a large variety of biomedical journals and the captions are not of the same high quality as radiology journals [27].

[1] http://goldminer.arrs.org/
[2] http://www.yottalook.com//

3 Methods

This section explains the methodology used for feature extraction and image description in the KHRESMOI project. We explain relevant feature extractors and image descriptors. A sub-set of these is implemented in the prototype framework. Its primary characteristic, is that it allows for a modular design of feature extractors, and addition of new methods used during indexing an retrieval. The design choices are discussed in Sec. 5

Throughout the section we refer to an image or volume in a set of N images by $\mathbf{I}_i$, $i = 1,2,\ldots,N$, to individual positions in an image by $\mathbf{x}$, and to the corresponding value in the image by $\mathbf{I}_i(\mathbf{x})$. A feature vector that correspond to a particular position in the image will be denoted by $\mathbf{f}(\mathbf{x})$, and a descriptor that represents an entire image by $\mathbf{F}_i$. Notation beyond this will be introduced in the respective sections describing features, and image descriptors.

3.1 Features

3.1.1 Local binary patterns

Note, the following description has been partly published in [3].

The original *linear binary pattern (LBP)* operator computes the local structure at a given position x in the image $\mathbf{I}$ by comparing the values of its eight neighborhood pixels $\{x_{N1}, x_{N2}, \ldots, x_{N8}\}$ with the value of $\mathbf{I}(x)$, to obtain a binary code where for each neighbor $\mathbf{f}(x)_{Ni} = 0$ if $\mathbf{I}(x) < \mathbf{I}(x_{Ni})$, and $\mathbf{f}(x)_{Ni} = 1$ otherwise. It can be implemented by summing weighted bits (Fig. 2(a)) yielding a value from 0 to 255 for each pixel that describes the neighborhood-relative gray-scale structure. Various extensions of the original operator have been published in recent years[3].

In KHRESMOI we identified LBP descriptors to be a suitable feature extractor to identify micro structures in tissues or bones, and on a larger scale, recognize their macro structure. In the framework we implemented a fast 3D version of LBPs with a special focus on the medical domain:

A Three-dimensional, Multi-scale LBP Descriptor: The three dimensional LBP descriptor is based on the same concept as in 2D, however, the number of neighbours is 26. Therefore, the 26 weighted bits result in value of 0 to 67108864 for each voxel. During initial experiments [3] we found that the local contrast is an important property in the medical domain, therefore, our method is based on LBP/C [33], which combines the base LBP operator $\mathbf{F}_{LBP}$, with a local contrast measure $\mathbf{F}_C$.

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 $\mathbf{F}_I$ of the 3x3x3 LBP cube to the feature vector. The relevance of both features, $\mathbf{F}_C$ and $\mathbf{F}_I$, is configurable and can be defined based on the modality. The LBP3d/CI descriptor $\mathbf{f}$ for a position x in an image is defined as:

$$\mathbf{f}(x) = [\mathbf{f}_{LBP3d}(x), c_c\mathbf{f}_C(x), c_i\mathbf{f}_I(x)] \quad (1)$$

In total the descriptor $\mathbf{f}(x)$ has 28 dimensions: 26 LBP bits, one contrast dimension, and one intensity dimension. The factors c_c and c_i determine the impact of the two features, contrast

[3] An extensive list of LBP bibliography can be found at `http://www.cse.oulu.fi/MVG/LBP_Bibliography` (accessed June 2011)

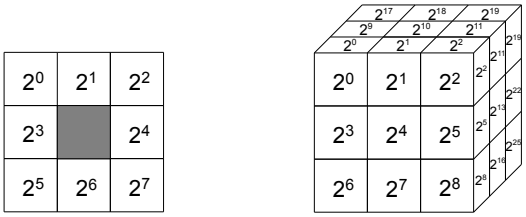


Figure 2: Weights of Local Binary Patterns (LBP): a. 2D LBP [32, 33], b. 3D LBP used by our texture descriptor $\mathbf{f}(x)$.

and intensity.

3.1.2 Wavelets

Note, this description is partly published in [5]

The Riesz transform is a multidimensional extension of the Hilbert transform, which maps any function $\mathbf{I}(x)$ to its harmonic conjugate and is a very powerful tool for mathematical manipulations of periodic signals [40]. For an image $\mathbf{I}_i(x) \in \mathbb{R}^2$, the different components of the Nth–order Riesz transform $\mathcal{R}$ are defined in the Fourier domain as

$$\widehat{\mathcal{R}^{(n_1,n_2)}}\mathbf{I}_i(\omega) = \sqrt{\frac{n_1+n_2}{n_1!n_2!}}\frac{(-j\omega_1)^{n_1}(-j\omega_2)^{n_2}}{||\omega||^{n_1+n_2}}\hat{\mathbf{I}}_i(\omega), \tag{2}$$

for all combinations of (n_1,n_2) with $n_1+n_2=N$ and $n_{1,2} \in \mathbb{N}$. $\hat{\mathbf{I}}_i(\omega)$ denotes the Fourier transform of $\mathbf{I}_i(x)$, where the vector ω is composed by $\omega_{1,2}$ corresponding to the frequencies in the two image axes. The multiplication by $j\omega_{1,2}$ in the numerator corresponds to partial derivatives of f and the division by the norm of ω in the denominator results in only phase information being retained. Therefore, the 1st–order $\mathcal{R}$ corresponds to an allpass filterbank with directional (singular) kernels $h_{1,2}$:

$$\mathcal{R}\mathbf{I}_i(x) = \begin{pmatrix}\mathcal{R}^{1,0}\\ \mathcal{R}^{0,1}\end{pmatrix} = \begin{pmatrix}h_1(x) * \mathbf{I}_i(x)\\ h_2(x) * \mathbf{I}_i(x)\end{pmatrix}, \tag{3}$$

where

$$h_{1,2}(x) = \frac{x_{1,2}}{2\pi||x||^3}, \tag{4}$$

and $x_{1,2}$ correspond to the axes of the image [44]. The Riesz transform commutes with translation, scaling or rotation. The orientation of the Riesz components is determined by the partial derivatives in Eq. (2). Whereas 2^N Riesz filters are generated by (2), only $N+1$ components have distinct properties due to commutativity of the convolution operators in (3) (e.g., $\partial^2/\partial x\partial y$ is equivalent to $\partial^2/\partial y\partial x$). The Riesz components yield a steerable filterbank [44] allowing to analyze textures in any direction, which is an advantage when compared to classical Gaussian derivatives or Gabor filters. Qualitatively, the first Riesz component of even order corresponds to a ridge profile whereas for odd ones we obtain an edge profile, but much richer profiles can be obtained by linear combinations of the different components. The templates of $h_{1,2}(x)$ convolved with Gaussian kernels for N=1,2,3 are depicted in Fig. 3. The Nth–order Riesz transform can be coupled with an isotropic multiresolution decomposition (e.g., Laplacian of Gaussian (LoG)) to obtain rotation–covariant (steerable) basis functions [44].

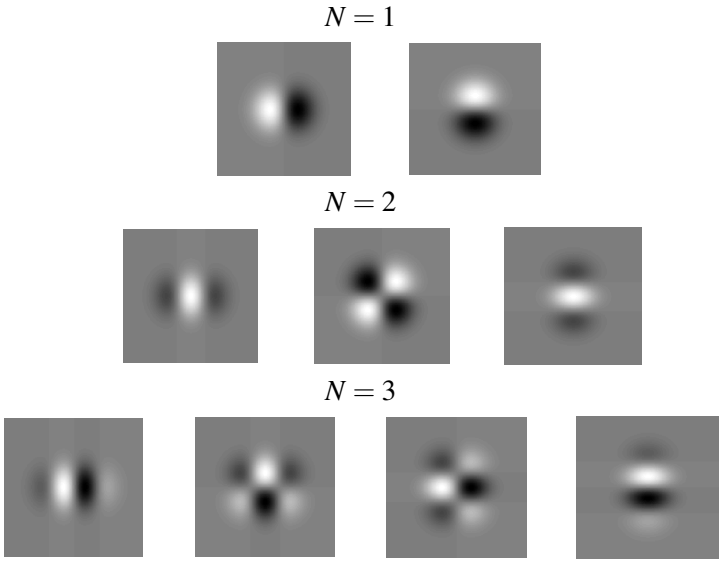
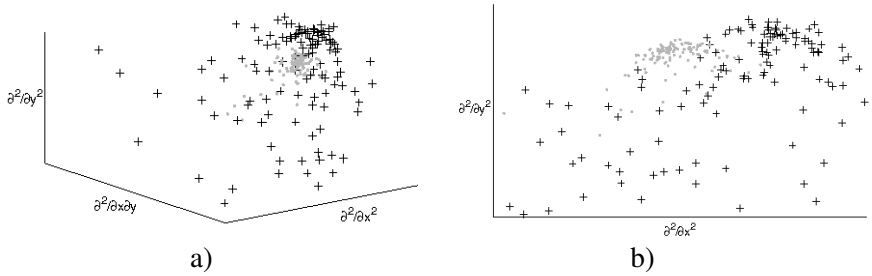


Figure 3: Templates corresponding to the Riesz kernels convolved with a Gaussian smoother for N=1,2,3.



a) b)

Figure 4: Riesz representation of healthy (gray dots) versus fibrosis (black crosses) patterns. a) Initial Riesz coefficients in 3D. b) The Riesz coefficients in 2D after having locally aligned the texture based on local prevailing orientation. The component corresponding to $\partial^2/\partial x\partial y$ is zero after local rotation and is not shown in b).

The main idea of the proposed approach is to derive texture signatures from multiscale Riesz coefficients. An example showing healthy and fibrosis tissue represented in terms of their Riesz components with N=2 is depicted in Fig. 4 a). In order to provide a local categorization of the image, image regions in 2D axial slices are divided into 32×32 overlapping blocks with a distance between contiguous block centers of 32. The Riesz transform is applied to each block, and every Riesz component $n = 1,\ldots,N{+}1$ is mapped to a multiscale representation by convolving them with four LoG filters of scales $s = 1,\ldots,4$ with a dyadic scale progression.

The local dominant texture orientations have an influence on the repartition of respective responses of the Riesz components [5], which is not desirable for creating robust features with well–defined clusters of instances. For example, a rotation of $\pi/2$ will switch the responses of h_1 and h_2 for N=1. To ensure that the repartitions of $w_{n,s}$ are comparable for two similar textures having distinct local prevailing orientations, the filters are oriented to have maximal response

along h_1. The dominant orientation θ_{dom} of h_1 at the position x_p is

$$\theta_{dom}(x_p) = \arg\max_{\theta\in[0,\pi]} \left(\left(h_1^{(\theta)} * g \right) * f \right)(x_p), \tag{5}$$

where $h_1^{(\theta)}(x)$ is h_1 rotated by θ and $g(x)$ is a Gaussian kernel. A local orientation is obtained by rotating every Riesz filter h_n with θ_{dom} and is done analytically [44]. 2nd–order Riesz coefficients of healthy and fibrosis tissue after local orientation are shown in Fig. 4 b).

In a total of $(N+1)\times 4$ subbands, the energy $w_{n,s}(x)$ of the Riesz coefficients over a block centered at x are used as texture features $f(x)$ as:

$$\mathbf{f}(\mathbf{x}) = [w_{1,1}(x), \ldots, w_{1,4}(x), w_{2,1}(x), \ldots, w_{2,4}(x), \quad \ldots \quad , w_{N+1,1}(x), \ldots, w_{N+1,4}(x)]. \tag{6}$$

3.1.3 Cooccurence

Gray Level Co-occurrence Matrices (GLCM) were proposed by Haralick in 1973 [13]. They are widely used statistical texture features. GLCM is based on the joint probability distributions of pixel pairs [43]. A GLCM is calculated for rectangular image patches P_s with size s centered at x in the image $I(x)$. The gray levels of the image are reduced to G gray levels. A GLCM C_d is calculated for a specific distance parameter d and is defined as:

$$C_d(i,j) = |(t,v) : P_s(t) = i, P_s(v) = j| \tag{7}$$

where t and v are pixel positions in the image patch P_s with gray levels i and $j \in 1 \ldots G$. $v = t + d$ and $|.|$ is the cardinality of a set. In other words C_d indicates how often a gray level i at position t occurs relative to the gray level j of pixel in the distance d. The distance can also be seen as an offset and is defined as $d(r,a)$, where r is a radial distance in pixels and a an angle in degrees. For every image patch four GLCMs with fixed r and the angles $0°, 45°, 90°, 135°$ are calculated to represent gray level coocurrences in various directions. To achieve rotation invariance the mean of this four GLCMs represents the final GLCM $C_{mean}(x)$ for an image patch centered at image position x. It is not feasible to use the whole GLCM as feature vector, therefore Haralick proposed 13 scalar measures $H_1 \ldots H_{13}$ that can be calculated from a GLCM to represent the underlying properties of texture [13]. Namley following Haralick features are used: Energy (Angular Second Moment), Contrast (Inertia), Correlation, Sum Of Squares (Variance), Inverse Difference Moment (Homogenity), Sum Average, Sum Variance, Sum Entropy, Entropy, Difference Variance, Difference Entropy, infomation measure of correlation 1 and information measure of correlation 2.

The feature vector $f(x)$ parameterized by the patch size s around x, the neighborhood distance $d(r,(0°,45°,90°,135°))$ and the number of gray levels G is then

$$\mathbf{f}_{(d;s;G)}(x) = [H_1(C_{mean}(x)), H_2(C_{mean}(x)), \ldots, H_{13}(C_{mean}(x))]. \tag{8}$$

3.1.4 Scale-Invariant Feature Transform (SIFT)

Note this description is partly published in [28]

The Scale–Invariant Feature Transform [20] (SIFT) has been widely used in object recognition [48, 35], scene categorization [14, 17], concept detection [45], and scalable image retrieval [14] and classification [31], as it has proven to be robust and distinctive [29].

The method of extracting the SIFT features – as described in [20] – 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:

$$\mathbf{DoG}(x,y;s) = \mathbf{L}(x,y;s+\Delta s) - \mathbf{L}(x,y;s) \approx \frac{\Delta s}{2}\nabla^2 L(x,y;s) \tag{9}$$

These minima and maxima are candidates as interest points (keypoints). Low contrast candidate points and edge response points along an edge are discarded for rubstness to noise. Finally, dominant orientations are assigned to localized keypoints to proivde 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 $i = 1\ldots16$ summarizing the contents over 4×4 subregions, taken by a 16×16 array around the keypoint. This results into an $4\times4\times8 = 128-d$ vector for each keypoint:

$$\mathbf{f}(\mathbf{x}) = [\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_{16}] \tag{10}$$

In recent studies a sparse sampling for the selection of the keypoints was employed successfully. For combatibility to the framework we used this technique for the lung tissue classification task instead of interest point detection.

3.1.5 Textons

In 1981 Julesz et al. [15] discovered that the pre-attentive human vision detects atomic structures like blobs, edges, crossed and terminators, while ignoring other structures. He named this basic patterns for texture description textons for the first time.

Malik et al. [25] provide a mathematical model for describing and generating this textons. Images are convolved with a set of oriented filters representing models of receptive fields of simple cells in visual cortex. The filterbank consists of three parts: Radially symmetric receptive fields are modeled by a Difference of Gaussians (DoG) with two Gaussians having different values of σ. Receptive fields of oriented even-symmetric cells are modeled as rotated copies of a Gaussian second derivative $fb_{even}(x,y) = G''_{\sigma1}(y)G_{\sigma2}(x)$. Where $G_\sigma(x)$ is a Gaussian with standard deviation of σ. The ratio $\sigma2:\sigma1$ represents the elongation of the filter. Oriented odd-symmetric fields are modeled by the Hilbert transform of fb_{even}: $fb_{odd}(x,y) = Hilbert(fb_{even}(x,y))$. The filters can have various scales and are zero-mean and L_1 normalized.

Clustering the filter responses with a K-means technique yields K cluster centers, representing prototypes of typical appearance, or textons T_k. The final textons feature vector is constructed by mapping the filter responses of each voxel to the most similar cluster center T_k^s by means of a Nearest Neighbour algorithm.

$$\mathbf{f}(x) = T_k^s(x). \tag{11}$$

3.2 Learning domain and data specific features: texture bags

A central insight when performing retrieval in medical imaging data, is that appearance features which characterize specific pathologies can by subtle compared to the natural variability in the healthy population. Therefore it is crucial to adapt descriptors to specific anatomical sites and their potential pathologies. The adaptation of descriptors allows for features that reflect the variability present at a specific site, as opposed to the variability in the entire body. This yields higher specificity, and is important if pathologies are of interest.

As we describe in [3], we learn the structure of features present in a training data set, to obtain a vocabulary of features that are optimal for describing the variability present in a specific region. The features extracted from training data are quantized by performing clustering on the feature descriptors F_i of a large set of voxels randomly sampled from the anatomical structure across the training set . We refer to the k clusters that formulate the texture vocabulary as texture words W_k. Analogous to Textons [18, 25], we represent each voxel with its closest texture word $\mathbf{W}_k^s$, i.e., with the index of the closest cluster center.

In figure 1, three example texture words are depicted that have been computed by the method explained above. The figure shows three texture words from a lung vocabulary and examples below that show corresponding real-world instances. If the processing step of learning texture vocabularies is executed on a different anatomical region, the learned texture vocabulary will encode a different set of domain specific structures, and therefore will be distinct from the vocabulary shown for lungs in figure 1.

In [3] we describe how to compute a feature of a region $\mathbf{R}$ by the histogram $h(\mathbf{R})$ of *texture words* $\mathbf{W}_k$ it contains. This is analogous to the *bag of visual word* paradigm of Sivic et al. [38]. We call this k-bin histogram $h(\mathbf{R})$ a *texture bag*. Similar to [38] describing local patches, our histogram describes a region in terms of its textural structure. We normalize the texture histogram to make comparison possible without considering the size of a region.

3.3 Image description

The features described in the previous sections capture local appearance properties of the image, or its context. In the following section we will explain how descriptors for entire images can be calculated. These *image descriptors* are either based on derived from the local features obtained in an image, or are descriptors that capture the image characteristics in a holistic approach. The aim is to find descriptors, that allow for efficient comparison, classification, and retrieval.

In practice different descriptors are needed to serve different purposes. For example, while holistic descriptors are well suited to retrieve images with specific anatomical regions within

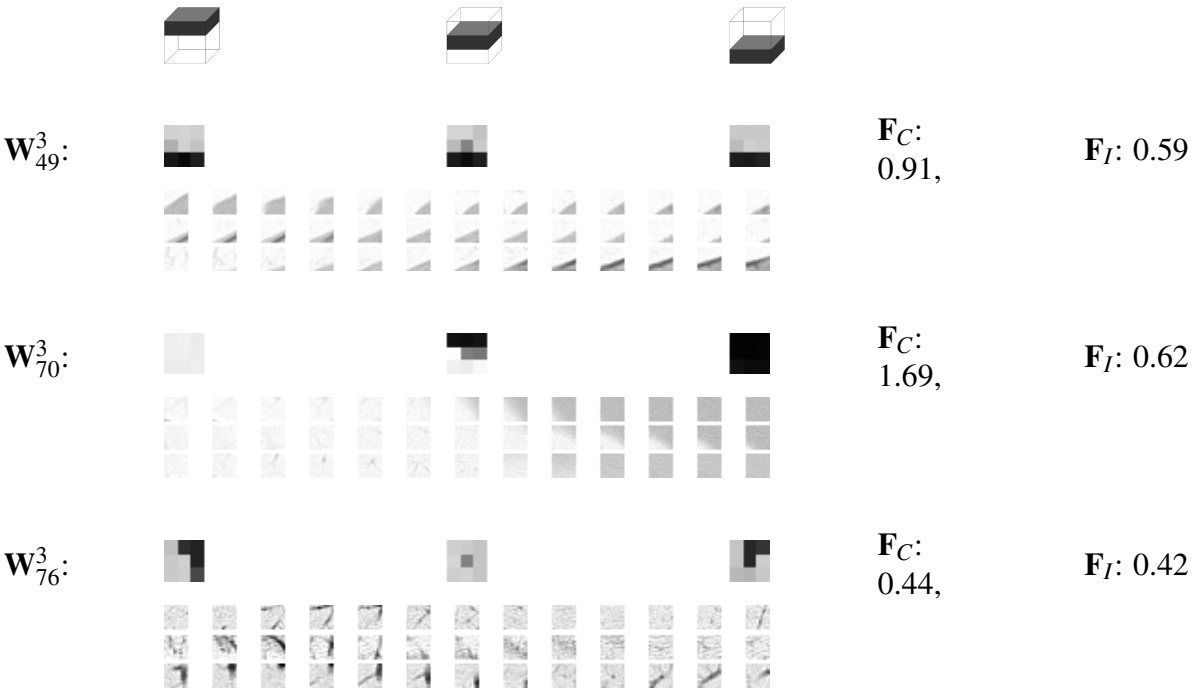


Table 1: Three example 3D *texture words* $\mathbf{W}^3_{49}$, $\mathbf{W}^3_{70}$, and $\mathbf{W}^3_{76}$ trained in an unsupervised manner on lung tissue on scale 3. The slices of the cube from top to bottom are depicted from left to right. Next to the cube are the measures for contrast ($\mathbf{F}_C$) and intensity ($\mathbf{F}_I$). Below each *texture words* are three examples of lung tissue belonging to this word. Note the similar structure of the example-tissues for each *texture word* [3].

a hospital system, local features are better suited to capture characteristics of pathologies. In the following we outline the most prominent features used in the project. Some have been published, and the respective references are provided in the individual sections. The descriptors are aimed at serving tasks such as those described in Sec. 2. To facilitate reading, we can sub-divide the descriptors in 4 categories:

1. 2D image retrieval
    - Bags of colors
    - Bags of visual words
2. 3D anatomy retrieval
    - Image miniatures
    - Distribution fields
    - Histograms of gradients
3. 3D pathology retrieval:
    - 2D and 3D anatomy retrieval:
4. Web-site, or document retrieval based on context:
    - Textual features

It is important to note that this is not an exhaustive list of possibly relevant descriptors, and one of the main design goals of the framework is to allow for straight-forward integration of new modules.

3.3.1 Image description for 2D image retrieval

Note this description is partly published in [28]

A state–of–the–art approach for image description using the SIFT feature in large datasets is the bag–of–visual–words representation. The pipeline of the bag–of–visual–words approach is shown in Figure 5. A training set of images is chosen and local descriptors (in the case of SIFT, 128–dimensional vectors) are extracted from interest points of each image of this set. The descriptors are then clustered using a clustering method into k and the centroids of the clusters are used as visual words. The visual vocabulary $\mathbf{V}$ represents all cluster centers

$$\mathbf{V} = \{v_1, \ldots, v_k\}, \;\; v_i \in \mathbb{R}^{128}, \;\; i = 1, \ldots, k \tag{12}$$

Then, the local visual 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 SIFT vector $\mathbf{f}(\mathbf{x})$ extracted from the image $\mathbf{I}$:

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

where

$$g_j(\mathbf{f}(\mathbf{x})) = \begin{cases} 1 \; if \; d_\varepsilon(\mathbf{f}(\mathbf{x}), v_j) \leq d_\varepsilon(\mathbf{f}(\mathbf{x}), v_l) \; \forall v_l \in \mathbf{V} \\ 0 \; otherwise \end{cases} \tag{13}$$

When a new image is classified, a similarity measure is used to compare the histogram of the new image with the histograms of the training images, providing a similarity score for images in the training set. Preliminary tests showed that the use of histogram intersection as a similarity measure outperformed other distance metrics both in speed and precision. Histogram intersection of 2 n-binned histograms $\mathbf{L}$ and $\mathbf{M}$ can be given by:

$$H(\mathbf{L}, \mathbf{M}) = \sum_{i=1}^{n} \min(\mathbf{L}(i), \mathbf{M}(i))$$

The SIFT implementation existing in the Fiji image processing package[4] was used for the local feature description of the images, while histogram intersection was applied for similarity calculation. *Note this description is partly included in a paper submitted to MICCAI 2012 "Medical Content–Based Retrieval for Clinical Decision Support" Workshop*

Since the SIFT descriptor uses a greyscale version of the image, a different technique can be used to include the chromatic information. The Bags–of–Colors (BoC) is a method to extract a color signature from images introduced by [46]. It is based on the Bag–of–Visual–Words image representation [12]. Each image is represented by a BoC from a color vocabulary C previously learned on a sub set of the collection.

We used the CIE (International Commission on Illumination) 1976 L*a*b (CIELab) space in our method because it is a perceptually uniform color space recommended by CIE. CIELab is a space defined by L for luminance and a, b for the color–opponent dimensions for chrominance [37, 1].

A color vocabulary $C = \{c_1, \ldots, c_{k_c}\}$, with $c_i = (L_i, a_i, b_i) \in CIELab$ is constructed by finding the most frequently occurring colors in each image of the subset of the collection, in our case from the 100 selected images. Then, the colors are clustered using a k–means algorithm [24]. We use for our experiments mainly $k_c = 200$ found by an analysis on the training set.

The final image descriptor of image $\mathbf{I}$, called Bag–of–colors, is defined as a vector $\mathbf{F}(\mathbf{x}) = \{\bar{c}_1, \ldots, \bar{c}_k\}$ such that, for each pixel $p_k \in \mathbf{I} \; \forall k \in \{1, \ldots, n_p\}$, with n_p being the number of pixels of the image $\mathbf{I}$:

$$\bar{c}_i = \sum_{k=1}^{n_p} \sum_{j=1}^{n_p} g_j(p_k) \;\; \forall i \in \{1, \ldots, k_c\}$$

where

$$g_j(p) = \begin{cases} 1 \; if \; d_\varepsilon(p, c_j) \leq d_\varepsilon(p, c_l) \; \forall l \in \{1, \ldots, k_c\} \\ 0 \; otherwise \end{cases} \tag{14}$$

[4] http://fiji.sc/

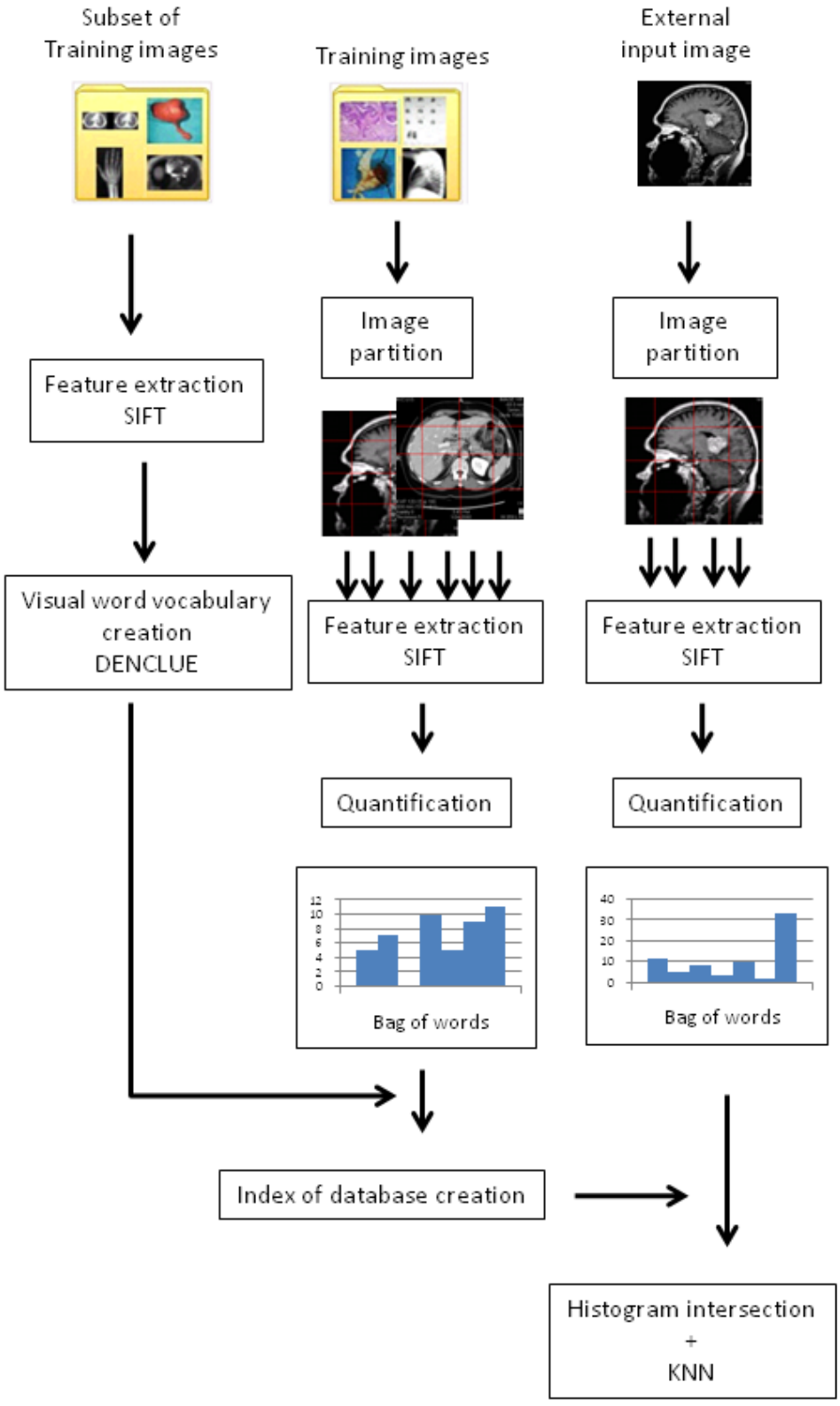


Figure 5: Overview of the approach for the extraction of visual words.

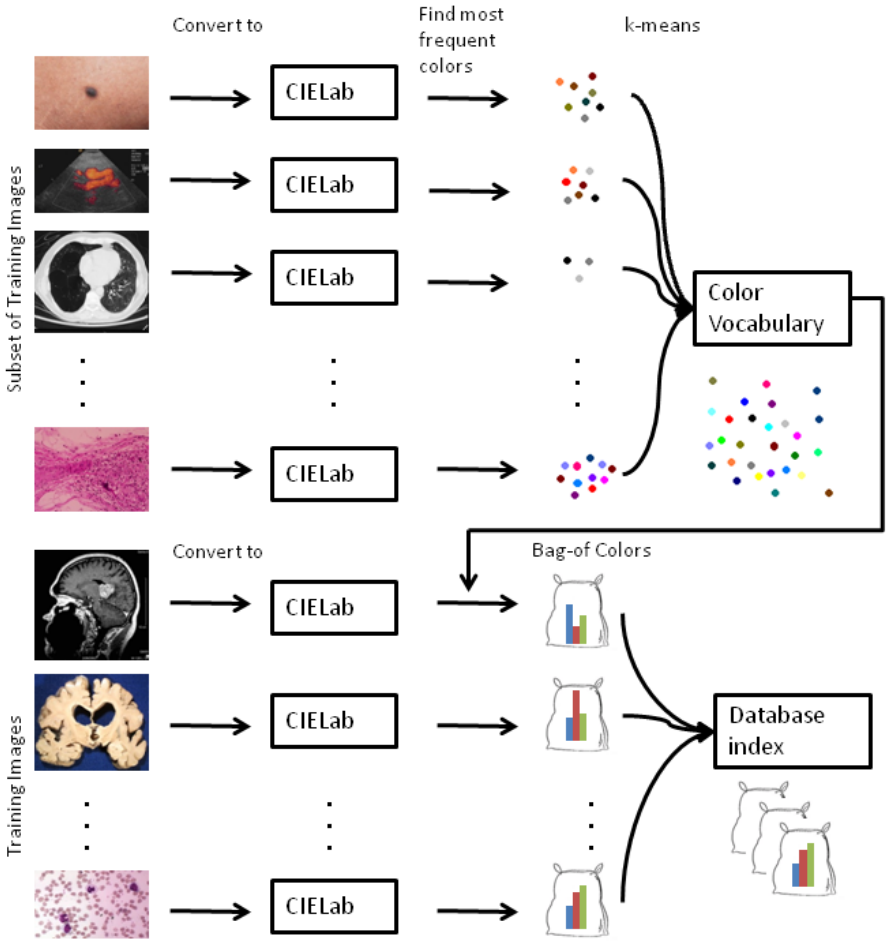


Figure 6: The procedure for constructing the BoC.

This procedure is described graphically in Figure 6.

3.3.2 Image Miniatures as Descriptors for 3D Anatomy Retrieval

Note, part of this work has been published in [6].

While the number of available training images in medical data sets is smaller than in [42], we argue that due to the constrained domain - only radiographs or CTs, only of the human body - the training space can also be considered to be densely populated and that a euclidean distance between image descriptors formed from miniature images provides a reasonable initialization for further optimization.

Given the set of training images $I = \{\mathbf{I}_1, \ldots, \mathbf{I}_N\}$ each image is rescaled to size 32×32 to form the descriptors $\mathbf{D} = (\mathbf{d}_i, \ldots, \mathbf{d}_N)$. PCA is applied to $\mathbf{D}$, retaining the factors $1 \ldots p$ with maximal variance to cover 98% of variance and projecting yields low dimensional descriptors $\mathbf{d}_i^{PCA}$. The resulting p-dimensional space $\mathbf{D}^{PCA} = \left(\mathbf{d}_1^{PCA}, \ldots, \mathbf{d}_N^{PCA}\right)$ thus contains all training data. To obtain a considerable speedup at query time, a kD-tree $\mathcal{K}$ is built from the training data.

Image Query Given a query or test image $\mathbf{I}_t$ and its associated descriptor $\mathbf{d}_t$, a k-nearest neighbor search is performed in the training space using the kD-tree $\mathcal{K}$, yielding an initial set of m descriptors ($m = 100$ in our experiments). Up to now the euclidean distance assumption was employed and the resulting distances e_{ij} between images i and j are used in the first evaluation approach to estimate the performance contribution of the following step.

To be able to account for rigid image deformations and contrast and brightness variations in the data, the following refined distance metric is employed: For each of the $\mathbf{d}_j \in \mathbf{d}_i, \ldots, \mathbf{d}_m$ miniatures the rotation α, scaling s and translation parameters t_x, t_y are estimated by maximizing the correlation

$$c_j = max \;\; corr(\mathbf{d}_t, T(\mathbf{d}_j, \alpha, s, t_x, t_y)) \tag{15}$$

where $T(\mathbf{d}_j, \alpha, s, t_x, t_y)$ represents the miniature $\mathbf{d}_j$ after rotation, scaling and translation. The actual maximization is performed using fixed increments for all 4 parameters, which allows for fast optimization using precomputed pixel indices. Using $2 - c_j$ as the new distance measure yields the final order by similarity of the m miniatures for the query image $\mathbf{I}_t$.

While a k-NN approach does not necessarily require a training phase, the kD-tree proves beneficial by ensuring retrieval times in the range of a few milli seconds for the first stage of the retrieval. The optimization of the rigid registration required in the order of a few seconds per query image in a Matlab implementation, which can be easily improved.

3D Volume Retrieval To perform retrieval of volumetric data, in our case 3D CTs, we extend the approach as follows. Descriptors $\mathbf{d}_i$ are computed as $16 \times 16 \times 16$ volumes and the transformation $T(\cdot)$ now takes into account the additional rotation parameters β and γ as well as the translation t_z. For each tuple $(\mathbf{d}_t, \mathbf{d}_j)$, we optimize

$$c_j = max \;\; corr(\mathbf{d}_t, T(\mathbf{d}_j, \alpha, \beta, \gamma, s, t_x, t_y, t_z)). \tag{16}$$

For computational efficiency, not the entire volume $\mathbf{d}_i$ is transformed, but only the axis-parallel 16×16 slices $\mathbf{d}^x, \mathbf{d}^y, \mathbf{d}^z$ through the center of the volume. This results in the more efficient maximization

$$\mathbf{d}^{xyz} = ((\mathbf{d}^x)^T, (\mathbf{d}^y)^T, (\mathbf{d}^z)^T)^T \tag{17}$$

$$c_j = max \;\; corr(d_t^{xyz}, T(d_j^{xyz}, \alpha, \beta, \gamma, s, t_x, t_y, t_z)). \tag{18}$$

3.3.3 Distribution Fields (DFs) for 3D Anatomy Retrieval

The second type of descriptors investigated are Distribution Fields (DFs) [36]. They split an image into b separate channels containing only information from pixels which lie in the corresponding gray level interval. A given normalized image I_i with values in the range $[0,1]$ is split into a set of channels $\mathcal{C}_i = \{\mathbf{C}_1^i, \ldots, \mathbf{C}_c^i, \ldots, \mathbf{C}_b^i\}$ such that channel $\mathbf{C}_c^i$ at position (x,y) is

$$\hat{\mathbf{C}}_c^i(x,y) = \mathbf{I}_i(x,y) > \frac{c-1}{b} \wedge \mathbf{I}_i(x,y) < \frac{c}{b}. \tag{19}$$

$$\mathbf{C}_c^i = \hat{\mathbf{C}}_c^i \circ \mathbf{G}(\sigma), \tag{20}$$

where $\mathbf{G}(\sigma)$ performs a spatial smoothing of the channel with a Gaussian filter with standard deviation σ. In our case the images are the miniatures, thus the descriptors are all of the same size. As suggested in [36], we can use the l_1-norm to compute distances d_{ij} between the image descriptors $\mathbf{C}^i$ and $\mathbf{C}^j$:

$$d_{ij} = \sum_{c,x,y} |\mathbf{C}^i_c(x,y) - \mathbf{C}^j_c(x,y)| \tag{21}$$

The miniature volumes employed for the 3D set are treated in the same way, yielding descriptors of size $16 \times 16 \times 16 \times b$. For both the 2D and 3D data set $b = 10$ channels were used.

3.3.4 Histograms of Gradients (HOGs) for 3D Anatomy Retrieval

To encode image information through a set of histograms of the gradient orientations in different parts of the image or image patch has seen tremendous success in computer vision [21]. It is also employed in the BVWs methods used for comparison in this paper.

Each miniature $\mathbf{I}_i$ is described by one SIFT-like descriptor $\mathbf{H}_i$ as follows. In the 2D case, using the usual 4×4 grid of histograms with 8 bins for the gradient directions $(0, \frac{\pi}{4}, \ldots, \frac{7}{4}\pi)$ yields a 128-dimensional descriptor. In the 3D case $\mathbf{H}_i$ is computed as a $4 \times 4 \times 4$ grid of histograms with $8 * 8$ directional bins, resulting in a 4096-dimensional descriptor space. The chi-squared distance $\chi^2(\mathbf{H}_i, \mathbf{H}_j)$ is used as distance metric.

3.3.5 Image description for 3D pathology retrieval

For the scenario described in the introduction, the retrieval query consists of a medical image $\mathbf{I}_Q$ and a marked query region $\mathbf{R}_Q$. Our method aims to retrieve images with regions most similar to $\mathbf{R}_Q$. To compare the texture of areas, the corresponding texture bags are compared by the diffusion distance. Figure 8 shows an overview of the retrieval pipeline. This section has been partly published in [3].

Precomputation of the Images in the Imaging Repository For the purpose of grouping similar areas and reducing the complexity of the three-dimensional image, we perform a precomputing step for each image $\mathbf{I}_j$ of the repository. This precomputing step fragments each image into several texture bags. We chose to use a supervoxel algorithm for this purpose and apply the method of Wildenauer et al. [47]. The result for a lung volume $\mathbf{I}_j$ is a three-dimensional oversegmentation $\mathbf{R}_{js}$ for the image j and the supervoxel index s, shown in figure 7. For each region $\mathbf{R}_{js}$, we precompute a *texture bag*, a histogram $h(\mathbf{R}_{js})$ of occurring texture words $\mathbf{W}_k$.

Computing Similarities to the Retrieval Query To compare the marked regions $\mathbf{R}_Q$ of the query image $\mathbf{I}_Q$ to all regions $\mathbf{R}_{js}$, a normalized texture word histogram $h(\mathbf{R}_Q)$ is computed. The distance between the histogram of the query region $\mathbf{R}_Q$ and the regions $\mathbf{R}_{js}$ is computed by the diffusion distance.

$$\mathbf{d}_{js} = d(\,h(\mathbf{R}_Q), h(\mathbf{R}_{js})\,) \tag{22}$$

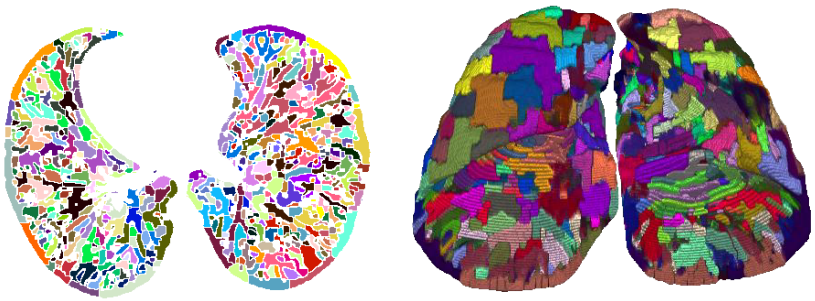


Figure 7: Supervoxel algorithm applied on lung volumes. This figure depicts the oversegmented regions $\mathbf{R}_{js}$ in 2D on the left, and in 3D on the right.

Ranking of the Image Set The final step of the retrieval pipeline computes a ranking of all images $\mathbf{I}_j$ based on the number of "close" regions $\mathbf{R}_{js}$ to the query region $\mathbf{R}_Q$. For this ranking, the number of regions $\mathbf{R}_{\mathbf{I},s}$ are considered that are amongst the most similar regions $\mathbf{R}_{js}$ to $\mathbf{R}_Q$, in terms of diffusion distance. The threshold t of regions taken into account is dependent on the average region size $\mathbf{R}_{js}$, therefore dependent on the number of superpixels s per image $\mathbf{I}_j$.

Figure 9 shows the result of two retrieval queries. The first example is a query to retrieve patterns that are typical for centrilobular emphysema: round black spots, with typically less tissue structure than healthy lung tissue. The second example query retrieves patterns that are characteristic for panlobular emphysema: large areas with very little tissue lung structure. The results on the right side show regions with small distance to the query region $\mathbf{R}_Q$, i.e., regions where the texture histograms $h(\mathbf{R}_Q)$ and $h(\mathbf{R}_{js})$ are similar.

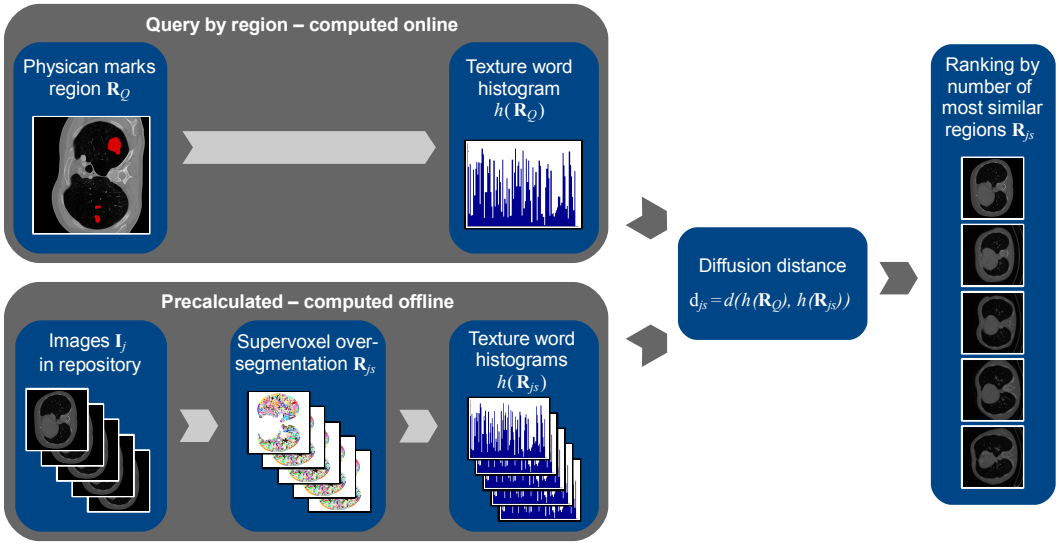


Figure 8: Overview of the 3D retrieval pipeline. Top left: the retrieval scenario, a physician marks a region in an image. Lower left: precomputation of the image set. Right side: comparison by distance (e.g., diffusion distance [19]) and ranking by the number of most similar regions [3]

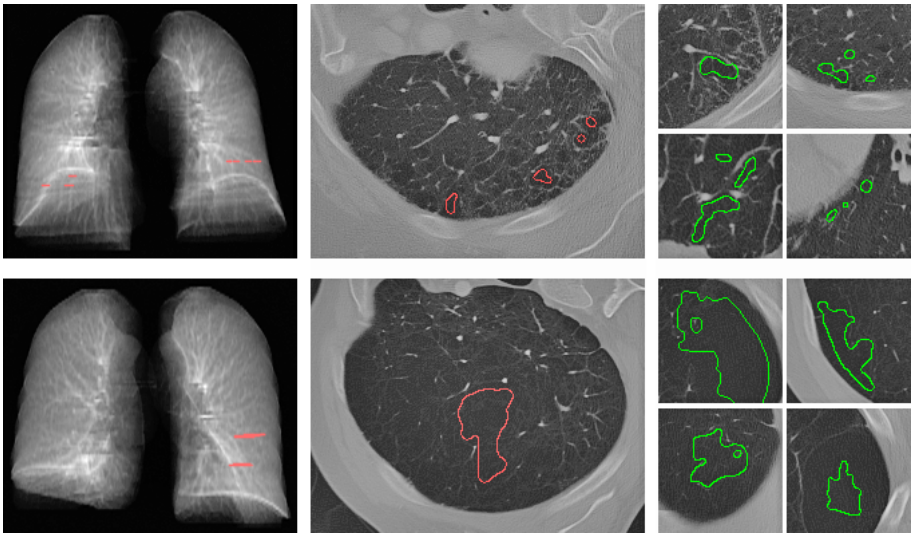


Figure 9: Retrieval ranking result of two distinct emphysemas with different tissue patterns (top: centrilobular emphysema, bottom: panlobular emphysema). The region highlighted in red on the left side shows the query region $\mathbf{R}_Q$ marked during search by a physician. On the right side, the green regions depict the four most similar regions $\mathbf{R}_{js}$ retrieved by our method [3].

3.3.6 Textual image description

In addition to *native* image features we use textual features collected from the web-pages or documents images are located in. These features are not the centre of this prototype, but we summarize their structure for completeness. Please refer to D1.2 [26] for a detailed discussion.

Web sites are crawled via distributed robots (today at a rate of $\sim$ 10000 pages/hour). A distributed crawling process is a challenge different from feature extraction and will not be addressed in this section. Gathered web pages are stored locally and pushed through a textual feature extraction workflow. Finally, the web page information is stored into a NoSql database (CouchDB [10]) and is ready for Lucene [8] indexation and deployment to a web application. The textual feature extraction workflow is a suite of Perl modules developed independently, covering several aspects presented below.

Html features Content is converted to UTF-8 and basic features are parsed:

- Title
- URL
- Host
- Content description
- Outward links

Meaningful content extraction On a web page, most of the content is irrelevant for indexing and a robust method had to be implemented to extract the meaningful parts from it. Several methods have been tested and rejected because of a lack of robustness and the difficulty to set

a threshold that works for most sites. Notably one adapted from [4] and a derivation of the PageRank algorithm [2]. We finally combined several ideas and extract text from the html tree in subgraph parts where the *fragmentation level* (the ratio between text length and number of children nodes) is the highest. Some of the implemented extraction methods are available on a web application at:

- interactive: http://znverdi.hcuge.ch/∼honbot/cgi-bin/url-measure.cgi
- batch: http://znverdi.hcuge.ch/∼honbot/cgi-bin/url-measure-extract-and-compare.cgi

Language detection Although it is often considered a solved problem, issues are raised when assigning a language to a web page's content. A *meta* tag with *Content-Language* information is rarely present. One page can contain several languages, either because of a publisher error or on purpose.

To address this issue, we have developed a Grails [39] web service used as a wrapper around 4 different publicly available methods:

- Language Detection [30]
- Langid.py [23]
- lc4j [34]
- Nutch [11]

To benchmark those methods, we used the EMEA corpus [41]. Each method has some strengths over the others and an optimal system could be built leveraging the training corpus and combining each output via an SVM or a decision tree. At the time being, we only use the Language Detection method [30].

Page relevancy score Even if a web site is mainly related to health topics, some pages (and sometimes most of them) are not relevant from the search engine's point of view. We have therefore implemented a metric to assign a score to each page, later used to boost the document score at search time.

Answering to the question *"is this page relevant for a health-oriented search engine?"* is ambiguous. We randomly drew 1000 web pages from our crawled sites and asked 7 people to anser *yes* or *no* to this question. Based on these collected data and consensus voting, we classified 83% of the pages in *keep* and *discard* categories.

We could then benchmark several classifiers. These classifiers were based on the *extracted meaningful content* previously described, taking into account several pieces of information:

- content length
- presence of MeSH terms
- presence of terms in the health-oriented EMEA [41] versus the general Europarl corpus [16]

Different classification algorithms were tested:

- Naive Bayes
- Support vector machines (SVM)
- Decision Tree

The most efficient classifier was build with the Naive Bayes method based on the EMEA/Europarl Corpus [16] . It was shown not to be robust enough to simply keep/discard web pages based on a score threshold, but such a score was an important improvement for the search results list order.

Custom page tagging This generic step consists in attaching virtually any type of textual information on a web page, based on external processing.

For example, it can be:

- terminology annotation based on keyword lists
- content tagging: for example *this page contains news feeds*, *this page has video*...
- MeSH term code (this process is under development)

Textual Features for Retrieval The standard Lucene approach for combining text attributes into searchable feature vectors is to encode them as separate Lucene *fields*, each attached to a *document*. A field is modeled as a feature vector $\mathbf{f}(x)$ of tokenized terms. At retrieval time, each field is compared against the input query using cosine similarity, normalized by the length of the field. This has the benefit of automatically boosting shorter fields, which are typically more informative. In the case of image retrieval, the smallest fields should be text like the HTML *alt* tag and the image file name.

After field similarity is computed, each corresponding document is scored by the sum of its fields' similarity values. A *boost value*, equal to the log-normalized value of the relevancy score as described above, is also applied. Documents are then returned in descending order of their scores.

So, for each field in a document (or image) $f \in d$, and each input query q, the similarity score is defined as:

$$similarity(q,f) = \sum_{t \in q \cap f} TF(t) \cdot IDF(t)^2 \cdot \frac{1}{||f||} \tag{23}$$

where $TF()$ and $IDF()$ are the commonly-used TF-IDF metrics, and $||f||$ is the Euclidian length normalization of the field.

When documents (or images) are retrieved, their score is defined as:

$$score(q,d) = coord(q,d) \cdot \log(boost(d)) \sum_{f \in d} similarity(q,f) \tag{24}$$

where $coord()$ is a coordination function to reward co-occurrence of terms, and $boost()$ is the relevancy score described above.

For more details, refer to the Lucene Java documentation for Similarity.java [9].

Future Development Currently our image retrieval system is in its alpha version. The system for processing textual documents retrieved from the web is well-developed, but the system for assigning relevant text (e.g. surrounding text, alt tags, and captions) to crawled images should be further elaborated, and the next months will be focused on this activity as well as on the relevancy ranking system. The exclusion of uninteresting and non-health-related images is a complex problem, which we need to work more closely on in order to identify only medical and health-related images. In the future we will extend our efforts in the web text domain to include a more specialized component for image crawling, annotation, and retrieval.

4 Prototype

The prototype consists of a modular framework that allows for the integration of feature extractors, the adaptation of extractors to specific image sets, and the generation of image descriptors. An exemplary set of features, used within KHRESMOI is integrated. A guide for the integration of new modules is given in App. A.

4.1 Introduction

The prototype framework consists of the actual descriptors, and an evaluation framework that allows for direct comparison of the features. The various texture extraction methods were tested. The pathology framework allows to easily plug-in and test features, classifiers, etc. For each processing step (computeFeatures, computePreSegmentation, computeResize, etc.) it defines a generic interface that will be used by the specific implementation. There is a central wrapper code that will decide, based on configurable settings, which implementation to execute.

4.2 Components

The framework contains the following components:

- `kSettings`: classes to configure the directory structure
- `PathologyFramework`: the main framework (see below)
- `various`: generic classes and functionality
- `medClasses`: generic functionality to load DICOM files, etc.
- `externals`: code pieces from external parties
- `testComputeFeatures.m`: script to run the wrapper on each image
- `testClassification.m`: script to test the classification
- `setpaths.m`: sets the MATLAB path for the entire framework
- `buildAll.m`: builds all mex files

Methods of the PathologyFramework

- `computeSegmentation`: segmentation algorithms for various anatomical regions
- `computeOverSegmentation`: algorithms to oversegment a volume (supervoxel, etc.)
- `computeResize`: algorithms to resize a volume
- `computeFeatures`: algorithms to extract features
- `computeDistance`: algorithms to compute distances

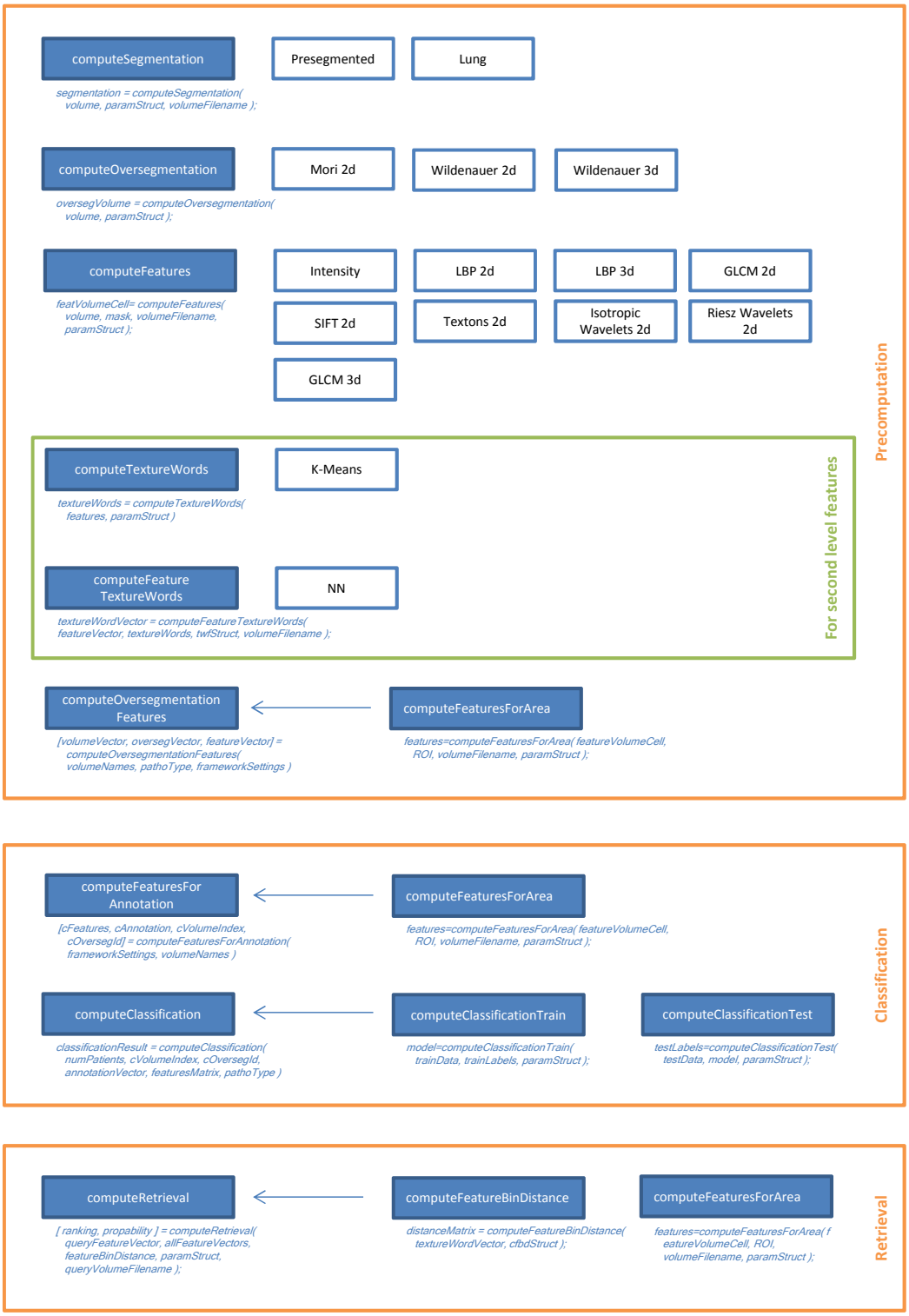


Figure 10: Overview of the pathology framework

- `computeTextureWords`: algorithms to compute a vocabulary for all volumes
- `computeTextureWordsFeatures`: algorithms to compute vocabulary features for a certain volume
- `computeClassificationFeatures`: algorithms to compute features for classification (histograms, etc.)
- `computeClassificationTrain`: implementations of classifiers to train texture
- `computeClassificationTest`: implementations of classifiers to test texture

4.3 Data Directory Structure

The directory structure where kSettings.basePath point to must have the following layout:

1. `datadir/`: all DICOM volume files, one in each sub-directory.
2. `annotation/`: all DICOM annotation files, one in each sub-directory.
3. `segmentation/`: all DICOM segmentation files, one in each sub-directory.
4. `moduledata/pathology/cache`: all computed results are stored in this directory using file names based on hash values to seperate different configurations.

4.4 Environment Setup and Configuration

This framework requires MATLAB 2011a or higher to run.

To configure the framework the following steps are required:

1. Running `setpaths.m` to set the MATLAB paths.
2. Running `buildAll.m` to compile the mex files.
3. Configuring `kSettings` to point to the correct image directories (see next chapter).

4.4.1 Configuring Directories

KSettings is the place for all MATLAB settings of the Khresmoi framework. Some settings may be machine or user dependent, therefore kSettings.m provides features such that parts of the configuration can be overwritten by a custom class derived from kSettings.

The framework is searching for a kSettings custom class in the following order:

1. Variable `host` and `user` are retrieved from the OS. If `kmyhostname.m` exists, host is overwritten by its result.
2. If the class `kSettings\_host\_user.m` exists, this becomes the `kSettings` custom class
3. else `kSettings\_host.m`
4. else `kSettings\_user.m`

The call order is

1. Call method of `preInitialize` of the kSettings custom class
2. Load all settings in `kSettings.m`
3. Call method of `postInitialize` of the kSettings custom class

4.5 Prototype download

The prototype can be downloaded for review at
`https://www.cir.meduniwien.ac.at/khresmoi/files/`
User: `khresmoiD22`, Password: `features2012`.

5 Prototype choices

The choices made during the prototype design were guided by the requirements of the modular retrieval framework, the characteristics of medical imaging data, and the associated information needs of radiologists.

Anatomy retrieval calls for descriptors, that are sufficiently specific to differentiate between anatomical structures in medical imaging data such as CT, or MRI. At the same time they have to cope with substantial natural variability of organ shape, and appearance in the population. In a large data retrieval situation, holistic features such as miniatures, or histograms of gradients proof to be a good choice.

Pathology retrieval is different from anatomy retrieval, since features have to capture subtle appearance variations that correlate with disease, but are only minor compared to the richness of appearance in the population. For this reason, it proofs necessary to adapt feature extractors, and image descriptors to specific anatomical regions, and corresponding candidate diseases. Features have to capture small scale appearance, and desciptors have to be learned to optimally reflect the appearance in a certain context, e.g., the lung.

2D image retrieval and modality recognition The situtation is similiar to anatomy retrieval, althought the variability in the data can be higher compared to medical imaging data, while at the same time the task of modality recognition can rely on coarser features compared to the identification of anatomy. Again features have to capture global image characteristics, while being invariant to local configurations (e.g., if a sketch is on the left, or the right side of an image). Similar to anatomy retrieval histogram based approaches show promising results in this context.

Modularity is a prime objective of the prototype, since we expect new features to be added during the project, or during deployment of the prototype. The key is that a bank of features is available during application, to allow for either manual, or ultimately automatic choice of features, and generation of descriptors, based on low level feature extractors. The prototype framework is a modular framework, that allows the extraction of primary-, and secondary features, image descriptors, and the learning of adaptive descriptors based on training data. Additional features can be implemented in a straight-forward fashion. The corresponding interfaces are defined in this document.

An exhaustive quantitative evaluation of the feature extractors, and image descriptors can be found in *deliverable D2.3*.

6 Conclusion

This deliverable contains a prototype framework for image feature extraction, and image description, and the corresponding documentation of the relevant methodology, and tasks within KHRESMOI.

The prototype is a modular framework that allows for feature extraction in images, and the generation of image descriptors to be used during retrieval. Three method families are explained: (1) the extraction of primary features that capture local image content, and are typically standard features such as wavelet coefficients, or gray level coocurrence matrices, (2) secondary features, that summarize the statistical properties of primary features extracted in a local neigh-

borhood, or region, and (3) image descriptors, that either aggregate local features to an image descriptor, or are calculated in a holistic fashion from the image.

In the context of medical image retrieval, learning of features is crucial, since the variability observed across even the healthy population, exceeds the subtle variations of local appearance correlating with specific pathologies. Successful retrieval of cases with relevant pathologies, thus relies on learning of anatomy specific feature vocabularies, that reflect the variability relevant for pathology description with high specificity.

Challenges, such as the efficient indexing of large scale data, and fast retrieval remain in the context of medical images. Since indices have to be region specific, improvements on the efficient building of indices, and a corresponding structure, that allows for an optimal representation and use of descriptors at different image scales remain. We expect that user tests will show the most important variants that are necessary in the retrieval system. A second challenge is that for fast retrieval, pre-calculation of parts of the processing chain can be necessary. Since the query depends on manually indicated regions of interest, that likely don't repeat, this is not trivial. Approaches from machine learning will be explored to organize and structure that entire data set, by combining weak labels on a case by case level, with locally extracted image features. The aim is to find sufficient structure in the data, in an unsupervise,d or semi-supervised manner, to be able to constrain query results, or to effectively combine textual and image queries.

References

[1] M.Sheerin Banu and Krishnan Nallaperumal. Analysis of color feature extraction techniques for pathology image retrieval system. IEEE, 2010.

[2] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the seventh international conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[3] Andreas Burner, Rene Donner, Marius Mayerhoefer, Markus Holzer, Franz Kainberger, and Georg Langs. Texture bags: Anomaly retrieval in medical images based on local 3d-texture similarity. *Workshop on Medical Content-based Retrieval for Clinical Decision Support at MICCAI 2011*, September 2011.

[4] Alex J. Champandard. The easy way to extract useful text from arbitrary html, April 2007.

[5] Adrien Depeursinge, Antonio Foncubierta-Rodríguez, Dimitri Van De Ville, and Henning Müller. Lung texture classification using locally—oriented riesz components. In Gabor Fichtinger, Anne Martel, and Terry Peters, editors, *Medical Image Computing and Computer Assisted Intervention —- MICCAI 2011*, volume 6893 of *Lecture Notes in Computer Science*, pages 231–238. Springer Berlin / Heidelberg, September 2011.

[6] 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 *Proc. MICCAI Workshop on Medical Content-based Retrieval for Clinical Decision Support*, 2011.

[7] René Donner, Georg Langs, Branislav Micusik, and Horst Bischof. Generalized Sparse MRF Appearance Models. *Image and Vision Computing*, 28(6):1031 – 1038, 2010.

[8] Apache Software Foundation. Lucene, 2011.

[9] Apache Software Foundation. Lucene similarity, 2011.

[10] Apache Software Foundation. Couchdb, 2012.

[11] Apache Software Foundation. Nutch, 2012.

[12] Leibe B Grauman, K. *Visual Object Recognition*. 2011.

[13] R. M. Haralick, Dinstein, and K. Shanmugam. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3:610–621, 1973.

[14] 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.

[15] Bela Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290(5802):91–97, March 1981.

[16] Philipp Koehn. Europarl: A multilingual corpus for evaluation of machine translation. Draft, 2002.

[17] 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.

[18] T. Leung and J. Malik. Recognizing surfaces using three-dimensional textons. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, pages 1010–, Washington, DC, USA, 1999. IEEE Computer Society.

[19] H. Ling and Okada K. Diffusion distance for histogram comparison. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1*, volume 1, pages 246–253, 2006.

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

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

[22] H J Lowe, I Antipov, W Hersh, and C A Smith. Towards knowledge-based retrieval of medical images. the role of semantic indexing, image content representation and knowledge-based retrieval. *Proc AMIA Symp*, pages 882–6, 1998.

[23] Marco Lui. Langid.py - language identifier, 2012.

[24] James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

[25] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *Int. J. Comput. Vision*, 43:7–27, June 2001.

[26] Niraj Aswani Mark A. Greenwood, Angus Roberts and Phil Gooch. Initial prototype for semantic annotation of the khresmoi literature. *Khresmoi project public deliverable*, 2012.

[27] Dimitrios Markonis, Ivan Eggel, Alba G.Seco de Herrera, and Henning Müller. The medGIFT group in ImageCLEFmed 2011. In *Working Notes of CLEF 2011*, 2011.

[28] Dimitrios Markonis, Alba Garcia Seco de Herrero, Ivan Eggel, and Henning Müller. Multi–scale visual words for hierarchical medical image categorisation. In *SPIE medical imaging: Advanced PACS-based Imaging Informatics and Therapeutic Application*, February 2012.

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

[30] Shuyo Nakatani. Language detection library for java, 2011.

[31] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2 of *CVPR*, pages 2161–2168, June 2006.

[32] T Ojala, M Pietikainen, and D Harwood. *Performance evaluation of texture measures with classification based on Kullback discrimination of distributions*, volume 1, page 582585. IEEE, 1994.

[33] 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.

[34] Marco Olivo. lc4j, a language categorization java library, 2011.

[35] Florent Perronnin and Chris Dance. Fisher kernels on visual vocabularies for image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[36] Laura Sevilla and Erik Learned-Miller. Distribution Fields. Technical Report UM-CS-2011-027, Dept. of Computer Science, University of Massachusetts Amherst, 2011.

[37] Gaurav Sharma and H. Joel Trussell. Digital color imaging. *IEEE Transactions on Image Processing*, 6(7):901–932, 1997.

[38] J. Sivic and A. 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*, volume 2, pages 1470–1477. IEEE Computer Society, 2003.

[39] SpringSource. Grails, 2011.

[40] Elias M. Stein and Guido Weiss. *Introduction to Fourier Analysis on Euclidean Spaces.* Princeton University Press, November 1971.

[41] Jörg Tiedemann. News from OPUS - A collection of multilingual parallel corpora with tools and interfaces. In N. Nicolov, K. Bontcheva, G. Angelova, and R. Mitkov, editors, *Recent Advances in Natural Language Processing*, volume V, pages 237–248. John Benjamins, Amsterdam/Philadelphia, Borovets, Bulgaria, 2009.

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

[43] Mihran Tuceryan and Anil K. Jain. Handbook of pattern recognition & computer vision. chapter Texture Analysis, pages 207–248. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2nd edition, 1998.

[44] Michael Unser and Dimitri Van De Ville. Wavelet steerability and the higher–order Riesz transform. *IEEE Transactions on Image Processing*, 19(3):636–652, March 2010.

[45] Koen E. A. van de Sande, Theo Gevers, and Arnold W. M. Smeulders. The university of amsterdam's concept detection system at imageclef 2009. *Lecture Notes in Computer Science*, 6242/2010:261–268, 2010.

[46] 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.

[47] H. Wildenauer, B. Micusk, and M. Vincze. Efficient texture representation using multi-scale regions. In *Proceedings of the 8th Asian conference on Computer vision - Volume Part I*, pages 65–74. Springer-Verlag, 2007.

[48] Jianguo Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. In *Proceedings of Conference on Computer Vision and Pattern Recognition Workshop (CVPRW 2006)*, page 13, June 2006.

A Contributing to the Pathology Framework

A.1 Implement a Texture Descriptor

The class `PathologyFramework/computeFeatures/computeFeaturesWrapper.m` contains the wrapper code that will decide which feature computation will be computed, based on method parameter of `kSettings.pathologyFeatureParameters`.

To implement a new texture descriptor follow this guideline:

1. Create a new directory under `PathologyFramework/computeFeatures/` with a unique name of your texture descriptor. Please include in the name whether it is a 2d or 3d texture descriptor. The naming convention is

   `<Algorithm>_<2d|3d>`

2. Implement the interface for texture-feature calculation, which is defined in

   `PathologyFramework/computeFeatures/computeFeaturesInterface.m`

   by creating a new MATLAB file in the created directory. Please follow the naming conventions

   `computeFeatures_<Algorithm>_<2d|3d>.m`

```
featVolume = computeFeatures( volume, paramStruct, volumeFilename )

% The function computeFeature returns feature-descriptors for a given
% volume.
%
% Parameters:
%   volume          a matrix containing the image data.
%   paramStruct     a structure that contains the parameters for the
%                   computeFeature function. paramStruct.method is
%                   mandatory, so that the wrapper executes the given
%                   implementation. These parameters may be defined in
%                   kSettings.m.
%   volumeFilename  a String containing the name of the volume. The
%                   purpose of this parameter is that the
%                   implementation may load additional data for a given
%                   volume.
%
% Returns:
%   featVolume      A cell of matrixes ndims(volume)+1, whereas the last
%                   dimension contains the features per pixel/voxel.
%
%                   For example, the LBP implementation will return for
%                   a 512x512x100 volume a cell with 4 elements, repre-
%                   senting the various scales. Each cell element is of
%                   dimension 512x512x100x28.
```

3. Configure your algorithm and the parameters that are required. Use your personal kSettings file to do so. The following section shows an example configuration of the LBP3D algorithm. The parameter `method` is mandatory for all implementations, as the wrapper will decide, based on it, which feature descriptor to call:

```
pathologyFeatureParameters = struct( ...
    'method',               'lbp3d', ...
    'implementation',       'lbp_mex', ... % 'LBP_MEX', 'LBP_MATLAB'
    'scale',                [1 2 3 4], ...
    'scaleInterpolation',   'linear', ...
```

4. Running the wrapper by executing

```
features=computeFeaturesWrapper.computeFeatures( volume, volumeName );
```

   will call the method

```
PathologyFramework/computeFeatures/lbp3d/computeFeatures\_lbp3d.computeFeatures( ...
    volume, paramStruct, volumeFilename )
```

   whereas the `paramStruct` parameter is the `pathologyFeatureParameters` struct of `kSettings` (defined in 3.), and therefore may contain implementation specific settings.

   This wrapper code also caches the results, so the method `computeFeatures` will not be called if features have been computed before.