SIA: Semantic Image Annotation using Ontologies and Image Content Analysis

Pyrros Koletsis
Dissertation thesis

Technical University of Crete (TUC)
Department of Electronics and Computer Engineering

Committee
Euripides Petrakis, Associate Professor (Supervisor)
Stavros Christodoulakis, Professor
Chrisa Tsinaraki, Visiting Lecturer

Chania 2009

Abstract

Image annotation is the task of assigning a class name or description to an unknown image. In this work, we propose SIA, a framework capable of automatically annotating images using information from ontologies in combination with low level image features (color and texture) which are extracted from raw image data. The method works for images of a particular domain. First, an ontology is constructed denoting characteristics of the various image classes in this domain. A set of low level image characteristics is also assigned to each class. Image annotation is then implemented as a retrieval process by comparing vectors of such low-level characteristics extracted from the input image and representative images of each class in the ontology respectively. A combined similarity measure is used between images. The relative importance of low-level features in this measure is determined using machine learning by decision trees. The result list of images are ranked in decreasing visual similarity. `AVR`*(Average Retrieval Rank) is used as a metric to estimate the semantic category where the image is possible to belong to (ie. the unknown image is assigned a class which is computed by voting among the top ranked retrieved images from the ontology). The experimental results demonstrate that approximately 70% of the input images are correctly annotated (ie. the method identified its class correctly). Experiments and evaluations were realized on an image dataset consisting of images belong to 30 dog breeds (semantic categories), which were collected from the World Wide Web (*`WWW`*).*

Contents

1 Introduction

1.1 Motivation

Recently, content-based image retrieval (CBIR) has received much interest due to the remarkable increase of audiovisual information in digital libraries and the World Wide Web. A typical problem in image information systems often occurs when an end-user is given with one or more images and is asked to assign a description to each one.

Manual annotation can provide rich image descriptions, however it is time consuming and thus expensive. On the other hand, annotation based on automatic feature extraction is relatively fast and cheap, however similarity between image features (extracted by image analysis) does not always correspond to semantic similarity as perceived by humans. This is referred to as the "semantic gap" problem [1]. The semantic gap is the lack of coincidence between the information that one can extract from the visual data and the interpreting that the same data have for a user in a given situation. Associating low-level features with semantic meanings is a possibility. It is still inaccurate how to associate semantic concepts with visual features effectively and efficiently.

1.2 Image Annotation

In general, the semantics consist of two parts that describe different aspects of visual data: one part contains the feature descriptions for the image itself (content semantics), and the other comprises of content descriptions from the human conceptual aspect (concept semantics). In this thesis an image ontology[1] is constructed semiautomatically. Semantic annotations corresponding to the image categories represented, are associated with the ontology image classes manually. These representations are enhanced by color and texture measurements which are extracted from images of these categories using image analysis.

Image annotation is viewed as an image classification problem: The system is provided with an unknown image and the problem relates to assigning a class name to it, that is mapping the unknown image to one of a number of known classes. The image then inherits the class properties and annotation of its assigned class.

Undoubtedly the number of classes can be huge. This is mostly due to the advent of Internet where a large number of images on every conceivable topic are available. In this

[1] Ontologies are consensual and formal specifications of conceptualizations. They provide shared understanding of a domain for communication across people and systems

work, the number of classes is limited to the classes of a specific application domain. More specifically, the deal with images of dog breeds (semantic classes).

1.3 Methodology: Main Idea

In this thesis we propose SIA (Semantic Image Annotation): an intelligent framework for image annotation using ontologies, by combining the analysis of visual content and the manually performed description of image data. High level descriptions and low-level information are efficiently stored in an ontology model providing formal descriptions. Low-level features have their implementation value enabling the ontology being an annotation or content-based image retrieval system. The problem of image annotation is treated as a retrieval problem which is facilitated by a) a weighting scheme on descriptors witch improves the retrieval performance, and b) a voting scheme for computing the semantic category of an unknown image from the categories of images in the retrieved set.

SIA works in four steps with the first two being part of the system's manufacture and the rest two being part of the actual annotation process of an unknown image.

Building the Ontology : First step is to build an ontology model for keeping all the necessary information about the images in the database. This includes both low-level features and concept descriptions. In this work, such concept descriptions are obtained from WordNet[2].

Image Similarity : An overall similarity measure is calculated between images. The relative importance of each low-level feature(color,texture) in this measure is determined using machine learning by decision trees. Weights are ranged between [0-1]. The combined similarity is computed as a weighted sum of similarities in each low-level feature's space.

Image Retrieval : Given an unknown image, the ontology is searched to retrieve the images most similar to it. Image matching is implemented using image content descriptions (color, texture and hybrid features). The combined similarity measure calculated above is used for image similarity. Finally the result images are ranked in decreasing order of similarity.

Image Annotation : The unknown image is classified into one of known semantic categories. The semantic image category with more instances in the retrieved set is chosen. Finally its description is assigned to the unknown image.

1.4 Structure of this thesis

The rest of this thesis is organized as followed:

Chapter 2 presents a survey of research related. Issues related to ontologies, image annotation and image content analysis are discussed.

Chapter 3 presents the key concepts underlying SIA, our methodology for the design of an ontological image annotation system.

Chapter 4 presents experimental setup including the data set and issues related to the evaluation methodology that has been followed are discussed. Results in the evaluation of the proposed methodology for the annotation of unknown images using an ontology for 30 dog breeds is presented.

Chapter 5 summarizes the main achievements of this thesis, discusses results obtained, and provides suggestions for further work.

2 Related Work

2.1 Image Annotation as a Retrieval Problem on Ontology Information

In dealing with the problem of the semantic gap, researches are trying to arrange low-level features to semantic meaningful categories (keywords) [3]. Ontologies are used to maintain keywords along with high-level information for semantic text retrieval purposes. It is widely accepted that the retrieval of images annotated with keywords may provide potentially better results. The quality of the retrieved results depends on the amount, quality, and consistency of the metadata associated with each image [4, 5].

Besides associating features to keywords, another important source of information is the relationship between semantic labels, often referred to as semantic ontology. Usually these ontologies provide a multi-layer tree structure hierarchy description of contents. This enables machines to identify the low-level feature descriptions for human conceptual items through the keywords given by users [6, 7].

Closely to our approach, recent attempts are based on the idea of image annotation using ontologies. High-level concepts are efficiently stored and automatically mapped to visual features or objects which are extracted by various image analysis techniques [8, 9]. Recent efforts are trying to arrange visual features to semi-concepts values. Image annotation in this case is based on semantic inference rules [10].

2.2 Image Content Analysis

The appropriate selection of low-level features could be a significant step in the construction process of an image annotation system. In this thesis MPEG-7 general visual descriptors (color ,texture) are used together with Tamura texture and Hybrid descriptors from LIRE [11]. A brief description of each descriptor is given below.

2.2.1 Color Features

Color is one of the most recognizable elements of image content and is the most commonly used feature in image retrieval because of its invariance with respect to image scaling, translation and rotation. Color features are independent of image size and orientation and can be used for describing content in still images and video.

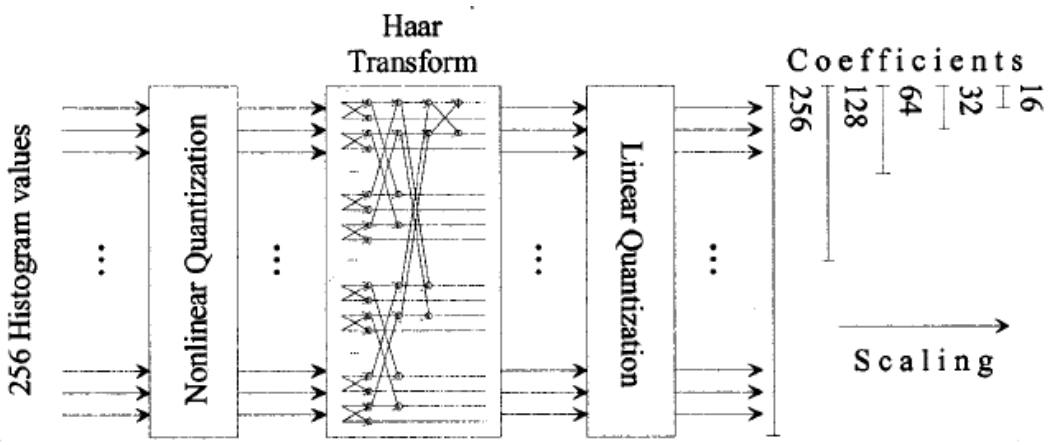

Figure 2.1: A Schematic diagram of SCD generation.

The MPEG-7 standard defines different color global descriptors , more details can be find in reference papers [12, 13] or in implemented systems [14, 15]. In this work the following descriptors have been considered.

▷ **Scalable Color Descriptor (SCD) :**

This descriptor is mainly a color histogram in the HSV color space encoded by a Haar transformation. It is used to find and compare images by their color characteristics. Using the Haar transform allows to sort the histogram bins by importance.

A uniform quantification is applied on the HSV image where *Hue* is splitted in 16 bins and where *Saturation* and *Value* have both 4 bins. Then the 256 histogram values are mapped into a "nonlinear" 4-bit representation, giving higher significance to the small values with higher probability. Then the Haar transform is performed to remove the "useless" information. Thus, the first bins contain the fundamental information and it is possible to scale the number of coefficients from 256 until 16. Figure 2.1 shows the block diagram of the SCD extraction process.

The default matching function for Scalable Color is based on the L1 metric. Eq. 2.1.

$$D_{\mathrm{SC}} = \sum_{i=1}^{N} |\mathbf{H}_{\mathrm{A}}[i] - \mathbf{H}_{\mathrm{B}}[i]|. \tag{2.1}$$

In principle, any other matching method suitable for histograms can be used, although it was found that L1 metric gave very good retrieval performance in the MPEG-7 core experiments.

Component	Subspace	Number of quantization levels for different numbers of histogramm bins			
		256	128	64	32
Hue	0	1	1	1	1
	1	4	4	4	4
	2	16	8		
	3	16	8	8	4
	4				
Sum	0	32	16	8	8
	1	8	4	4	4
	2	4			
	3	4	4	2	1
	4			1	

Figure 2.2: HMMD color space quantization for CSD.

▷ **Color Structure Descriptor (`CSD`) :**

The `CSD` is a color feature descriptor that captures both color content (similar to a color histogram) and information about the structure of this content(position of color).

The `CSD` works on a special version of the `HMMD` color space defined by a non-uniform color space quantification. First, the `HMMD` color space is divided into five subspaces. This division is performed according to the DIFF value where subspaces 0, 1, 2, 3, and 4 correspond respectively to DIFF intervals [0, 6], [6, 20], [20, 60], [60,110], and [110, 255]. Then a pixel is quantized along the Hue and Sum axes according to its subspace (see Figure 2.2).

Once the quantization step is done, the `CSD` is computed by visiting all locations in the image. At each location, the color C_m (with $m \in [0, M-1]$) of all the pixels contained in the 8x8 structuring element overlaid are retrieved. The `CSD` bins are incremented according to these colors. In other words, even if 14 of the 64 pixels are defined by color C_1 in the structuring element of Figure 2.3, the bin $C1$ is only incremented of one. This structuring element is exploited to avoid the lost of structure with typical histograms.

The number of structuring points is always 64 and the distance between them increases with the image size (Figure 2.4).

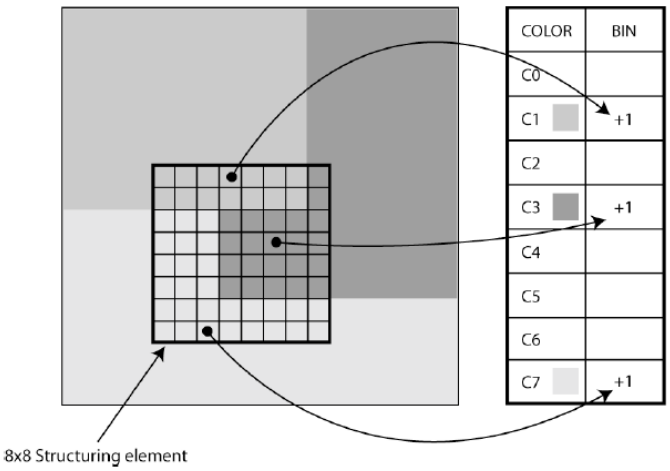

Figure 2.3: Histogram accumulation at one location in the image.

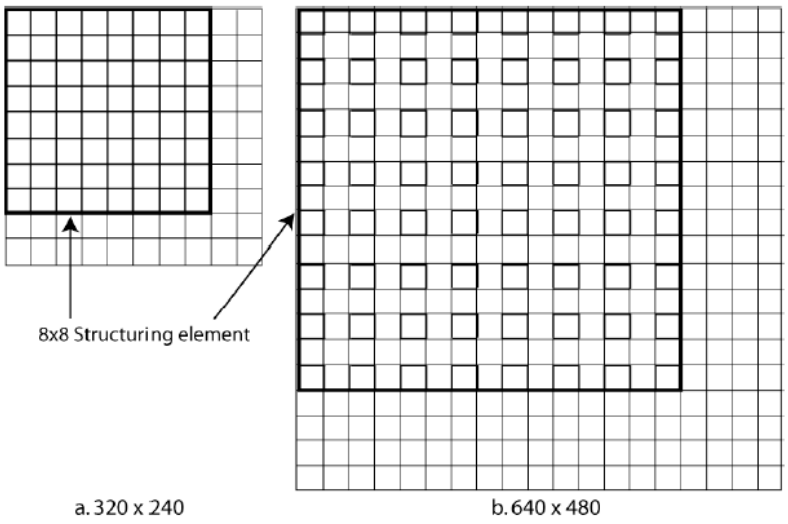

Figure 2.4: 64 structuring points for 2 structuring element sizes.

Once the `CSD` histogram is computed, a non-linear quantization step is performed to obtain a 8-bits coding for each bins. Since the structure of the descriptor is the same as for color histogram, the same matching functions can be used. The default matching function is the L1 metric, as in the case of Scalable Color.

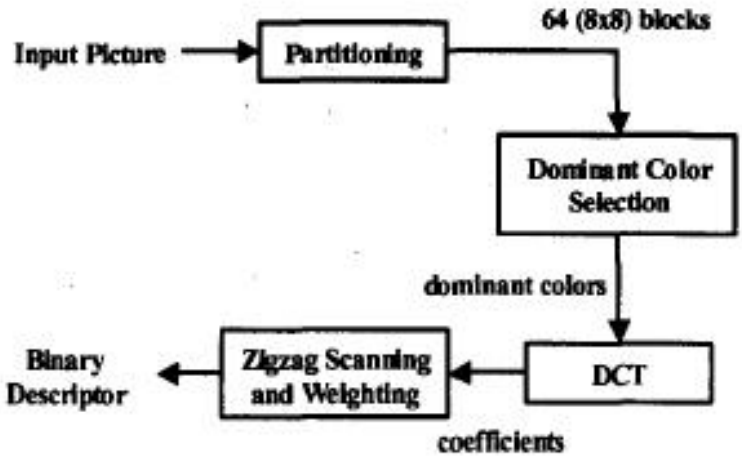

Figure 2.5: A Schematic diagram of CLD generation.

▷ **Color Layout Descriptor (CLD) :**

The CLD descriptor captures the spatial layout of the representative colors on a region or image. Representation is based on coefficients of the Discrete Cosine Transform. This is a very compact descriptor being highly efficient in fast browsing and search applications. It provides image-to-image matching as well as ultra high-speed sequence-to-sequence matching.

The Color Layout uses an array of representative colors for the image, expressed in the YCbCr color space, as the starting point for the descriptor definition. The size of the array is fixed to 8x8 elements to ensure scale invariance of the descriptor. The array obtained in this way is then transformed using the Discrete Cosine Transform (DCT), which is followed by zig-zag re-ordering(Figure 2.5).

A representative color was chosen for each block by averaging the values of all the pixels in each block. This results in three 8x8 arrays, one for each color component. This step is directly visualized in the first window of Figure 2.6. Each 8x8 matrix was transformed to the YCbCr color space (second window of Figure 2.6). Next each 8x8 matrix was transformed by 8x8 DCT to obtain 3 8x8 DCT matrices of coefficients, one for each YCbCr component (third window of Figure 2.6). The CLD descriptor was formed by reading in zigzag order 6 coefficients from the Y-DCT-matrix and 3 coefficients from

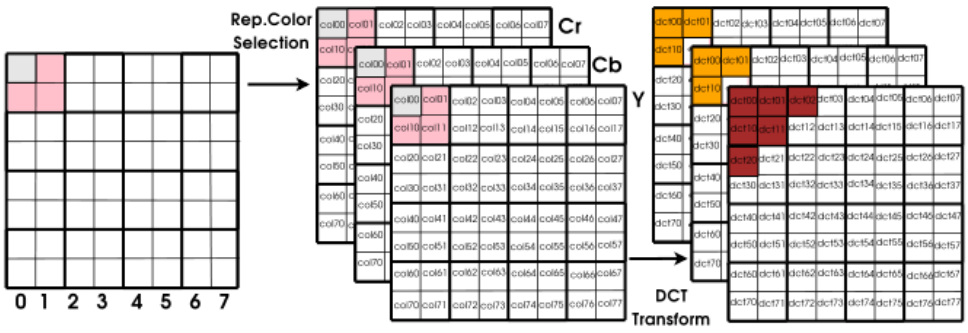

Figure 2.6: Stages of `CLD` computation.

each DCT matrix of the two chrominance components. The descriptor is saved as an array of 12 values.

The default matching function is essentially a weighted sum of squared differences between the corresponding descriptor components. Eq. 2.2

$$D_{\mathrm{CL}} = \sqrt{\sum_i w_i^y (Y_i - Y_i')^2} + \sqrt{\sum_i w_i^b (Cb_i - Cb_i')^2} + \sqrt{\sum_i w_i^r (Cr_i - Cr_i')^2}, \tag{2.2}$$

where Y , Cb and Cr are the DCT coefficients of the respective color components, w_i^y , w_i^r , w_i^b are weights chosen to reflect the perceptual importance of the coefficients and the summation is over the number of coefficients.

2.2.2 Texture Features

Color descriptors has shown good performance for descriminating images based on color. However, in many cases, color is usually insufficient for discriminating between images with the same color but different texture. For example, it is impossible to discriminate between the images a maltese and a husky dog using color features alone. Texture features are capable of recognizing repeated patterns in an image, analyzing the energy distribution in the frequency domain.

▷ **Edge Histograms Descriptor (`EHD`) :**

The edge histogram descriptor captures the spatial distribution of edges, somewhat in the same spirit as the CLD. Finally, the quality of reference papers [12, 13] and the number of implemented systems [14] have helped in our choice.

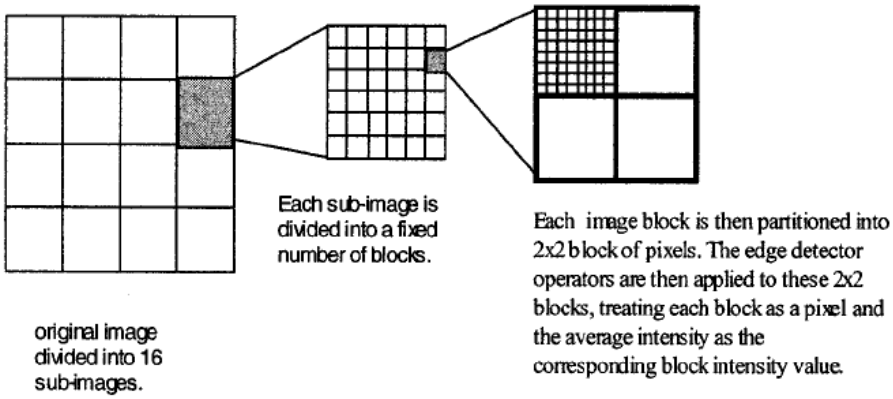

Figure 2.7: Subimages and macroblocks decompositions.

The edge histogram descriptor represents the spatial distribution of five types of edges, namely four directional edges and one non-directional edge. Edges play an important role for image perception. For example, natural images tend to have a non-uniform edge distribution, thus edges descriptors have been used to classify city versus landscape.

This descriptor is implemented as follows: Firstly, a gray-intensity image is divided in 4×4 sub-images. Each sub-image has its own local histogram with 5 bins. These 5 bins correspond to the 5 edge types: vertical, horizontal, 45°diagonal, 135°diagonal, and isotropic.

In order to fill the local histograms each sub-image is divided in macroblock. A macroblock is composed by 2×2 macropixels and is associated to an edge type. Figure 2.7 gives details on how the macroblock are built.

To associate a macroblock with an edge type, a convolution with 5 simple edge detectors (Eq. 2.3) is performed and the one with the strongest reply is linked to the macroblock.

$$\left|\begin{matrix} 1 & -1 \\ 1 & -1 \end{matrix}\right| \left|\begin{matrix} 1 & 1 \\ -1 & -1 \end{matrix}\right| \left|\begin{matrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{matrix}\right| \left|\begin{matrix} 0 & \sqrt{2} \\ \sqrt{2} & 0 \end{matrix}\right| \left|\begin{matrix} 2 & -2 \\ -2 & 2 \end{matrix}\right| \tag{2.3}$$

The local histogram is therefore built counting the result of each macroblock. Finally the global histogram is quantified in 3 bits per bin ($\texttt{EHD}(i) \in [0-7]$). Table 2.1 shows a typical output of the `EHD` where the 80 bins of the global histogram are divided in 4x4 local histograms.

Note that there are a total of 80 bins, 3 bits/bin, in the edge histogram. One can use the 3-bit number as an integer value directly and compute the L1 distance between two edge histograms.

5 1 6 6 5	4 4 7 6 4	1 7 5 7 3	1 7 5 4 2
6 1 4 6 6	4 4 7 6 4	4 3 5 6 6	6 3 7 4 3
3 3 6 7 5	4 3 5 7 6	2 5 5 7 5	1 5 6 4 5
4 3 6 7 3	4 3 3 7 5	2 5 3 7 5	2 5 4 6 6

Table 2.1: Typical output of `EHD` with 80 bins.

▷ **Tamura Features :**

Tamura et. al. [16] proposed texture features that correspond to human visual perception. They defined six textural features (coarseness, contrast, directionality, line-likeness, regularity and roughness) and compared them with psychological measurements for human subjects. The first three features described below attained very successful results and are used in our evaluation, both separately and as joint values.

Coarseness has a direct relationship to scale and repetition rates and was seen by Tamura as the most fundamental texture feature. An image will contain textures at several scales coarseness aims to identify the largest size at which a texture exists, even where a smaller micro texture exists. Computationally one first takes averages at every point over neighborhoods the linear size of which are powers of 2. The average over the neighborhood of size $2^k \times 2^k$ at the point (x, y) is Eq. 2.4

$$A_k(x,y) = \sum_{i=x-2^{k-1}}^{x+2^{k-1}-1} \sum_{j=y-2^{k-1}}^{y+2^{k-1}-1} f(i,j)/2^{2k} \quad . \tag{2.4}$$

Then at each point one takes differences between pairs of averages corresponding to non-overlapping neighborhoods on opposite sides of the point in both horizontal and vertical orientations. In the horizontal case this is Eq. 2.5

$$E_{k,h}(x,y) = |A_k(x+2^{k-1},y) - A_k(x-2^{k-1},y)| \tag{2.5}$$

At each point, one then picks the best size which gives the highest output value, where k maximizes E in either direction. The coarseness measure is then the average of $S_{opt}(x,y) = 2^{k_{opt}}$

Contrast aims to capture the dynamic range of grey levels in an image, together with the polarization of the distribution of black and white. The first is measured using the standard deviation of grey levels and the second the kurtosis a_4. The contrast measure is therefore defined as Eq. 2.6

$$F_{con} = \sigma/(\alpha_4)^n \quad \text{where} \quad \alpha_4 = \mu_4/\sigma^4 \tag{2.6}$$

Experimentally, Tamura found n = 1/4 to give the closest agreement to human measurements. This is the value we used in our experiments.

Directionality is a global property over a region. The feature described does not aim to differentiate between different orientations or patterns, but measures the total degree of directionality. Two simple masks are used to detect edges in the image. At each pixel the angle and magnitude are calculated. A histogram, H_d , of edge probabilities is then built up by counting all points with magnitude greater than a threshold and quantizing by the edge angle. The histogram will reflect the degree of directionality. To extract a measure from H_d the sharpness of the peaks are computed from their second moments.

Finally distances between images vectors were calculated upon feature vectors using the Manhattan metric.

2.2.3 Hybrid Features

Hybrid descriptors can be formulated by incorporating several low-level features (color,texture) to a new descriptor. We will describe two new low-level features which contain both color and texture information. These descriptors are both part of LIRE [11].

▷ **Color and Edge Directivity Descriptor(`CEDD`) :**

`CEDD` is a new low level feature that combines, in one histogram, color and texture information. `CEDD` size is limited to 54 bytes per image, rendering this descriptor suitable for use in large image databases.

First, the image is devided in a fixed number of blocks (eg 3x3). In order to extract the color information, a set of fuzzy rules undertake the extraction of a Fuzzy-Linking histogram. This histogram stems from the HSV color space. Twenty rules are applied to a three-input fuzzy system(one for each HSV) in order to generate eventually a 10-bin quantized histogram. Each bin corresponds to a preset color. The number of blocks assigned to each bin is stored in a feature vector. Then, 4 extra rules are applied to a two input fuzzy system, in order to change the 10- bins histogram into 24-bins histogram, importing thus information related to the hue of each color that is presented

Next, the 5 digital filters that were proposed in the MPEG-7 Edge Histogram Descriptor (see 2.2.2) are also used for exporting the information which is related to the texture of the image, classifying each image block in one or more of the 6 texture regions that has been fixed, shaping thus the 144 bins histogram.

With the use of the Gustafson Kessel fuzzy classifier 8 regions are shaped, which are then used in order to quantize the values of the 144 CEDD factors in the interval 0-7,

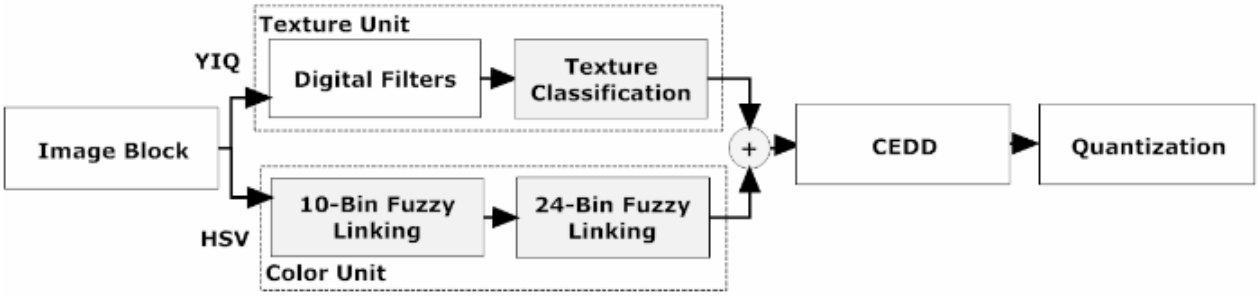

Figure 2.8: A Schematic diagram of CEDD generation.

limiting thus the length of the descriptor in 432 bits You can see the schematic diagram of CEDD in Figure 2.8

For the measurement of the distance of CEDD between images, Tanimoto coefficient is used. Eq. 2.7

$$T_{ij} = t(x_i, x_j) = \frac{x_i^T x_j}{x_i^T x_i + x_j^T x_j - x_i^T x_j} \tag{2.7}$$

Where x^T is the transpose vector of x. In the absolute congruence of the vectors the Tanimoto coefficient takes the value 1, while in the maximum deviation the coefficient tends to zero. Additional information about the descriptor can be found at [17].

▷ **Fuzzy Color and Texture Histogram(FCTH) :**

This feature is named FCTH -Fuzzy Color and Texture Histogram - and results from the combination of 3 fuzzy systems. FCTH size is limited to 72 bytes per image. FCTH works exactly the same with CEDD with a little difference in texture information extraction.

Initially the image is segmented in a present number of blocks. Next extracts the same color information as CEDD.

For the extraction of texture information each image block is transformed with Haar Wavelet transform and a set of texture elements are exported. These elements are used as inputs in a third fuzzy system which converts the 24-bins histogram in a 192-bins histogram, importing texture information in the proposed feature.Eight rules are applied in a three-input fuzzy system.

For the quantization process Gustafson Kessel fuzzy classifiers are used. You can see the schematic diagram of FCTH in Figure 2.9

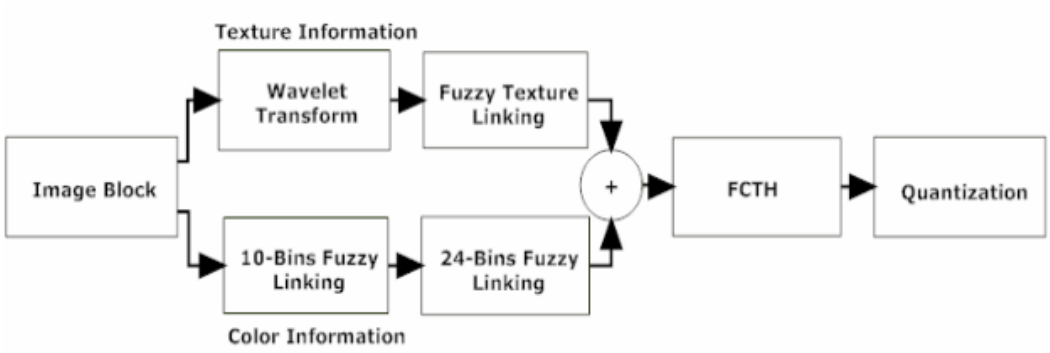

Figure 2.9: A Schematic diagram of `FCTH` generation.

For the measurement of the distance of FCTH between the images, Tanimoto coefficient is used. Eq. 2.7. Additional information about the descriptor can be found at [18].

3 Proposed Method

In this chapter we present SIA, our proposed intelligent framework for image annotation. Ontologies provide a formal way of successfully capturing both semantic and low level knowledge that this is necessary for representing the image class particular to the application domain at hand and their relationships. The architecture of the proposed system consists of several modules, the most important of them being.

a) Ontology Construction
b) Image Content Analysis
c) Image Similarity Computation
d) Image Annotation

The first two being part of system's manufacture and the rest two being part of the annotation process.

3.1 Ontology Construction

In this section, the construction of an ontology model for providing shared semantic interpretation of image contents is presented. The image ontology has two main components namely, the class hierarchy of a domain and the textual description of this domain. The textual, is further divided into low-level descriptions, namely the text description and visual text description (what people see).

3.1.1 Class Hierarchy

The class hierarchy of the image domain is generated based on a formal class hierarchy depending on following nouns hierarchy of Wordnet [2] of the domain at hand. Thus information concerning domain's relationships and classifications is provided. Another reason of using WordNet is that it is one of the largest conceptual hierarchies available but our approach can easily extend to other domains.

In this work a class hierarchy under the domain of dog breeds is constructed. The dog breed taxonomy is generated following the nouns hierarchy of Wordnet (ie. dog, working group, Alsatian). The leaf classes in the hierarchy represent the different semantic categories depicted to all images in our database (ie. actual dog breeds). Each image of a certain breed is represented as an instance of a leaf class. For example leaf class Labrador has 6 instances as many as the images of Labrador in our database (see Figure 3.1 on the left side). The image is connected with its corresponding instance in the ontology by an object property called *hasImage*. The domain of the last property is a

list of instances of the leaf classes in the hierarchy, and range, a list of URI's values of all images in our database.

3.1.2 Descriptions

Image descriptions, are divided into three layers. corresponding to text description, visual text description and finally low-level description respectively. As it is obvious from their names, they are made to describe different aspects of image content.

▷ **Text Description :**

In this part, the high-level narrative information of dog descriptions from external information source is collected and encapsulated into classes and instances. The Wordnet annotation and the information about habitat are represented as instances of the '*wordnet*' and '*habitat*' classes respectively. These classes are subclasses of text description class. For example, in class '*wordnet*', instances are like '*breed originally from Labrador having a short black or golden-brown coat*', and in '*habitat*' class, instances are like '*indoor,outdoor*'. (see Figure 3.1 on the right and center side)

▷ **Visual Text Description :**

In this part, emphasis is given to descriptions that human would give to images (what people see). These descriptions are following a specific hierarchy based on the topic described. '*Size*' and '*shape*' are classes that were generated describing the size and the shape of the dog. As shown in Figure 3.1, instances of these classes are '*small*','*medium*' and *'regular'* ,'*chaby'* respectively.

▷ **Low level Description :**

Low-level features are assigned to classes, subclasses and instances. A low-level description class has three subclasses, each one of them corresponding to different types of low level features (color,texture,hybrid). Continuing the abstraction process, the three previously generated classes contain subclasses, which are the descriptors presented in chapter 2 respectively (e.g. EHD,CEDD,CSD). Low level descriptors extracted from images, are represented as instances of the leaf classes (actual low-level features).

Low-level descriptors presented in 2 can have different format (vector ,signatures) for the low-level representation of image. Lux et. al. [11] by using Lucene index transformed signatures or vectors extracted by the features implementations into text. This contributed in the final representation of instances of low-level features in the ontology (ie. strings). Once low-level features are extracted from instances (images) of semantic categories, automatically were stored in the corresponding classes and at the same time were connected to previously mentioned semantic instances through an object property (ie. hasColorLayout , hasCEDD).

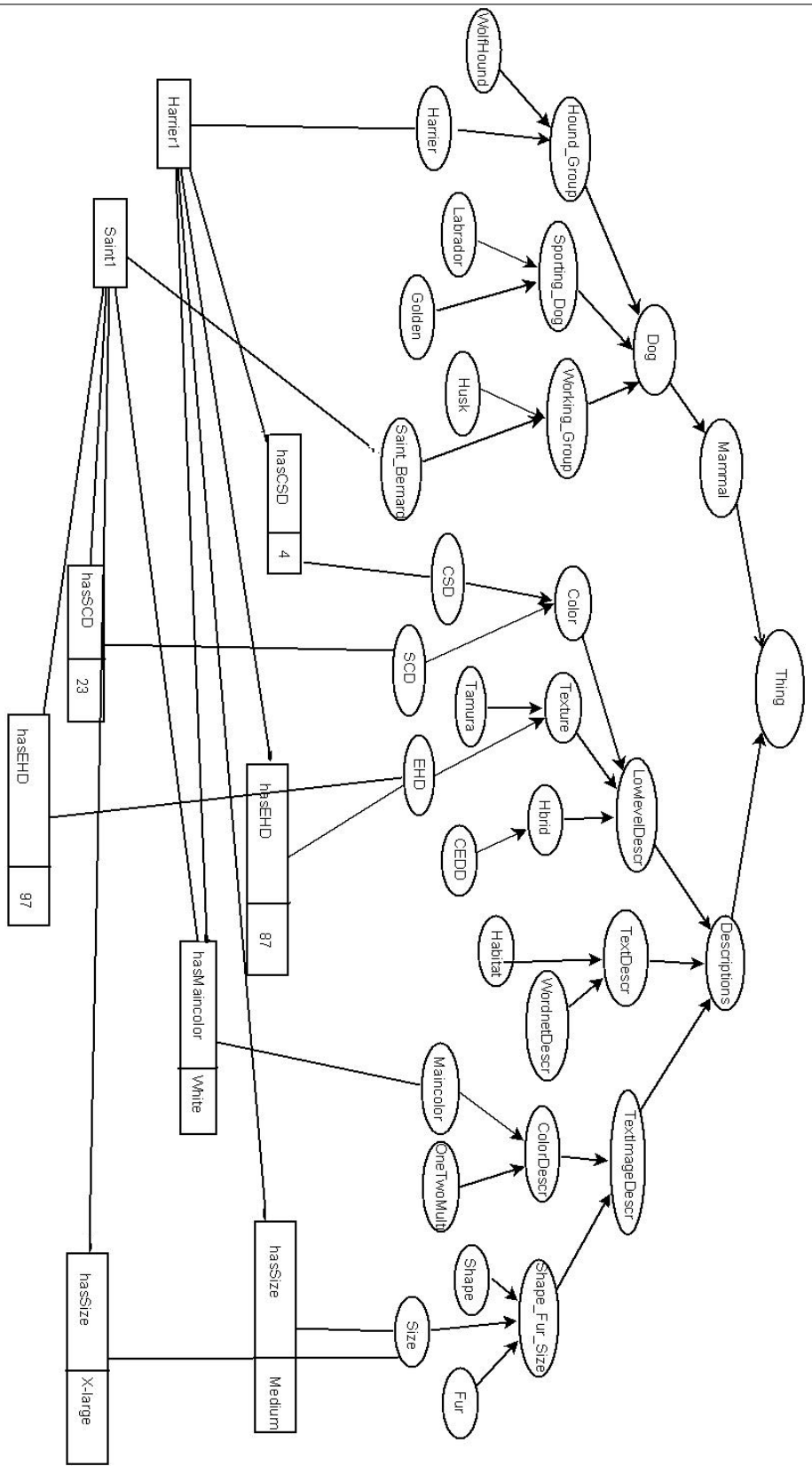

Figure 3.1: The SIA ontology for the 30 dog breeds.

3.1.3 Image Ontology

Summarizing, the SIA ontology (shown in Figure 3.1) consists for the following parts: a) A nouns class hierarchy of dog breeds with instances of leaf classes representing images and b) Descriptions arranged in class hierarchies with instances of leaf classes representing low-level description and high level information. Semantic categories (images) are aslo associated to low-level description and high level information. Object properties are used to connect instances of semantic classes (images) with instances from classes containing description (low-level or high level). Some properties that were generated are *hasCEDD*, *hasSize*, *hasColorLayout* , *hasWordnet* (Figure 3.1).

Dog taxonomy, visual text description and text description were generated manually using Protege [19]. The Wordnet annotation was managed automatically using wordnet tab of Protege. Low-level description and association to images were generated using Protege-Owl api[20]. Image feature extraction is implemented using Lire [11]. Finally an image ontology in OWL [21] language under the domain of dog breeds was successfully constructed.

Figure 3.2: Original and Region Of Interest image

3.2 Image Content Analysis

3.2.1 Region Of Interest(ROI) selection

SIA annotation starts with an unknown image as input (i.e., the image of a dog in this work). The input image may contain several regions, and it is rather natural to assume that some of them may be more relevant for the user's information need than others (e.g., the foreground or the center of the image might be more relevant than the background or elements towards the boundary of the image). Systems like Blobworld [22] allow the user to select the region of interest from the automatically segmented image. Apart from segmentation, background subtraction techniques can be used as a region of interest selection method [23]. However, automatic segmentation and background subtraction techniques do not always give desired results, (e.g., if the object is partially occluded or consists of several parts).

In this work, dog's head is chosen as the most representative part of a dog image for further analysis. Background subtraction and segmentation techniques were used to exclude dog's head from the image. Unfortunately these techniques gave us unreliable results (ie., also background along with the regions of the head).

A sub-image of the original image containing dog's head with as less as possible background is our final ROI. This sub-image is excluded by the original image with a rectangle selection. All experiments and evaluations held upon the ROI image for the images in the database and the query image as well. Figure 3.2 shows the original and the ROI image.

3.2.2 Distance Normalization

Image similarity between query images and images in the database, is calculated not only considering image distance beween individual pair of images, but also with respect to all distances between query image and images in the database.

In this work several MPEG-7 visual descriptors are chosen as features to describe images. The distance between two images, as a sum a weighted distances corresponding to the descriptors applied for representing image content (e.g. weighted Euclidean for CLD, L1 for CSD, Manhattan metric for Tamura). Notice that individual distances compute values in different ranges. For instance, the CLD distance ranges from 0 to 470.4 and CSD distance ranges from 0 to 65536. In this work, all image distances are normalized in the range of [0-1].

Three normalization methods have been tested in accordance to data for a decision tree:

Norm1: for each element i in a ranked list of k elements having distance d, the i normalized distance is

$$D = \frac{d_i}{d_{max}} \tag{3.1}$$

where d_{max} is the maximum distance?

Norm2: Linear scaling to unit range

$$D = \frac{d_i - \min}{\max - \min + 1} \tag{3.2}$$

where min and max are respectively the minimum and maximum possible values of d.

However, norm1 and linear normalization have been found problematic when applied to MPEG-7 descriptors. This is mainly to the following two reasons. Firstly, the maximum possible distance varies significantly among the MPEG-7 descriptors. Secondly, for a particular database, the distances of each descriptor usually fall into a small subrange of the entire possible range. As a result, the linear normalization will possibly compact the distances into a very narrow and indiscriminate range within [0, 1] and distances of different descriptors will be mapped to different subranges within [0, 1] which makes the distances of different descriptors incomparable.

In this work we apply Gaussian normalization that puts equal emphasis on the distances in each of the feature spaces. By doing this, we normalize the distances of each descriptor into a normal distribution with standard deviation being equal to 1.0.

$$D = \frac{1}{2}(1 + \frac{d_i - \mu}{3 * \sigma}) \tag{3.3}$$

where μ is the mean value of the distances and σ is the standard deviation of the distances, both are calculated from the image database. The advantage of Gaussian

normalization over 3.1 and 3.2 is that the presence of a few abnormally large or small values does not bias the importance of a feature measurement in computing the similarity between two images.

3.3 Image Similarity Computation

Given an unknown image, the ontology is searched to retrieve the images most similar to it. Image matching is implemented using image content descriptions (color, texture and hybrid features) between query image and images in the ontology. As previously mentioned, images in the ontology are represented as instances of semantic categories. Low-level features, of these instances, are stored in the ontology as low-level image descriptions. Thus, for calculating distances between query image and instances, low-level features are retrieved from the ontology. All distances are normalized in the range [0,1].

Real image are often rich in content and their information can rarely be represented by a single feature or attribute. A common fix to this problem is to represent image content by a set of features (eg features of color, texture, hybrid). Also, in most cases, different low-level features tend to be complementary to each other considering results of a retrieval process. In this work combination of multiple features is managed with an overall similarity measure.

In our system image similarity function between two images is defined as:

$$D(A,B) = \sum_i w_i d(A_i, B_i) \tag{3.4}$$

Where A, B are two images, i is the number of features and d(A_i,B_i) is the normalized distance between two images for feature i. Terms w_i represent the relative importance of the features themselves. The last issue before defining an image similarity function is the specification of weights. Weight specification utilizes the decision tree.

▷ **Decision Tree :**

Baratis in [24] proposed that weights are computed based on properties of a trained decision tree as follows.

$$w_{f_i} = \sum_{node_j = f_i} \frac{Maxdepth + 1 - depth(f_i)}{\sum_j Maxdepth + 1 - depth(node_j)} \tag{3.5}$$

Where fi is every feature, $node_j$ is each node of the decision tree and Maxdepth is the maximum depth of the tree. The summation is taken over all nodes. This formula suggests that the higher a feature is and the more frequently it appears, the higher its weight will be. As will be shown below, the final tree will contain 5 features, indicating

the remaining feature are surplus and are replaced by others. Features not appearing in the decision tree are not taken into account in the computation of similarity between two images.

The training of the decision tree relies on a training data set. In this work an instance in the dataset is represented by a pair of images. Low-level features are represented by the similarity measure computed between the two images. Finally the class that the image pair belong to, is determined by the decision tree. In this work, the decision tree is built upon images of 30 classes (ie. representing 30 dog breeds). Image pairs are classified into two classes(similar,no-similar). Tree nodes contain low-level attributes for training while leaf nodes contain classified instances(similar,no-similar). The stratified-cross validation gives 80.1521% correctly classified instances in the final leaf nodes (similar, no-similar). The training data set has 1446 instances, 590 similar pairs and 856 no-similar ones. Each image pair is represented by the Gaussian normalized distance computed over the set of features (i.e., a set of 7 distances corresponding to distances of the 7 image features). The decision tree accepts a pair of images (in fact similarities of 7 features) as input and computes whether the two images are similar or not. Weka [25] was used as an interface of testing and visualizing the decision tree. C4.5 (J48) [26] was the learning method applied and stratified cross-validation [27] was the method for testing the decision tree.

Similar images to the query are obtained from the ontology according to the overall similarity measure computed by Eq. 3.4. Finally the retrieved images are ranked in decreasing order of visual similarity.

3.4 Image Annotation

Image annotation is a process that takes as input an unknown image and assigns to it a label denoting its category along with a text description. In this work the semantic category the query image belongs to, is calculated based on the analysis of the retrieval results. After semantic category estimation, query image then inherits all high level information of the category estimated.

▷ **Semantic Category Estimation :**

Given an unknown image as input, a set of 15 images most similar to it are retrieved from the ontology. Image search is performed by applying Eq. 3.4. The next steps involves selection of the best semantic class (ie. which is assigned to the input image).

Three methods computing the most characteristic semantic category (the annotation) to the unknown image have been tested:
a)**Best match:** Select the class of the image which is most similar to the input image.

b)**Maxoccurence:** Select the class that has the maximum number of instances in the answer set. If multiple classes have the same number of instances, employ best match between competing classes.

c)**AVR:**
Last, we used AVR [12]. For a query q with a ground-truth size of NG(q), we define rank(k) as the rank of the kth ground-truth image on the top-N result list. The average retrieval rank is then computed as follows.

$$AVR(q) = \sum_{k=1}^{NG(q)} \frac{rank(k)}{NG(q)} \tag{3.6}$$

In this work, AVR is used as standard metric of semantic category estimation of unknown image (correctly classified image) based on retrieval results. A ranked list of instances is obtained answering the image query. The result list includes all semantic categories of our ontology (breeds), having a fixed number of instances(six images per breed). For each of the semantic categories(breeds) AVR is computed. From all AVR calculated(same number as semantic categories), the class having the best AVR (eg the least) would be the semantic class the query image belongs to. If multiple classes have the same AVR, best match is employed between competing classes.

The change with actual AVR is that the ground-truth images in the database, for a given query, are half (three) from their actual number (six)[1].

[1]Building an ontology has to have instances in a specific class that could correspond to different users needs. In our system we have 6 images per breed so we are guessing that a particular query could be similar to 3 of them.

Algorithm 1 Semantic Category Estimation using AVR

$rankbreedimage(qi1)$ = best first rank in ranked list for picture from breed i for the query q
$rankbreedimage(qi2)$ = second best rank in ranked list for picture from breed i for the query q
$rankbreedimage(qi3)$ = third best rank in ranked list for picture from breed i for the query q

query q
List list= similar retrieved images in decreasing order of similarity
$list[0]$ = best image
for eachbreed==i **do**
  AVR[i]=(rankbreedimage(qi1)+rankbreedimage(qi2)+rankbreedimage(qi3))/3 3.6
end for
bestavr=bubblesort(AVR,breeds)
bestavr[0]=best breed

▷ **Interoperability :**

An annotation can be expressed in different formats. This format is important to ensure interoperability with other (semantic web) applications. MPEG-7 is often used as the metadata format for exchanging automatic analysis results whereas OWL and RDF are more appropriate in the Semantic Web world. The semantic class the query image belongs to is estimated. Information about the class is encapsulated in the ontology with properties of OWL language. An additional important issue is to achieve semantic interoperability between OWL and MPEG-7.

The final step includes mapping owl classes and properties of them to MPEG-7 format so that the multimedia content services offered by different vendors may interoperate. Our main objective, in the final annotation, is to describe the owl class recognized, its superclass and some descriptive owl object properties containing high level information. Transformation rules, between OWL and Mpeg-7, were adopted in order to achieve semantic interoperability.

Tsinaraki et.al [28] proved that OWL Ontologies can be transformed into MPEG-7 Abstract Semantic Entity Hierarchies. OWL domain ontology classes and individuals are represented as MPEG-7 semantic elements of type 'SemanticBaseType'. The 'AbstractionLevel' element of the 'SemanticBaseType' and the MPEG-7 semantic relationships are used to capture ontology semantics. An abstract semantic entity that represents a domain ontology class is related with each of its subclasses through a pair of 'Relation' elements of type 'generalizes'/' specializes'. The properties defined in the domain ontology classes are transformed into 'Property' elements (datatype properties)

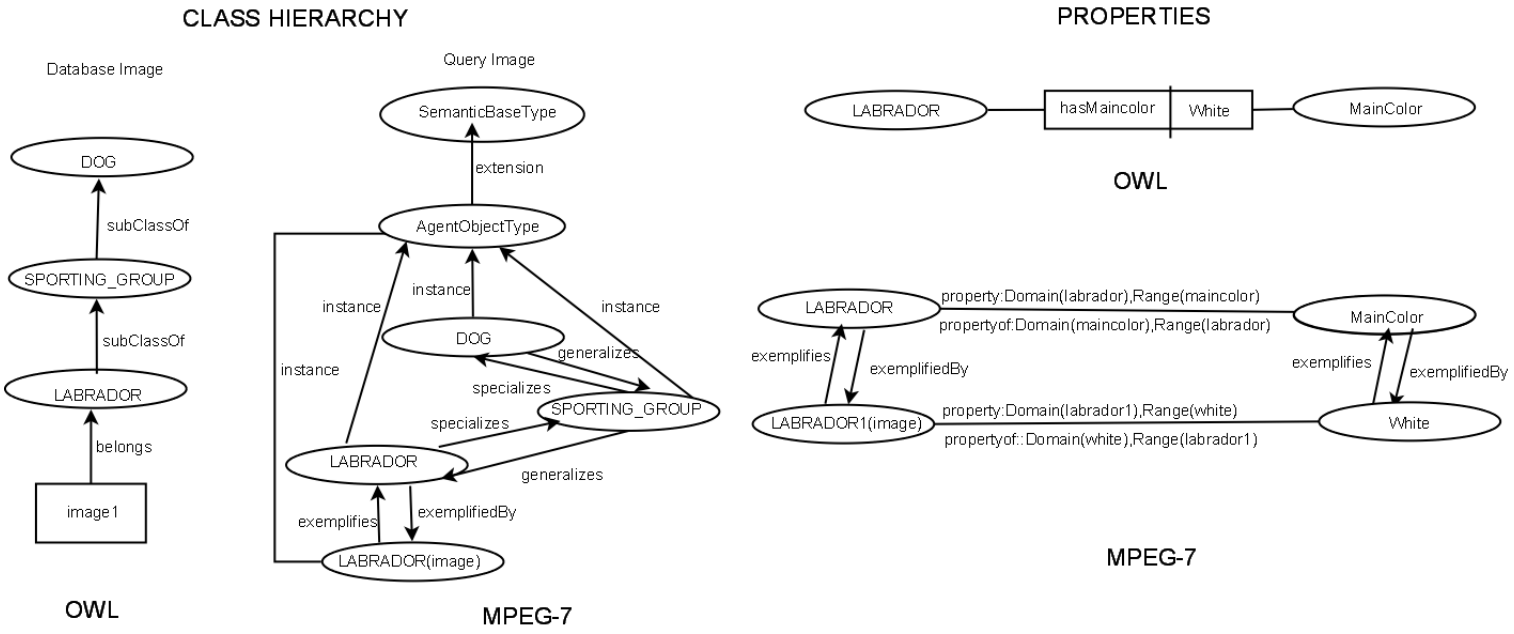

Figure 3.3: Mapping OWL to Mpeg7

or into pairs of 'property'/'propertyOf' relationships that associate the semantic entities representing the classes and the property values (Object properties). In addition, an abstract semantic entity that represents a class is related with the concrete semantic entities representing the class individuals through pairs of 'exemplifies/exemplifiedBy' relationships. In this work images are represented as individuals of abstract semantic entities. Thus images are represented as MPEG-7 semantic elements of type 'SemanticBaseType'. In addition, images are related with abstract semantic entities through pairs of 'exemplifies/exemplifiedBy' relationships. Finally in the 'SemanticBaseType' representing the image, an element of type 'MediaOccurenceType' with an 'URI' value is added in order to provide information about the uri of the image.

Since the image is classified in one of known classes, it is considered as an individual of its corresponding class. Some contents of the annotation file are the same for every query image. This includes description of the ontology, some abstract semantic entities and abstract semantic properties (ie. dog class). Next the semantic class recognized (dog breed) and its superclass (group) are represented with semantic elements of type 'SemanticBaseType'. Their abstraction connection are represented with 'Relation' elements of type 'generalizes'/'specializes'. The query image is represented as individual through pairs of exemplifies/exemplifiedBy relationships with the semantic entity is related to. In the 'SemanticBaseType' representing the query image an element of 'MediaOccurenceType' is added to provide the 'URI' of the query image. Figure 3.3. Trasformation rules where applied thanks to connection between Protege-Owl api [20], and MPEG-7 MDS (Multimedia Description Schemes) api [29].

4 Experiments

This section presents experimental results. Firstly, the database of images on which experiments were performed is over-viewed. Next some results on retrieval and annotation are presented and discussed.

4.1 Experimental Setup

Most CBIR Systems choose COREL Image Database as their testing database. Other popular databases include the public Wang dataset [17], Flickr [15] and S3 data set of MPEG-7 [18].

In this work a certain domain has gained our attention thanks to the possibility of testing well known low level features in image recognition. In addition a certain domain could have good ontological description. Our system is evaluated on images of dog breeds including about 180 color images divided into 30 semantic categories(breeds). Thirty of about one hundred dog breeds that exist worldwide were selected, with our choice depending on breed's popularity. Annotation of an unknown image is a process of assigning to it a label denoting its category along with a text description. Figure 4.1 shows some picture of our database in mosaic after ROI selection.

Figure 4.1: Example images of dog species after ROI select.

4.2 SIA Results

SIA is implemented as a system that has the same characteristics with all typical QBE(Query By Example) systems.

In a retrieval system, it is interesting to know if an image is either relevant or non-relevant to a particular query. Thus, two classic indicators are defined by information retrieval researchers:

$$precision = \frac{|\{\text{Relevant images}\} \cap \{\text{Retrieved images}\}|}{|\{\text{Retrieved images}\}|}$$

$$recall = \frac{|\{\text{Relevant images}\} \cap \{\text{Retrieved images}\}|}{|\{\text{Relevant images}\}|}$$

In our work we used average retrieval-recall as a quality measure based on Eq. 4.1.

$$P(r) = \sum_{i=1}^{N_q} \frac{P_i(r)}{N_q} \tag{4.1}$$

where N_q number of queries and $P_i(r)$ - precision at recall level r for i^{th} query.

▷ **Multiple Descriptors vs. Single Descriptor :**

All low-level features analyzed in chapter 2 checked for their retrieval performance when applied by themselves only. Different retrieval processes have been considered according to each low-level feature space and similarity measure. The results were compared with retrieval results based on a combined similarity measure using multiple low-level features. Thirty queries were provided, as many as the different dog breeds in our database. In Figure 4.2 all results present averages over 30 queries between 1 and 15 answers. Different methods are represented by different colors.

Our perception experiments assert that using a single descriptor for retrieval has drawbacks and does not perform well in most situations. Multiple descriptors can improve the retrieval performance and give better results even if their relative importance is considered as equal, on an overall similarity measure.

▷ **Feature weighting by decision trees :**

Decision trees were used to calculate the relative importance of low-level features in a certain domain. Figure 4.3 illustrates an example decision tree. Tree nodes contain low-level attributes for training while leaf nodes contain classified instances(similar,no-similar).

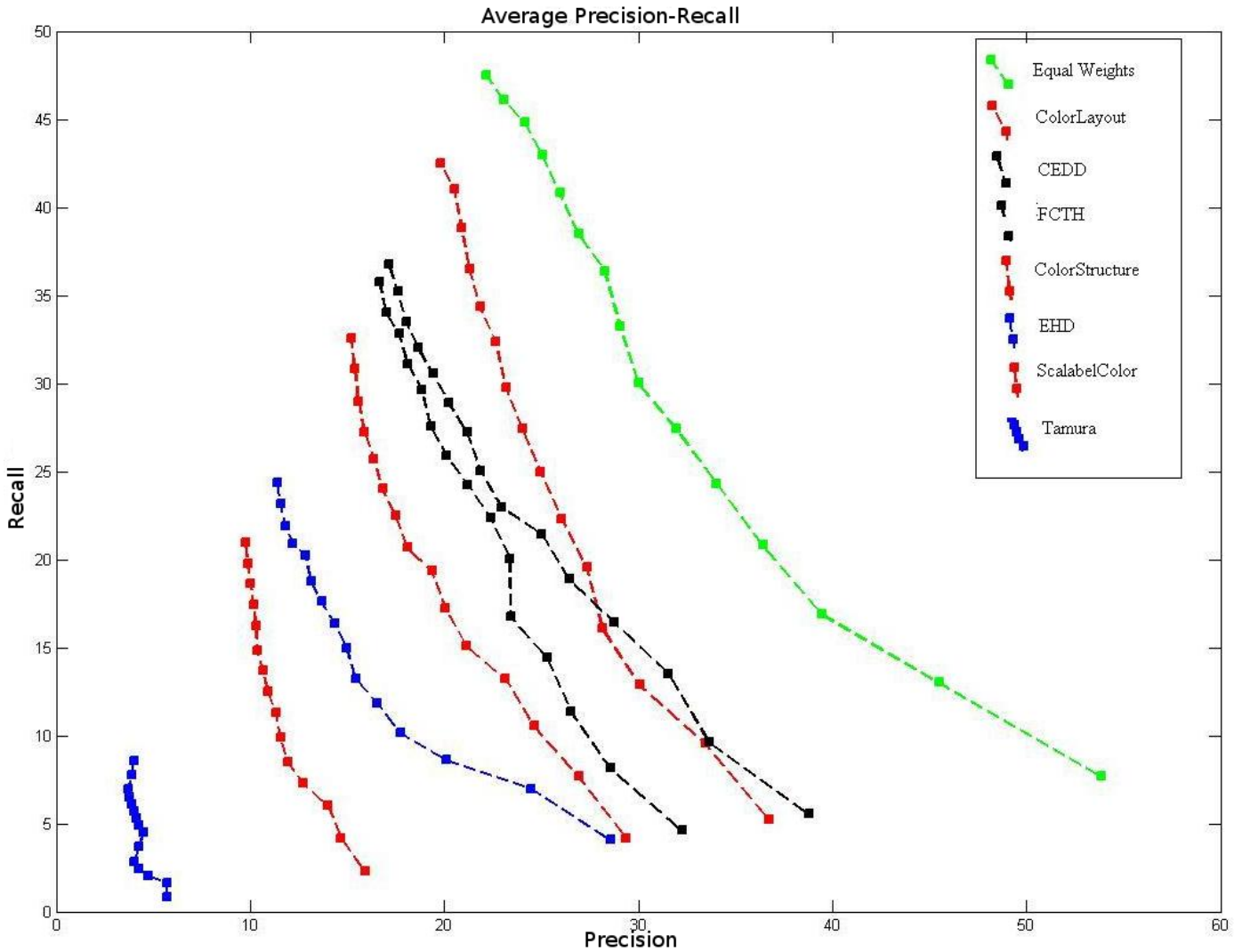

Figure 4.2: Average Precision-Recall for 30 Queries over all descriptors.

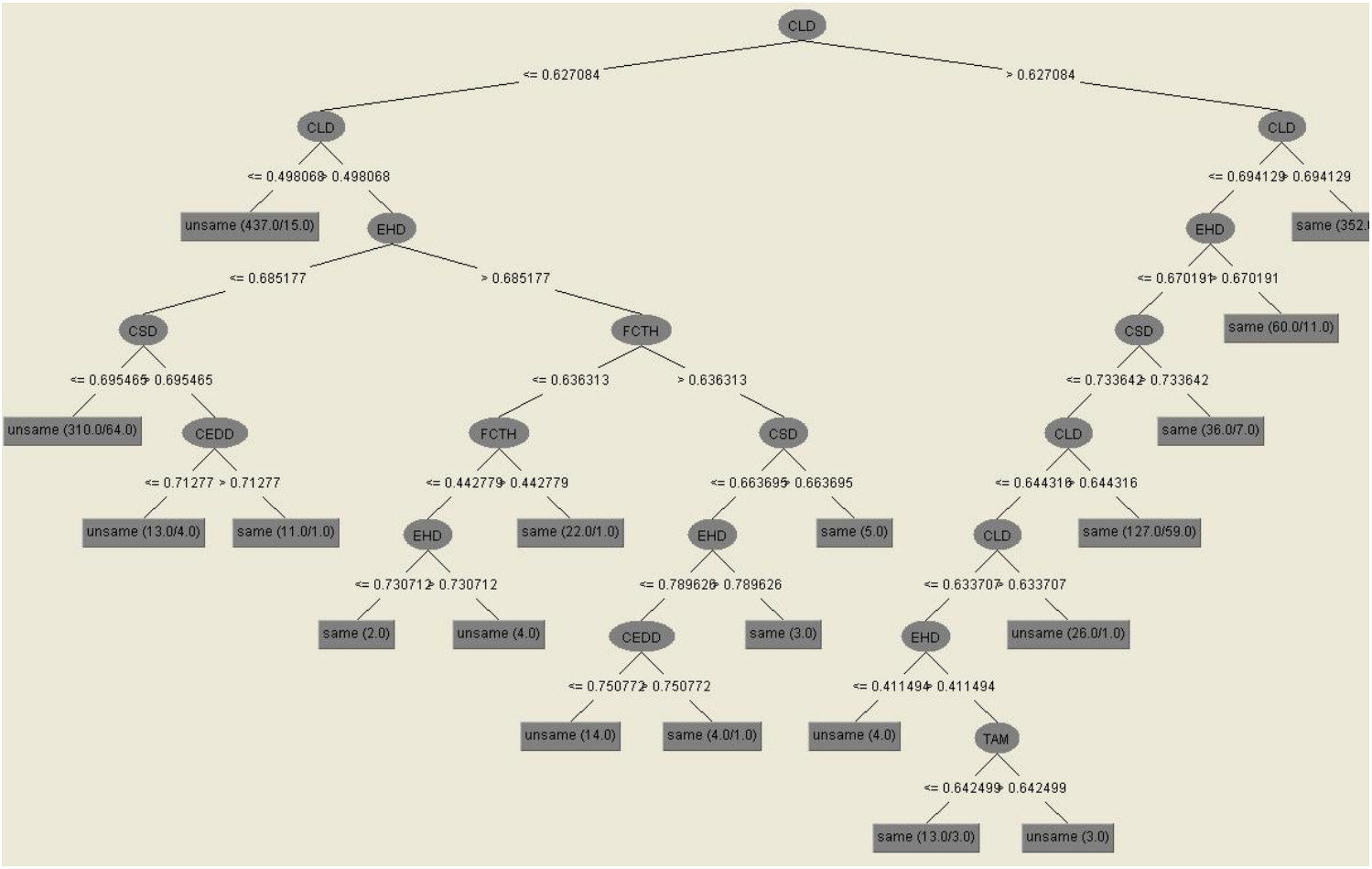

Figure 4.3: Decision Tree.

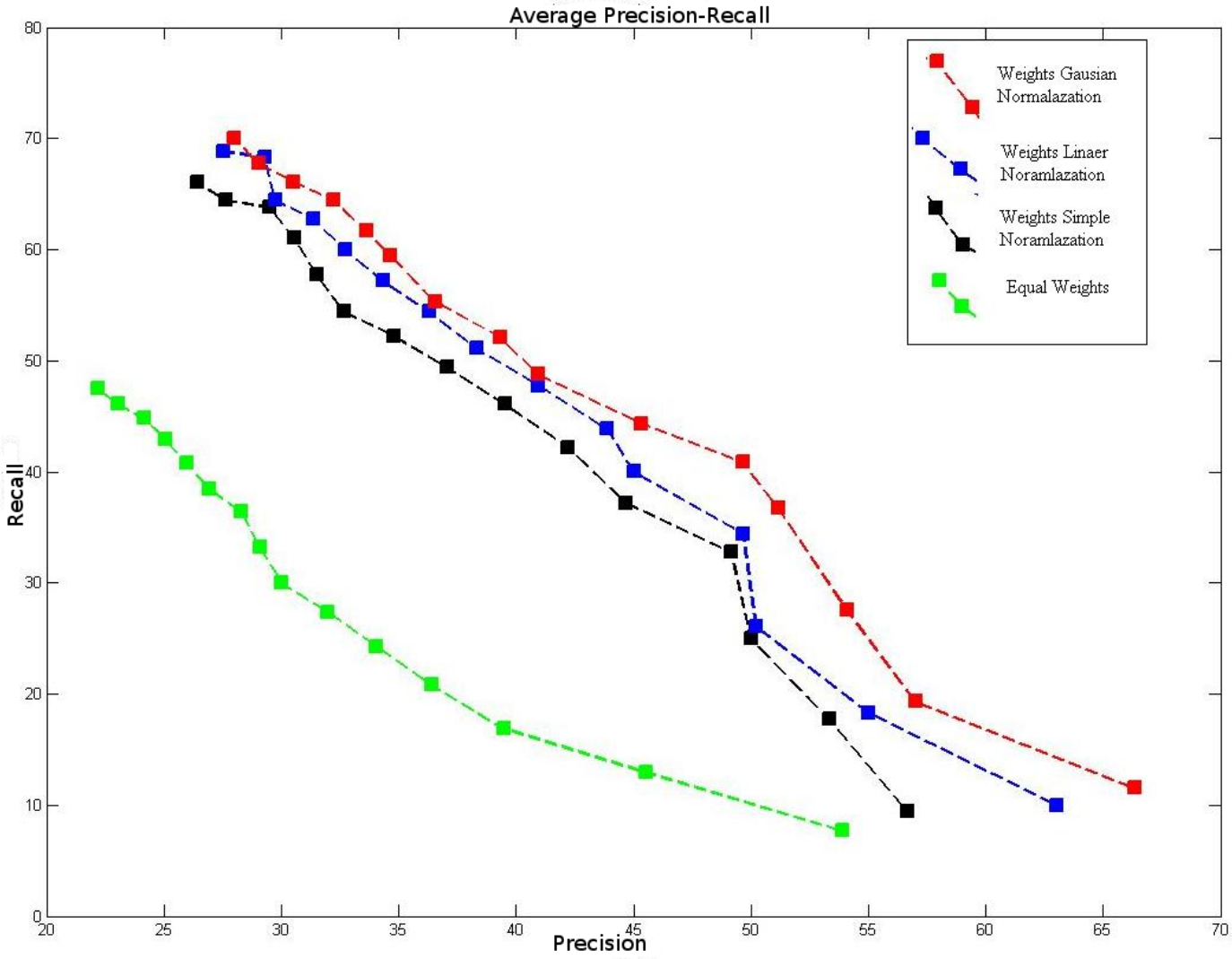

Figure 4.4: Average Precision-Recall for 30 Queries corresponding to different distance normalization methods.

First impression looking this diagram is that to classify a pair of images(similar,no-similar) we need to look first the root node descriptor similarity. As a result descriptor in root node must have bigger weight than others in an overall similarity measure.

The purpose of the following experiment is to demonstrate that Gaussian normalization results in most effective computation of similarity. Each normalization method required different training for learning feature weight and a different decision tree was constructed using the same test data. Fig. 4.4 illustrates average precision/recall corresponding to equal weight, linear and simple normalization methods, and finally Gaussian normalization, all discussed in subsection 3.2.2. Obviously, Gaussian normalization performs better than all its competitors achieving up to 10% better recall and 20% better precision.

Figure 4.5: Retrieval for query Doberman

Finally we present some actual examples of retrieval in 4 different queries(breeds). The combined similarity measure (equ 3.4) is used in these examples. Feature weights were derived from the decision tree in figure 4.3. Distances of features (decision tree data) were normalized with Gaussian normalization. Figures 4.5, 4.6, 4.7, 4.8 present examples of retrieval results.

In all figures, on the upper left corner is the query image and others images are the results in decreasing order of similarity. Apart from good results we can mention that first results of retrieval are close to the query not only in a machine view but also in human's perception system.

Figure 4.6: Retrieval for query Cocker Spaniel

Figure 4.7: Retrieval for query Cairn-Terrier

Figure 4.8: Retrieval for query Collie

4.3 Annotation

This section presents experimental results obtained on semantic category estimation of an unknown image and results in interoperability between OWL and MPEG-7.

▷ **Semantic Category Estimation :**

To properly annotate an unknown image, its semantic category should be first estimated (classification). In this work the semantic category is estimated based on further analysis of the retrieval results. Each query image retrieves, a ranked list of images is produced in decreasing order of similarity. Each of the known classes contain 6 pictures in the ranked list adopted. As we previously mentioned three methods have been tested for the semantic category estimation based on retrieval results. Best match, max occurrence in 15, AVR. Figure 4.9 shows some results.

Evaluation of the methods are shown in Figure 4.9. Methods were tested over 30 queries and average of right classification (system identified its semantic class correctly) is being shown.

In case of Best Match, 53% of the queries belong to the same class as the first ranked image, 10% of the queries belong to the same class as the second best image in the ranked list, and 7% of queries belong to the same class with the third best image in the ranked list.

In case of max occurrence in 15, 60% of the queries belong to the class that has the maximum number of instances in first 15 answers, 12% of the queries belong to the class that has the second maximum number of instances in first 15 answers, and 10% of the queries belong to the class that has the third maximum number of instances in first 15 answers.

In case of AVR, 63% of the queries belong to the class with the best AVR (in fact the less), 20% of the queries belong to the class with the second best AVR, and 6% of the queries belong to the class with the third best AVR.

As we can see with AVR about 90% of queries are classified correctly in the first three best AVR, corresponding to semantic classes.

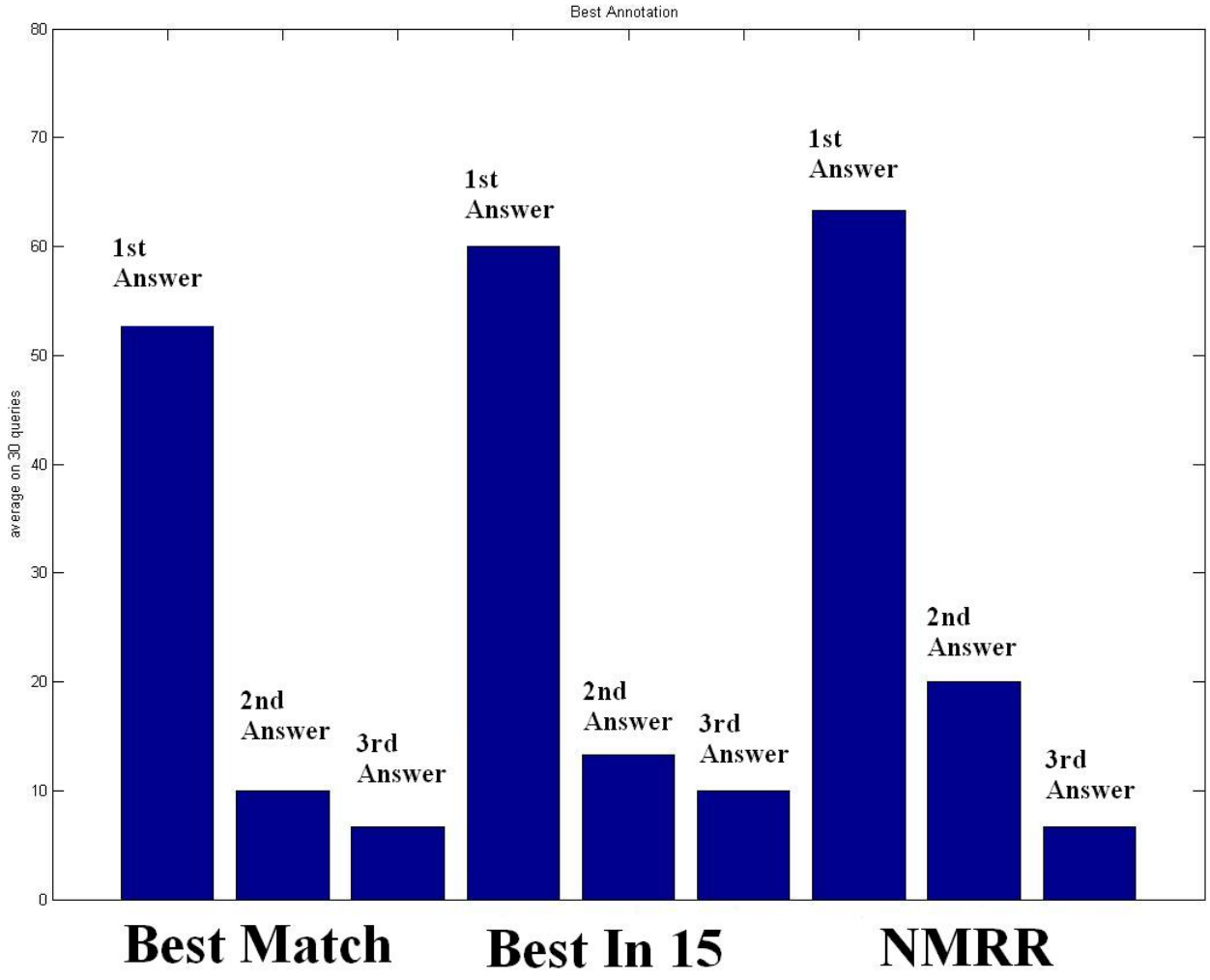

Figure 4.9: Results for Best Annotation

▷ **Interoperability :**

The query image inherits high level information from its assigned semantic class. Information about the class is encapsulated in the ontology with properties of OWL language. An important issue is to achieve semantic interoperability between OWL and MPEG-7. This step relates to mapping owl classes and properties of them to MPEG-7 format so that the multimedia content services offered by different vendors may interoperate. We our focusing on adjusting information, of the class recognized, its superclass and some high level descriptive object properties in the ontology.

Some contents of the annotation file are the same for every query image. This includes description of the ontology, some abstract semantic entities and abstract semantic properties (ie. dog class). Fig 4.10

In Fig 4.10 a brief description of our ontology is presented. The description of the abstract class dog, is achived with a semantic entity dog, by an element of type SemanticBaseType. The Dimension attribute of AbstractionLevel has the value 1, showing that the semantic entity is abstract. Next some abstract object properties of our ontology are described. Object properties are represented by pairs of 'property'/'propertyOf' relationships that associate the semantic entities representing the classes and the property values. The domain and target values in the above relationships have the respective values of the object properties in the ontology.

```
<?xml version="1.0" encoding="UTF-8"?>
<Mpeg7 xmlns="urn:mpeg:mpeg7:schema:2001" xmlns:xsi="http://www.w3.org/2001/XMLSchema-inst
  <Description xsi:type="urn:SemanticDescriptionType" xmlns:urn="urn:mpeg:mpeg7:schema:200
    <Semantics id="Dogs">
      <AbstractionLevel dimension="1"/>
      <Label>
        <Name>Dog breed ontology</Name>
      </Label>
      <Property>
        <Term>
          <Name>href</Name>
          <Definition>file:/C:/Documents%20and%20Settings/pierce/workspace/onto/dog1.owl</
        </Term>
      </Property>
      <SemanticBase xsi:type="urn:AgentObjectType" id="Dog">
        <AbstractionLevel dimension="1"/>
        <Label>
          <Name>Dog semantic class</Name>
        </Label>
        <Relation type="property" source="#Dog" target="#Habitat">
          <Header xsi:type="urn:DescriptionMetadataType">
            <Comment>
                  <FreeTextAnnotation>hasHabitat</FreeTextAnnotation>
            </Comment>
          </Header>
        </Relation>
        <Relation type="propertyof" source="#Habitat" target="#Dog">
          <Header xsi:type="urn:DescriptionMetadataType">
            <Comment>
                  <FreeTextAnnotation>hasHabitat</FreeTextAnnotation>
            </Comment>
          </Header>
        </Relation>
        <Relation type="property" source="#Dog" target="#Size">
          <Header xsi:type="urn:DescriptionMetadataType">
            <Comment>
                  <FreeTextAnnotation>hasSize</FreeTextAnnotation>
            </Comment>
          </Header>
        </Relation>
        <Relation type="propertyof" source="#Size" target="#Dog">
          <Header xsi:type="urn:DescriptionMetadataType">
            <Comment>
                  <FreeTextAnnotation>hasSize</FreeTextAnnotation>
            </Comment>
          </Header>
        </Relation>
        <Relation type="property" source="#Dog" target="#wordnet">
          <Header xsi:type="urn:DescriptionMetadataType">
            <Comment>
                  <FreeTextAnnotation>haswordnet</FreeTextAnnotation>
            </Comment>
          </Header>
        </Relation>
        <Relation type="propertyof" source="#wordnet" target="#Dog">
          <Header xsi:type="urn:DescriptionMetadataType">
            <Comment>
                  <FreeTextAnnotation>haswordnet</FreeTextAnnotation>
            </Comment>
```

Figure 4.10: Annotation of Abstract contents

Continuing the annotation process, some inadequate information about the semantic class recognized are inserted. That is its superclass (group of dogs belong) and final annotation on the image. In 4.11 and 4.12 we present the final annotation of pictures belonging to semantic categories of labrador-retriever and german-sheperd respectively.

Abstract semantic entities, group and dog breed, are represented through SemanticBaseType elements. Through pair of 'Relation' elements of type 'generalizes'/'specializes' we represent superclasses and subclasses respectively. In addition with the Dimension attribute of AbstractionLevel having the value 1 we show that this classes are abstract. Last an image is considered as an individual of abstract semantic entity related with each of the semantic entities through pairs of 'exemplifies'/'exemplifiedBy' relationships. The Dimension attribute of AbstractionLevel has value 0 showing that this class is concrete (image). Finally the image inherits high level description in pairs of 'property'/'propertyOf' relationships with domain and range the actual values of the class recognized. The above relationships represent ontology object properties.

```
<SemanticBase xsi:type="urn:AgentObjectType" id="Sporting_Dog_Group">
  <AbstractionLevel dimension="1"/>
    <Label>
      <Name>Sporting_Dog_Group semantic class</Name>
    </Label>
  <Relation type="specializes" source="#Sporting_Dog_Group" target="#Dog"/>
  <Relation type="generalizes" source="#Dog" target="#Sporting_Dog_Group"/>
</SemanticBase>
<SemanticBase xsi:type="urn:AgentObjectType" id="Labrador_Retriever">
  <AbstractionLevel dimension="1"/>
    <Label>
      <Name>Labrador_Retriever semantic class</Name>
    </Label>
  <Relation type="specializes" source="#Labrador_Retriever" target="#Sporting_Dog_Group"/>
  <Relation type="generalizes" source="#Sporting_Dog_Group" target="#Labrador_Retriever"/>
</SemanticBase>
 <SemanticBase xsi:type="urn:AgentObjectType" id="Labrador_Retriever1">
  <AbstractionLevel dimension="0"/>
    <Label>
      <Name>Labrador_Retriever1 image</Name>
    </Label>
  <MediaOccurrence>
    <MediaInformationRef href="C:/Documents and Settings/pierce/Desktop/imagefinal/back1/t
  </MediaOccurrence>
  <Relation type="exemplifies" source="#Labrador_Retriever1 " target="#Labrador_Retriever'
  <Relation type="exemplifiedBy" source="#Labrador_Retriever" target="#Labrador_Retriever1
  <Relation type="property" source="#Labrador_Retriever1" target="#Indoor-Outdoor">
    <Header xsi:type="urn:DescriptionMetadataType">
      <Comment>
            <FreeTextAnnotation>hasHabitat</FreeTextAnnotation>
      </Comment>
     </Header>
  </Relation>
  <Relation type="propertyof" source="#Indoor-Outdoor" target="#Labrador_Retriever1">
    <Header xsi:type="urn:DescriptionMetadataType">
      <Comment>
            <FreeTextAnnotation>hasHabitat</FreeTextAnnotation>
      </Comment>
    </Header>
  </Relation>
  <Relation type="property" source="#Labrador_Retriever1" target="#Large">
    <Header xsi:type="urn:DescriptionMetadataType">
      <Comment>
            <FreeTextAnnotation>hasSize</FreeTextAnnotation>
      </Comment>
    </Header>
  </Relation>
  <Relation type="propertyof" source="#Large" target="#Labrador_Retriever1">
    <Header xsi:type="urn:DescriptionMetadataType">
      <Comment>
            <FreeTextAnnotation>hasSize</FreeTextAnnotation>
      </Comment>
    </Header>
  </Relation>
  <Relation type="property" source="#Labrador_Retriever1" target="#breed originally from L
    <Header xsi:type="urn:DescriptionMetadataType">
      <Comment>
            <FreeTextAnnotation>haswordnet</FreeTextAnnotation>
```

Figure 4.11: Labrador Annotation

```
<SemanticBase xsi:type="urn:AgentObjectType" id="Herding_Dog_Groups">
  <AbstractionLevel dimension="1"/>
    <Label>
      <Name>Herding_Dog_Groups semantic class</Name>
    </Label>
  <Relation type="specializes" source="#Herding_Dog_Groups" target="#Dog"/>
  <Relation type="generalizes" source="#Dog" target="#Herding_Dog_Groups"/>
</SemanticBase>
<SemanticBase xsi:type="urn:AgentObjectType" id="German_Shepherd">
  <AbstractionLevel dimension="1"/>
    <Label>
      <Name>German_Shepherd semantic class</Name>
    </Label>
  <Relation type="specializes" source="#German_Shepherd" target="#Herding_Dog_Groups"/>
  <Relation type="generalizes" source="#Herding_Dog_Groups" target="#German_Shepherd"/>
</SemanticBase>
 <SemanticBase xsi:type="urn:AgentObjectType" id="German_Shepherd1">
  <AbstractionLevel dimension="0"/>
    <Label>
      <Name>German_Shepherd1 image</Name>
    </Label>
  <MediaOccurrence>
    <MediaInformationRef href="C:/Documents and Settings/pierce/Desktop/imagefinal/back1/t
  </MediaOccurrence>
  <Relation type="exemplifies" source="#German_Shepherd1 " target="#German_Shepherd"/>
  <Relation type="exemplifiedBy" source="#German_Shepherd" target="#German_Shepherd1"/>
  <Relation type="property" source="#German_Shepherd1" target="#Outdoor">
    <Header xsi:type="urn:DescriptionMetadataType">
      <Comment>
            <FreeTextAnnotation>hasHabitat</FreeTextAnnotation>
      </Comment>
     </Header>
  </Relation>
  <Relation type="propertyof" source="#Outdoor" target="#German_Shepherd1">
    <Header xsi:type="urn:DescriptionMetadataType">
      <Comment>
            <FreeTextAnnotation>hasHabitat</FreeTextAnnotation>
      </Comment>
    </Header>
  </Relation>
  <Relation type="property" source="#German_Shepherd1" target="#Large">
    <Header xsi:type="urn:DescriptionMetadataType">
      <Comment>
            <FreeTextAnnotation>hasSize</FreeTextAnnotation>
      </Comment>
    </Header>
  </Relation>
  <Relation type="propertyof" source="#Large" target="#German_Shepherd1">
    <Header xsi:type="urn:DescriptionMetadataType">
      <Comment>
            <FreeTextAnnotation>hasSize</FreeTextAnnotation>
      </Comment>
    </Header>
  </Relation>
  <Relation type="property" source="#German_Shepherd1" target="#11 breed of large shepherd
    <Header xsi:type="urn:DescriptionMetadataType">
      <Comment>
            <FreeTextAnnotation>hasWordnet</FreeTextAnnotation>
      </Comment>
```

Figure 4.12: Alsatian Annotation

5 Conclusion

An image annotation system using ontologies and low-level image analysis is presented and discussed. SIA deals with images of dog breeds but may be easily extend to handle any image domain. This would simply require creation of different domain ontology and a different training stages adjusted to the images of the new domain. In SIA, ontoligies provide the means for capturing low level and semantic information (including text and low level image features) for the images of the application domain at hand. Alternatively, the process of image annotation using SIA can be viewed as an attempt for narrowing the semantic gap between low - semantic level features which are typically easily extracted from unknown images. SIA then computes a semantic annotations for the unknown image that includes not only a class name but also a description of its class.

Firstly an ontology model is built capturing multiple low-level descriptors and high level text information in a hierarchical way. Instances of leafs classes in the ontology are used for retrieval and annotation. The process of annotating an unknown image is implemented in steps. A query image is provided by the user to obtain similar images from the ontology. Image matching is implemented using image content descriptions. Two color descriptors , two texture descriptors and two hybrid(color and texture) descriptors were used. An overall distance measure between images is proposed as a weighted sum of differences on the above features. The relative importance of features in this distance (their weights) are computed using machine learning by decision trees. The semantic category of an unknown image is computed based on AVR(Average Retrieval Rank). For interoperability reasons the final annotation of the image is in MPEG-7 format. This is achieved through mapping OWL classes and properties, to elements of MPEG-7 MDS(Multimedia Description Schemes).

▷ **Future Work :** The building of the ontology enables other groups to define their own domain knowledge for image retrieval and annotation. The definitions in the ontology can be easily shared and exchanged. Furthermore this ontology can be easily extended to upper categories of human perception (animals) or picturable nouns hierarchy of Wordnet [2].

In a machine learning view, decision trees include number of parameters to discuss such as the degree of pruning, data over-fitting and test-method. Experiments on this parameters could lead to better weighting schemes on low-level features for retrieval purposes.

In interesting extension relates to calculating an overall similarity between images using multiple decision trees as weighting scheme of features in regions of images. Next an overall similarity between images is derived with the sum of similarities that came from regions of an image.

Bibliography

[1] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-Based Image Retrieval at the End of the Early Years. *IEEE transactions Pattern Analysis Machine Intelligence*, 22 - 12:1349 – 1380, 2000. 5

[2] A. Miller. Wordnet: a lexical database for english. *Commun. ACM*, 38(11):39–41, 1995. 6, 19, 45

[3] X. Zhou, M. Wang, Q. Zhang, J. Zhang, and B. Shi. Automatic Image Annotation by an Iterative Approach: Incorporating Keyword Correlations and Region Matching. In *CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 25–32, New York, NY, USA, 2007. ACM. 8

[4] H. Wang, L. T. Chia, and S. Liu. Image Retrieval ++–Web Image Retrieval with an Enhanced Multi-Modality Ontology. *Multimedia Tools Appl.*, 39(2):189–215, 2008. 8

[5] H. Wang, S. Liu, and L.-T. Chia. Does Ontology Help in Image Retrieval? In *MULTIMEDIA '06: Proceedings of the 14th annual ACM international conference on Multimedia*, pages 109–112, New York, NY, USA, 2006. ACM. 8

[6] E. Hyvonen, A. Styrman, and S. Saarela. Ontology-based image retrieval. In *Towards the semantic web and web services, Proceedings of XML Finland 2002 Conference*, pages 15–27, October 21–22 2002. 8

[7] S. Jiang, T. Huang, and W. Gao. An Ontology-based Approach to Retrieve Digitized Art Images. In *WI '04: Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 131–137, Washington, DC, USA, 2004. IEEE Computer Society. 8

[8] A. Th. Schreiber, B. Dubbeldam, J. Wielemaker, and B. Wielinga. Ontology-Based Photo Annotation. *IEEE Intelligent Systems*, 16(3):66–74, 2001. 8

[9] V. Mezaris, I. Kompatsiaris, and M. G. Strintzis. Region-based image retrieval using an object ontology and relevance feedback. *EURASIP J. Appl. Signal Process.*, 2004:886–901, 2004. 8

[10] K.-W. Park, J.-W. Jeong, and D.-H. Lee. OLYBIA: Ontology-Based Automatic Image Annotation System Using Semantic Inference Rules. In Kotagiri Ramamohanarao, P. Radha Krishna, Mukesh K. Mohania, and Ekawit Nantajeewarawat,

editors, *DASFAA*, volume 4443 of *Lecture Notes in Computer Science*, pages 485–496. Springer, 2007. 8

[11] M. Lux and S. A. Chatzichristofis. LIRE: Lucene Image Retrieval: An Extensible Java CBIR Library. In *MM '08: Proceeding of the 16th ACM international conference on Multimedia*, pages 1085–1088, New York, NY, USA, 2008. ACM. 8, 16, 20, 22

[12] B. S. Manjunath, J.-R. Ohm, V. V. Vasudevan, and A. Yamada. Color and Texture Descriptors. *IEEE Transactions on Circuits and Systems for Video Technology*, 11:703–715, 2001. 9, 13, 27

[13] P. Salembier and T. Sikora. Introduction to MPEG-7: Multimedia Content Description Interface. John Wiley & Sons, Inc., New York, NY, USA, 2002. 9, 13

[14] M. Lux, J Becker, and H. Krottmaier. Caliph & Emir: Semantic Annotation and Retrieval in Personal Digital Photo Libraries. In *Proceedings of CAiSE'03 Forum at 15th Conference on Advanced Information Systems Engineering*, 2003. 9, 13

[15] Benoit Rat. Semantic Images Annotation and Retrieval. Master's thesis, EPFL, LCAV, EPFL, 1015 Lausanne, March 2008. 9, 30

[16] H. Tamura, S. Mori, , and T. Yamawaki. Texture Features Corresponding to Visual Perception. *IEEE Transactions on Systems, Man and Cybernetics*, 8(6):460–473, 1978. 15

[17] S. A. Chatzichristofis and Y. S. Boutalis. CEDD: Color and Edge Directivity Descriptor: A Compact Descriptor for Image Indexing and Retrieval. In Antonios Gasteratos, Markus Vincze, and John K. Tsotsos, editors, *ICVS*, volume 5008 of *Lecture Notes in Computer Science*, pages 312–322. Springer, 2008. 17, 30

[18] S. A. Chatzichristofis and Y. S. Boutalis. FCTH: Fuzzy Color and Texture Histogram - A Low Level Feature for Accurate Image Retrieval. In *WIAMIS '08: Proceedings of the 2008 Ninth International Workshop on Image Analysis for Multimedia Interactive Services*, pages 191–196, Washington, DC, USA, 2008. IEEE Computer Society. 18, 30

[19] Protégé Project. The Protégé Ontology Editor and Knowledge Acquisition System. 22

[20] H. Knublauch, R. W. Fergerson, N. F. Noy, and M. A. Musen. The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications. pages 229–243. 2004. 22, 29

[21] M. Dean and G. Schreiber. OWL Web Ontology Language Reference. W3C recommendation, W3C, February 2004. 22

[22] C. Carson, M. Thomas, S. Belongie, J. Hellerstein, and J. Malik. Blobworld: a System for Region-based Image Indexing and Retrieval. Technical report, Berkeley, CA, USA, 1999. 23

[23] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut": Interactive Foreground Extraction using Iterated Graph Cuts. *ACM Trans. Graph.*, 23(3):309–314, 2004. 23

[24] E. Baratis, E.G.M. Petrakis, and E.E. Milios. Automatic Website Summarization by Image Content: A Case Study with Logo and Trademark Images. *IEEE Transactions on Knowledge and Data Engineering*, 20(9):1195–1204, 2008. 25

[25] S. R. Garner. WEKA: The Waikato Environment for Knowledge Analysis. In *In Proc. of the New Zealand Computer Science Research Students Conference*, pages 57–64, 1995. 26

[26] Ian H. Witten. 3.2 Decision Trees. In *Data Mining : Practical Machine Learning Tools and Techniques with Java Implementations*, pages 58–63, 2000. 26

[27] Ian H. Witten. 5.3 Cross-validation. In *Data Mining : Practical Machine Learning Tools and Techniques with Java Implementations*, pages 125–127, 2000. 26

[28] C. Tsinaraki, P. Polydoros, and S. Christodoulakis. Interoperability Support between MPEG-7/21 and OWL in DS-MIRF. *IEEE Trans. on Knowl. and Data Eng.*, 19(2):219–232, 2007. 28

[29] Ana. Benitez, D. Zhong, S.-F. Chang, and John R. Smith. MPEG-7 MDS Content Description Tools and Applications. In *CAIP '01: Proceedings of the 9th International Conference on Computer Analysis of Images and Patterns*, pages 41–52, London, UK, 2001. Springer-Verlag. 29