Ministry of Higher Education and Scientific Research
University of Carthage
Tunisia Polytechnic School

Graduation Project Report

Image Annotation Based on Partially Labeled Data

Host University

Supervised by

Dr Hichem Frigui
Professor, CECS at UofL

Ms. Wided Souidene
Professor, TPS

Elaborated by

Ameni TRABELSI
3rd year engineer student
SISY (Signals and Systems)

From 1 March to 31 August 2016

Acknowledgments

First of all, I would like to thank University of Louisville for offering me this opportunity to have this memorable experience and to work in these encouraging conditions.

I would also love to thank Dr. Frigui, my supervisor and Professor of computer science and computer engineering at University of Louisville, for his support and assistance during these 6 months and for his precious remarks. I would also express my gratitude to Mrs. Wided Souidene, my supervisor at Tunisia Polytechnic School, for her continuous assistance.

I would definitely express my gratitude to Tunisia Polytechnic School for this rich and fruitful training and courses during these 3 years.

Last but not least, I would never forget the support and the encouragement that I had from my father, my mother and my brother during my studies.

Abstract

Image Annotation based on partially labeled data consists in assigning labels to new-to-the system images based on learning data in which the labels are bag-level only. The image annotation in this work is both image-level and region-level. The groundtruth data is only image-level labeled. The image annotation requires image segmentation, feature extraction and Multiple Instance Learning Classification.

Contents

CONTENTS

List of Figures

Introduction

An object annotation system is used to detect real world objects from an image of the world. Object recognition is performed by human effortlessly and instantaneously. However, this task is surprisingly difficult. Algorithmic description of this task for machine implementation has been very difficult.

The object annotation problem can be defined as a labeling problem based on partially or totally labeled data models. Formally, given an image containing one or more objects of interest that can be foreground objects or background and a set of labels corresponding to models already known by the system, the system should assign labels to the regions of new unlabeled images correctly. In our work, we will be trying to annotate new-to-the-system images based on partially labeled data models. In fact, the learned models used to annotate the regions of an image will be based on coarse image level labeling.

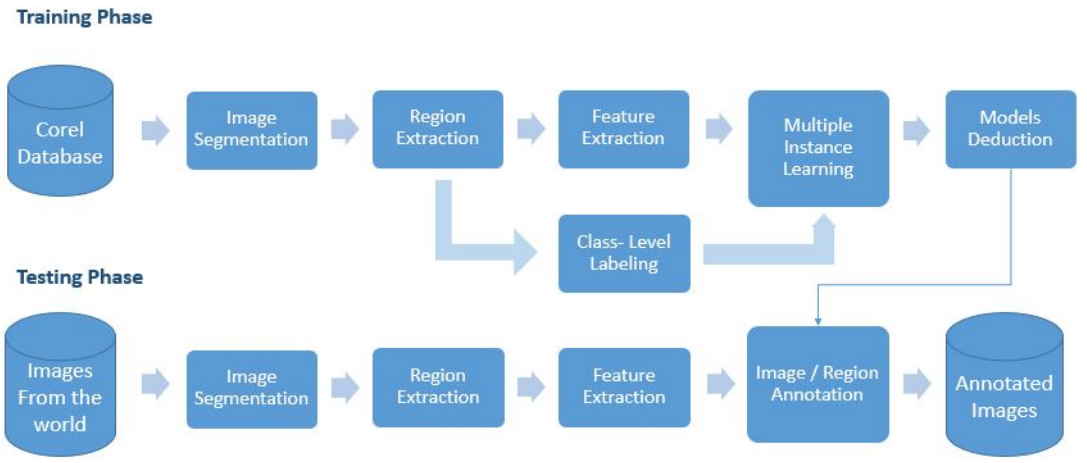


Figure 1: Training and Testing Phase of Image Annotation Process

The object annotation problem is very tied to the segmentation problem. Segmentation cannot be done without at least a partial recognition of objects and object annotation is not possible without segmentation.

Although image annotation can be done without any previous segmentation and region extraction, in general region level image annotation has shown to give better results and labels of better quality[1][2]. The latter is due to the fact that extracted regions give more precise information that could be more general in case of image-level annotation[1][2][3][4]. Obviously, a better quality of segmentation is expected to give better performance of region labeling techniques.

Image segmentation have been a field of deep research and important advances, however most segmentation algorithms tend to fall in one of two extremes. On the one hand, many sophisticated techniques have shown very good results but they were too time consuming and thus not appropriate for some applications such as image retrieval and object recognition [5]. On the other hand, there exist methods that are simple and fast such as the grid segmentation method but that do not provide a good support for automatic annotation[6]. A logical alternative consists of the methods that offer a tradeoff between efficiency and quality. This project presents one of such alternative segmentation techniques that aims at giving support to automatic image annotation and content-based image retrieval (CBIR) [7].

1 Project Context and Specifications

1.1 Host Lab Presentation

1.1.1 University of Louisville

University of Louisville is a research university located in Kentucky's largest metropolitan area. It is a state supported university that has joined the university system in 1970. It was established in 1798 and had been a municipally supported public institution for many decades before joining the university system. The university has three campuses. The first is the 287-acre Belknap Campus which is the biggest and houses 7 of the 12 colleges and schools. The second is the Health Science Center which houses the university's health related programs and the University of Louisville Hospital. The last is the 243-acre Shelby Campus that is located in eastern Jefferson County.

The university of Louisville has a total student population of 22,000 and can major in more than 170 areas of study including business, engineering and medicine. U of L is also known for its Louisville Cardinals varsity teams, which compete in the NCAA I Atlantic Coast Conference. The university's budget is estimated to 1,228,604,700$. It enrolls around 6,997 persons in faculty and staff and more than 141,000 Alumni members residing in the United States and around the world.

1.1.2 The Multimedia Research Lab of UofL

The J. B. Speed School was first established in 1925 under the name of Speed Scientific School as a memorial to James Breckenridge Speed with an endowment by his son and daughter. James Breckenridge Speed was an industrial pioneer in the city of Louisville. He was a leader in the establishment of Louisville's street railway system. He is known for many achievements in terms of coal interests and he was among the first to recognize the significance that Portland Cement would have in the growth of America. In 2003, the J.B. Speed Scientific School has changed its name to J. B. Speed School of Engineering.

The Multimedia Research Lab is a part the Department of Computer Engineering and Computer Science at UofL. The main research interests can be divided into two main streams. First, many work has been done in the areas of Multimedia Information Indexing and Retrieval, Multimedia Data Mining, Multimedia Information Fusion, clustering and classification of multimedia objects. Second, the MRL group is involved in real time land mine detection project using Ground Penetrating Radar.

1.2 Project Specifications

As mentioned above the Multimedia research lab has various fields of research interests. One of the main subjects is Image Information Retrieval. The main challenge of this work is to achieve non-complex efficient and rapid algorithm that is able to retrieve significant information from images and that shows satisfying results and good performance. The main tasks of this work are:

- Achieve a robust image segmentation using low-level features and unsupervised learning algorithms.
- Extract color, texture and structure features to describe image regions.
- Using coarse image level training data, learn models for different image regions using multiple instance learning algorithms.
- Test new images with unknown labels to label its regions using the deduced models of the previous step.

2 State Of The Art

2.1 Feature Extraction

A good feature should characterize the salient properties of an object and can discriminate between objects from different categories.It is, in fact, supposed to ignore irrelevant information during classification. For example, in the case of detecting a person, the color of the clothing, the shadow, the presence or absence of a hat on the head of the person are all types of information that maybe irrelevant to the classification task. However, the presence of arms and legs, hair, face, and many other components could be the main information that helps the classifier distinguish a person from other objects.

2.1.1 Color Features

Color is one of the most important features of images. Color features are defined subject to a particular color space or model. A number of color spaces have been used in literature, such as RGB, LUV, HSV and HMMD. Once the color space is specified, color feature can be extracted from an image or from an image region. A number of important color features have been proposed in the literatures. Some of these features are outlined in the following subsections.

Color Moments

Color moments are one of the simplest yet very effective features. Common moments include the mean, the standard deviation, and skewness in each color channel. Let μ_i,σ_i,$\gamma_i(i = 1, 2, 3)$ denote the mean, standard deviation, and skewness of each color channel of an image respectively. These moments are computed using:

$$\mu_i = \frac{1}{N} \sum_{j=1}^{N} f_{ij} \tag{2.1}$$

$$\sigma_i = (\frac{1}{N}\sum_{j=1}^{N}(f_{ij} - \mu_i)^2)^{\frac{1}{2}} \tag{2.2}$$

$$\gamma_i = (\frac{1}{N}\sum_{j=1}^{N}(f_{ij} - \mu_i)^3)^{\frac{1}{3}} \tag{2.3}$$

Where f_{ij} is the color value of the i^{th} color channel (e.g; red, green, blue) of the j^{th} image pixel and N is the total number of pixels in the image.

Color Histograms

Another common color descriptor that can describe the content of an image is the color histogram. The color histogram, as defined in [8], consists of discretizing the image colors, given a discrete color space (e.g., R,G,B), and counting the occurrence of each discrete color in the image. The color histogram method is discrete and simple to implement [9]. But one of the disadvantages of this method is that it does not take into consideration any spatial information. Thus it is not robust to significant appearance changes [10].

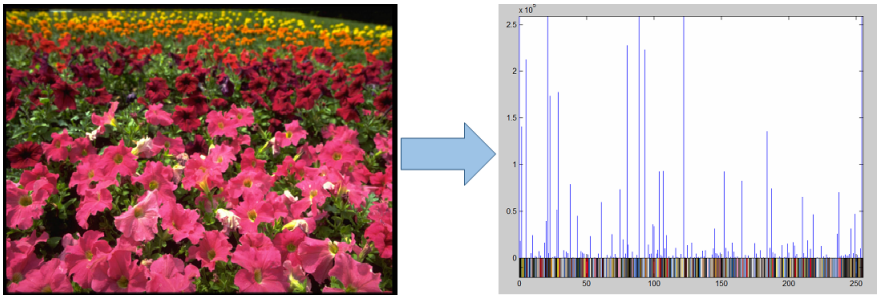


Figure 2.1: An example of Color Histogram

Color Structure Descriptor

The color structure descriptor (CSD) represents an image by its color distribution [11].

In contrast to the image histogram, the CSD does not compute a simple frequency of each color bin. Instead it computes the frequency while taking into account the co-occurrence of different colors in a small neighborhood. In particular, for any position in the image, a block is taken into consideration and the bins of the

existing colors are incremented by 1.

Hence, unlike the color histogram, this descriptor can distinguish between two images in which a given color is present in identical amounts but where the structure of the groups of pixels having that color is different in the two images. The figure (2.2) [12] shows two different images with two iso-color planes: Grey and Black. Using a standard color histogram, the two images are described as identical, as they have the same number of Black and Grey pixels. However, as we can see the structure of the two images differs significantly.

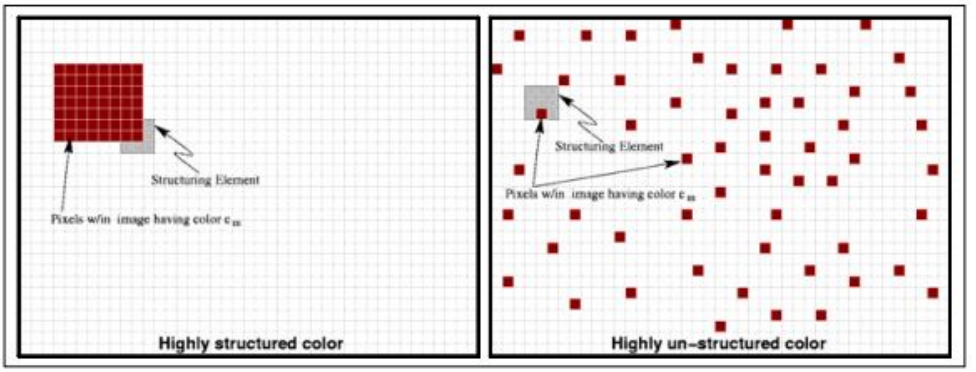


Figure 2.2: Two images having the same color histogram. But the right one has much more grey components in CSD description

2.1.2 Texture Features

Texture is another feature that can help in segmenting images into regions of interest and classifying those regions. In some images, it can be the defining characteristic of regions and critical in obtaining a correct analysis.

Texton Feature

Textons refer to fundamental micro-structures in natural images (and videos) and are considered as the atoms of pre-attentive human visual perception (Julesz, 1981).

Texton is where an image is modeled by feature texton histogram over a dictionary of textons. Textons were shown to be very useful for recognition tasks from gray-valued images [14].

The texton learning based methods [14,15, 16,17, 18] can be categorized with the following attributes: dense image sampling; image filter response and patch features; a posterior partitioning of the feature space; hard label assignment; histogram based image representation. In [14], for each class, the training images are registered and filtered to produce a set of 48-dimensional response vectors. Then the feature vectors are clustered by using the K-means clustering method and the texton histogram is used as the texture model for classification.
A gray image is convoluted with a filter bank. In our work we use a 49*49*48 texton filter bank[14]. First, a training phase needs to be done to build the texton library or filter bank. Second, each local feature of the image is assigned to the nearest texton in the library. That's why, the library size should not be very big so the feature extraction will not be time consuming.

Texton-based texture classifiers classify textures based on their texton frequency histogram.A codebook should be created by extracting texture patches and converting them to textons.Texton feature is a frequency histogram of the codebook.

Local Binary Patterns

The Local Binary Patterns (LBP)[19] is an efficient operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considering the results as a binary number.

In the Simple LBP, or basic LBP, we consider 3x3 pixel blocks and each pixel within the block is thresholded by the central pixel of the block. The result is blocks containing ones (if pixel is larger than center pixel) and zeros (if pixel is smaller than center pixel). After that, the sum of each binary pixel of the resulted block multiplied by 2 power its position in the block will be assigned to the central pixel.

$$LBP = \sum_{p=0}^{P-1} s(g_p - g_0)2^p \tag{2.4}$$

Where P=9 (we have 9 pixels per block).g_0 is the gray level value of the central pixel and g_p is the gray level value of p^{th} pixel in the block. We have 9 pixels per block so the maximum value a pixel can get by LBP is $2^8 = 256$. In (2.4) s is the sign function defined as follow:

$$\begin{cases} s(g_p - g_0) = 1 \text{ if } g_p \geq g_0 \\ s(g_p - g_0) = 0 \text{ else} \end{cases} \tag{2.5}$$

A variation of the LBP, called Efficient LBP [19], takes into consideration the rotational aspect of LBP. For example, let g_c denotes the gray level of an arbitrary pixel (x,y). $g_c = I(x, y)$ and g_p is the gray value of a sampling pixel in an evenly spaced circular neighborhood of P sampling points and radius R. We have:

$$g_p = I(x_p, y_p), p = 0..P - 1 \tag{2.6}$$

$$\begin{cases} x_p = x + Rcos(2\pi p/P) \\ y_p = y - Rcos(2\pi p/P) \end{cases} \tag{2.7}$$

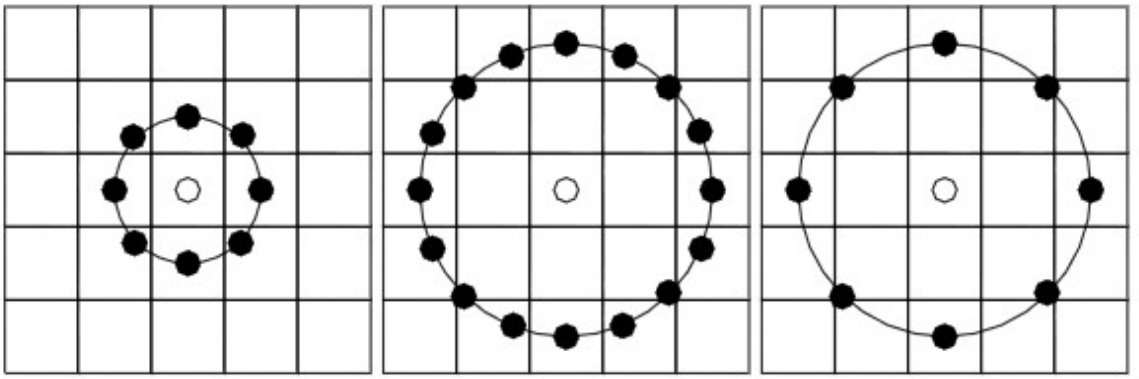


Figure 2.3: The circular neighborhoods of (P, R) = (8, 1), (16, 2) and (8, 2) respectively

The efficient LBP consists of testing the different rotations of a given (P, R) that is achieved by rotation invariant mapping, and then retaining the configuration that gives the minimum values of LBP [19].

$$LBP^{ri}_{P,R} = \min_i ROR(LBP_{P,R}, i) \tag{2.8}$$

In (2.8) $ROR(x, i)$ denotes the circular bitwise right rotation of bit sequence x by i steps. For example, 8-bit LBP codes 100000010b, 00101000b and 00000101b all map to the minimum code 00000101b [19].

Shift based LBP is computed via calculating the minimum of the different rotations of the result of the difference between the original image and the shifted image. In fact the image is shifted by an arbitrary vector. Then a difference between the shifted and the original image is calculated to determine the relevant LBP elements. A rotational invariant LBP is then applied to the difference to deduce the LBP histogram.

Edge Histogram Descriptor

In the Edge Histogram Descriptor (EHD)[29], an image is divided into 4x4 sub-images. Each sub-image is further divided into small square blocks called image-blocks. Then, an edge histogram is generated to represent the edge distribution in the sub-image by concatenating edge histograms of the image-blocks of that sub-image. For the different edge types, we define 5 edge filters. The first four edge filters describe vertical edge, horizontal edge, 45 degree and 135 degree diagonal edges. The 5th edge filter describe the non-directional edge i.e. the edges do not belong to any of the four described edges.

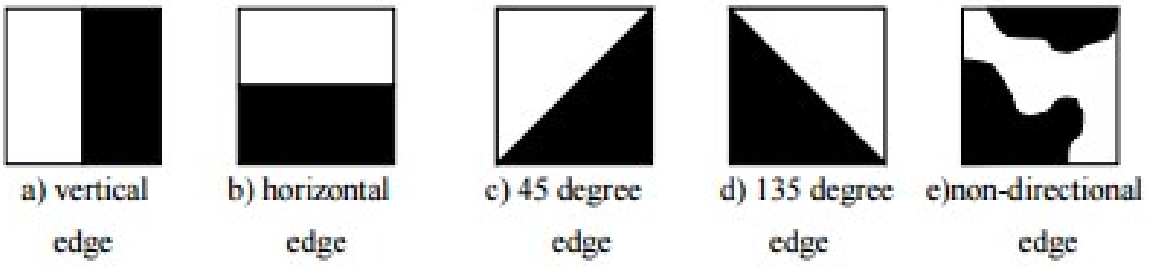


Figure 2.4: Different types of EHD Edges

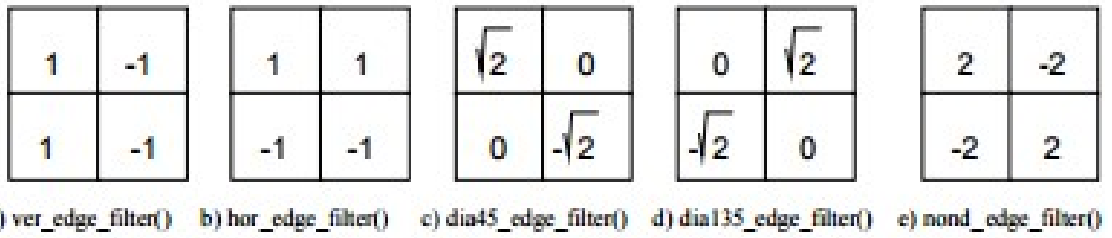


Figure 2.5: Filters used in EHD

EHD consists in extracting information about the different types of edges an image can contain and thus can give relevant information about the different shapes and forms in an image.

Histogram of Oriented Gradients

Another texture feature that is based on edges is The Histogram of Oriented Gradient(HOG)[34]. This feature can capture edge or gradient structure that is very

characteristic of local shape. It is also relatively invariant to local geometric and photometric transformations. Moreover, HOG cannot be affected by cell rotations and translations. HOG is flexible in terms of spatial and orientation sampling densities, and after normalization it can achieve illumination invariance.

To compute HOG, an image is divided into cells and for each cell a local 1-D histogram of gradient directions or edge orientations over the pixels of the cell is accumulated. The combined histogram entries form the HOG feature. In the following, we outline the different steps involved in computing the HOG descriptor.

HOG approximates the two components I_x and I_y of the gradient of I by central differences:

$$\begin{cases} I_x(r;c) = I(r;c+1) - I(r;c-1) \\ Iy(r;c) = I(r-1;c) - I(r+1;c) \end{cases} \tag{2.9}$$

The gradient is then transformed to polar coordinates, with the angle constrained to be between 0 and 180 degrees, so that gradients that point in opposite directions are identified:

$$\begin{cases} \mu = \sqrt{I_x^2 + I_y^2} \\ \theta = \frac{180}{\pi}(tan^{-1}(\frac{I_y}{I_x}) mod \pi) \end{cases} \tag{2.10}$$

Once the gradient is computed at every pixel of the window, HOG divides the window into adjacent, non-overlapping cells of size CxC pixels. Then, a histogram of the gradient orientations, binned into B bins, is computed for each cell.

After computing a histogram for each cell, HOG groups the cells into overlapping blocks of 2x2 cells each, so that each block has size 2Cx2C pixels. Then the four cell histograms in each block are concatenated into a single block feature b, and normalized using:

$$b \leftarrow \frac{b}{\sqrt{\|b\|^2 + \epsilon}} \tag{2.11}$$

In (2.11), ϵ is a small positive constant that avoids division by zero.
Cell histograms need to be normalized to reduce the effect of changes in contrast between images of the same object.

2.1.3 Scale-Invariant Feature Transform

In this work, we will be interested in the SIFT descriptor [25]. The first step, is to detect key points. Several algorithms were developed in this area of research

namely the Harris corner detector algorithm, the Hessian detector algorithm, the SURF detector and the Laplacian of Gaussian.

Laplacian of Gaussian

To detect the interest points, we use the difference of Gaussian (DoG) filter to approximate the normalized Laplacian of Gaussian (LoG) filter. The DoG filter significantly accelerates the computation process.

The LoG filter can be defined as follow [27]. First, a Gaussian scale space is constructed and candidate points are extracted by searching local extrema in a series of DoG images. To do so, the scale space or image pyramid is constructed by a variable-scale Gaussian, which is:

$$G(x, y, \sigma) = \frac{1}{2\Pi\sigma^2} exp(\frac{-(x^2 + y^2)}{2\sigma^2}) \tag{2.12}$$

where (x,y) denotes the coordinate of a point and the scale factor is σ .
Mikolajczyk found that the extrema of scale-normalized Laplacian of Gaussian (LoG), $\sigma^2\nabla^2 G$, are the most stable local points on an image compared to a range of other image functions such as gradient, Hessian and Harris. Moreover, the difference-of-Gaussian filter provides a close approximation to the scale-normalized LoG, which is:

$$G(x, y, k\sigma) - G(x, y, \sigma) = (k - 1)\sigma^2\nabla^2 G \tag{2.13}$$

where k is a constant factor. So scale space extrema extracted in the DoG function convolved with the image, D(x,y,σ), are regarded as candidate points, and

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \tag{2.14}$$

where I(x,y) is an input image.

An effective way to construct $D(x, y, \sigma)$ and detect local extrema is by following these steps. The initial image is incrementally blurred with successively larger Gaussian filters to produce images which are separated by a constant multiplicative factor in scale-space. Afterwards, difference of Gaussian images are produced by subtracting each blurred image from the adjacent image. For detecting the local extrema of $D(x, y, \sigma)$, each sample point is compared to its eight neighbors in the current image and nine neighbors in the scale above and below. The sample point is selected as a candidate point if its value is larger or smaller than all of its neighbors.

SIFT Descriptor

Once a key point candidate has been found by comparing a pixel to its neighbors, the next step is to perform a detailed fit to the nearby data for location, scale, and ratio of principal curvatures. This information allows points to be rejected that have low contrast (and are therefore sensitive to noise) or are poorly localized along an edge.

For each image sample, L(x, y), at this scale, the gradient magnitude, m(x, y), and orientation, σ(x, y), is precomputed using pixel differences:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y+1))^2} \quad (2.15)$$

$$\theta(x,y) = \tan^{-1}\left(L(x,y+1) - L(x,y+1)\right)/(L(x+1,y) - L(x-1,y)) \quad (2.16)$$

An orientation histogram is formed from the gradient orientations of sample points within a region around the key point. The orientation histogram has 36 bins covering the 360 degree range of orientations. Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian-weighted circular window with a σ that is 1.5 times that of the scale of the key point. The next step is to compute a descriptor for the local image region that is highly distinctive yet is as invariant as possible to remaining variations [25].

A SIFT key-point is a circular image region with an orientation (Figure (2.6)). It is described by a geometric frame of four parameters: the key-point center coordinates x and y, its scale (the radius of the region), and its orientation (an angle expressed in radians). The SIFT detector uses as key-points image structures which resemble "blobs". By searching for blobs at multiple scales and positions, the SIFT detector is invariant (or, more accurately, covariant) to translation, rotations, and re scaling of the image [23].

The key-point orientation is also determined from the local image appearance and is covariant to image rotations. Depending on the symmetry of the key-point appearance, determining the orientation can be ambiguous. In this case, the SIFT detectors returns a list of up to four possible orientations, constructing up to four frames (differing only by their orientation) for each detected image blob. In Figure (2.7) an example of three images are shown as well as the detected key-points using SIFT.

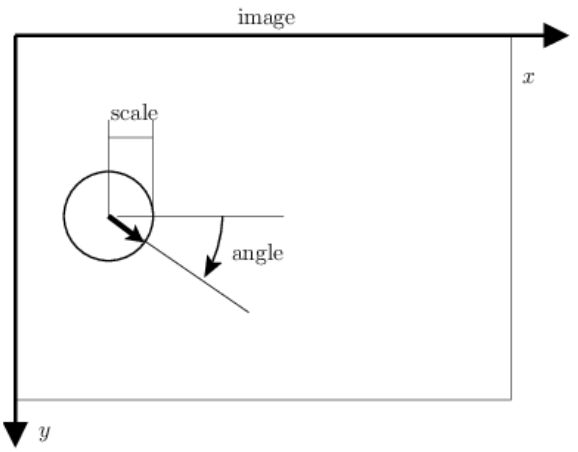


Figure 2.6: A SIFT key point

Figure 2.7: Sample Images with Their Detected Key-points.

2.2 Clustering

Image clustering and categorization is a means for high-level description of image content[30]. The goal is to find a mapping of the archive images into classes (clusters) such that the set of classes provide essentially the same information about the image archive as the entire image-set collection. The generated classes provide a concise summarization and visualization of the image content that can be used for different tasks related to image database management.

There exist several methods, we are going to outline some relevant ones in the following subsections.

2.2.1 K-means

K-Means is the simplest and most commonly used algorithm employing a square error criterion. It starts with a random, initial partition and keeps reassigning the samples to clusters, based on the similarity between samples and clusters, until a convergence criterion is met. Typically, this criterion is met when there is no reassignment of any sample from one cluster to another that will cause a decrease of the total squared error.
K-means algorithm is popular because it is easy to implement, and its time and space complexity is relatively small.
One of the drawbacks of K-means is that it does not take into account the shape or the size of the data. K-means will always try to minimize the within cluster sum of squares which will not give perfect clustering all the time.

The simple K-means clustering algorithm is computationally efficient and gives surprisingly good results if the clusters are compact, hyper-spherical in shape, and well separated in the feature space. The basic steps of the K - means algorithm are:
1. select an initial partition with K clusters containing randomly chosen samples, and compute the centroids of the clusters.
2. generate a new partition by assigning each sample to the closest cluster center.
3. compute new cluster centers as the centroids of the clusters.
4. repeat steps 2 and 3 until an optimum value of the criterion function is found (or until the cluster membership stabilizes).

2.2.2 Expectation Maximization

Expectation-maximization (EM)[32] is a method to find the maximum likelihood estimator of a parameter θ of a probability distribution.

The maximum likelihood estimate of θ maximizes p(y | θ), but in some cases this may be hard to find. That's when EM is useful as it takes your observed data y, iteratively makes guesses about the complete data x, and iteratively finds the θ that maximizes p(x | θ) over θ. In this way, EM tries to find the maximum likelihood estimate of θ given y.

The expectation maximization algorithm computes probabilities for each possible completion of the missing data. These probabilities are used to create a weighted training set consisting of all possible completions of the data. Finally, a modified version of maximum likelihood estimation that deals with weighted training examples provides new parameter estimates.
In other words, The EM iteration alternates between performing an expectation (E) step, which creates a function for the expectation of the log-likelihood evalu-

ated using the current estimate for the parameters, and a maximization (M) step, which computes parameters maximizing the expected log-likelihood found on the E step. These parameter-estimates are then used to determine the distribution of the latent variables in the next E step.

2.2.3 Spectral Clustering

The spectral clustering [33] consists in clustering data that is connected but not necessarily compact or clustered within convex boundaries.

The basic idea of spectral clustering consists of defining an Affinity matrix A in an Rn space, using a Gaussian kernel K or an Adjacency matrix (i.e. A {i,j }=Δ { i, j }).
After that, we construct the Graph Laplacian from A (i.e. decide on a normalization) and solve an Eigenvalue problem, such as:

$$Lv = \lambda v \tag{2.17}$$

(or a Generalized Eigenvalue problem L v=λ D v)
Once determined, we select k eigenvectors {v{i}, i=1, k } corresponding to the k lowest (or highest) eigenvalues λ{i}, i=1,..,k } , to define a k-dimensional subspace P^tLP and form clusters in this subspace using k-means

2.3 Image Segmentation based on Contour Detection

Feature extraction based on color and texture features is not the only approach to use to segment an image. In fact, there is the possibility to detect the contours in an image and then cluster the image using these extracted contours. The contours are predicted by calculating the posterior probability of a boundary at each pixel of the image at a given orientation θ and this by measuring the difference in local image brightness, color and texture channels. The goal is to calculate the gradient signal of the image. This gradient is computed by placing a circular disc at location (x, y) split into two half-discs by a diameter at angle θ. For each half-disc, we calculate the histogram of the intensity values of the pixels of I covered by it. The gradient magnitude G at location (x, y) is defined by the χ^2 distance between the two half-disc histograms g and h:

$$\chi^2(g,h) = \frac{1}{2}\sum_i \frac{(g(i)-h(i))^2}{(g(i)+h(i))} \tag{2.18}$$

Then a second-order Savitzky-Golay filtering is applied to enhance local maxima and smooth out multiple detection peaks in the direction orthogonal to θ.

$$mPb(x, y, \theta) = \sum_{s} \sum_{i} \alpha_{i,s} G_{i,\sigma(i,s)}(x, y, \theta) \tag{2.19}$$

The contour detection algorithm combines the oriented gradient signals obtained from transforming an input image into four separate feature channels and processing each channel independently. The first three correspond to the CIE lab color-space which are brightness, color a and color b channels. The fourth channel is a texture channel, in which we worked with Textons.
We then linearly combine these local cues into a single multi-scale oriented signal: where s indexes scales, we chose different scales to work with which are $[\sigma/2, \sigma, 2\sigma]$ for each brightness, color and texture, i indexes feature channels (brightness, color a, color b, texture), and $G_{i,\sigma(i,s)}(x, y, \theta)$ measures the oriented gradient of channel I between the position I of the two histograms of the disc of radius σ(i,s) at angle θ placed at position (x,y). Various values of Ï have been testes and finally σ=5 was chosen for brightness and σ=10 for color and texture. The parameter θ was chosen as the maximum of mPB with 8 different values of θ in the interval [0,pi] (we chose 8 equally spaced values of θ in [0,pi]).

$$mPb(x, y) = \max_{\theta} mPb(x, y, \theta) \tag{2.20}$$

After these steps, time to segment the image based on these cues. So the first step is to define the affinity matrix which in this case is defined by the exponential of the maximal value of mPb along a line connecting two pixels. We connect all pixels i and j within a fixed radius r .

$$W_{ij} = exp(-\max_{p \in \overline{ij}} mPb(p)/\rho) \tag{2.21}$$

where $\overline{ij}$ is the line connecting i and j and ρ is a constant. Then we compute:

$$D_{ii} = \sum_{j} W_{ij} \tag{2.22}$$

and then we solve the system.

$$(D - W)v = \lambda Dv \tag{2.23}$$

with v are the generalized eigenvectors $\{v_0, , v_n\}$ corresponding to the smallest n+1 eigenvalues $\lambda_0, \ldots, \lambda_n$. We work with sPb instead of mPb because mPb fires at all the edges while sPb extracts only the most salient curves in the image. SPb is defined as follow:

$$sPb(x, y, \theta) = \sum_{k=1} n \frac{1}{\lambda_k} . \nabla_{\theta} v_k(x, y) \tag{2.24}$$

where $\nabla_\theta v_k(x,y)$ is the convolution of the eigenvector v_k with Gaussian directional derivative filters at multiple orientations θ and λ_k is the k^{th} eigenvalue. At the end we have the globalized probability of boundary defined as follow:

$$gPb(x,y,\theta) = \sum_s \sum_i \beta_{i,s} G_{i,\sigma(i,s)}(x,y,\theta) + \gamma.sPb(x,y,\theta) \tag{2.25}$$

2.4 Multiple Instance Learning

The Multiple Instance Learning was first introduced by [Dietterich et al., 1997] in the context of drug activity prediction. In the Multiple Instance Learning task we learn a classifier based on a training set of bags. Compared to the standard supervised learning task in which we learn a classifier based on a training set of feature vectors where each feature vector is associated to a class label, a bag in a Multiple Instance Learning task contains many feature vectors called instances in the MIL terminology.

A bag is a set $X = \overrightarrow{x_1}, ..., \overrightarrow{x_N}$ where the elements $\overrightarrow{x_i}$ are feature vectors called instances, and the cardinality N can vary across the bags. All the instances live in a d-dimensional feature space, $\overrightarrow{x_i}$ called instance space.

The objective of MIL problem is to learn a model, at training time, which can be used to predict the class labels of unseen bags. In the classification level, we can consider binary classification or multiple classification. In a binary classification problem, a bag can be either positive or negative.
The objective is then to estimate a classification function $F(X) \in [0,1]$ that provides the likelihood that X is positive. In order to learn such a function, we are given a training set with M bags and their corresponding labels, $\tau = (X_1, y_1), ..., (X_M, y_M)$ is the label of X_i (y_i=0 if X_i is negative, and y_i=1 if it is positive). In a multiple classification of P classes, the bag labels y_i can take values between 1 and P.

In addition to the bag level classification function F(X), many methods try to learn an instance level classification function f(x_i) that operates directly on the instances xi. We use uppercase to refer to bags X and to the bag-level classifier F, and lowercase to refer to instances x and to the instance-level classifier f. The multiple instance classification methods can be categorized according to how the information existent in the MI data is exploited. We can separate the Instance Space paradigm, the Bag Space paradigm and the Embedded Space paradigm. In the Instance Space paradigm, the discriminative information is considered to lie at the instance-level. In fact, a discriminative instance level classifier f(x) is trained to separate the instances in positive bags from those in negative ones. Based on it, given a new bag X the bag level classifier F(X) is obtained by simply aggregating

instance level scores f(x).

This type of paradigm is based on local, instance-level information, in the sense that the learning process considers the characteristics of individual instances, without looking at more global characteristics of the whole bag.

In the Bag Space paradigm, the discriminative information is considered to lie at the bag-level. In this paradigm each bag X is treated as a whole entity, and the learning process discriminates between entire bags. As a result, it obtains a discriminative bag-level classifier F(X) which makes use of the information from the whole bag X in order to take a discriminative decision about the class of X.

This type of paradigm is based on global, bag-level information, because the discriminative decision is taken by looking at the whole bag, instead of aggregating local instance-level decisions.

Given the fact that the bag space is a non-vector space, the BS methods make use of non-vectorial learning techniques. As far as we know, all the existent non-vectorial techniques work through the definition of a distance function D(X,Y) that provides a way of comparing any two non-vectorial entries X and Y (where these entities are bags in our problem). Once this distance function has been defined, it can be used into any standard distance based classifier such as K Nearest Neighbor (K-NN), or similarity into any kernel-based classifier such as SVM.

In the Embedded Space paradigm, each bag X is mapped to a single feature vector which summarizes the relevant information about the whole bag X. As a result the original bag space is mapped to an embedded space of vectors where the discriminative classifier is learned. The ES paradigm is also considered as based on global, bag-level information, in the sense that the bag X is represented by a feature vector that summarizes the relevant information about the whole bag. The difference between the BS and the ES lies in the way the bag-level information is extracted. In the BS paradigm, this is done implicitly through the definition of a distance or kernel function. In contrast, in the ES paradigm, the extraction of information of the whole bag is performed explicitly through the definition of a mapping function that defines how the relevant information is represented into a single vector. Several theories and algorithms have been developed in this area.

2.4.1 Axis Parallel Rectangle

This algorithm was first developed by Dietterich at al. in the context of drug activity prediction. This algorithm assumes that a bag is considered positive if at least one of its instances is labeled positive and a bag is negative if all its instances are labeled negative. This method consists in expanding or shrinking a hyper rectangle

in the instance feature space in order to maximize the number of instances from different positive bags enclosed by the rectangle while minimizing the number of instances from negative bags inside the rectangle. A bag is classified positive if at least one of its instances is situated within the axis parallel rectangle and it is classified negative otherwise.

2.4.2 k Nearest Neighbors

The purpose of the k Nearest Neighbors (k-NN) algorithm is to use a training set in which data points or instances are separated into several separate classes to predict the classification of a new testing set. we consider each of the characteristics in our training set as a different dimension in some space, and take the value an observation has for this characteristic to be its coordinate in that dimension, so getting a set of points in space. We can then consider the similarity of two points to be the distance between them in this space under some appropriate metric. The way in which the algorithm decides which of the points from the training set are similar enough to be considered when choosing the class to predict for a new observation is to pick the k closest data points to the new observation, and to take the most common class among these. This is why it is called the k Nearest Neighbors algorithm.

2.4.3 Soft Margin Support Vector Machine

In machine learning, support vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis.
In this report, we will not focus on the definition of SVM but we will treat the mixed integer formulation of MIL as a generalized soft-margin SVM [38]. This latter can be written as follows:

$$\begin{aligned} mi-SVM: \quad & \min_{y_i} \min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & \forall i, y_i(< w, x_i > +b) \geq 1-\xi_i, \xi_i \geq 0, y_i \in \{-1,1\}. \end{aligned} \tag{2.26}$$

Notice that in the standard classification setting, the labels yi of training patterns xi would simply be given, while in (2.26) labels yi of patterns xi not belonging to any negative bag are treated as unknown integer variables. In mi-SVM one thus maximizes a soft-margin criterion jointly over possible label assignments as well as hyper planes.
The mi-SVM formulation leads to a mixed integer programming problem. One has to find both the optimal labeling and the optimal hyperplane. On a conceptual

level this mixed integer formulation captures exactly what MIL is about, i.e. to recover the unobserved instance labels and to simultaneously find an optimal discriminant.
An alternative way of applying maximum margin ideas to the MIL setting is to extend the notion of a margin from individual patterns to sets of patterns(instances). It is natural to define the functional margin of a bag with respect to a hyperplane by:

$$\gamma_i \equiv Y_I \max_{i \in I}(< w, x_i > +b) \tag{2.27}$$

This generalization reflects the fact that predictions for bag labels take the form:

$$\widehat{Y_I} = sgn \max_{i \in I}(< w, x_i > +b) \tag{2.28}$$

For a positive bag the margin is defined by the margin of the most positive instance, while the margin of the negative bag is defined by the least negative instance. We define the MIL version of the soft margin classifier by:

$$\begin{aligned} Mi - SVM: \quad & \min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_I \xi_I \\ \text{s.t.} \quad & \forall I, Y_I \max_{i \in I}(< w, x_i > +b) \geq 1 - \xi_i, \xi_i \geq 0. \end{aligned} \tag{2.29}$$

For negative bags one can unfold the max operation by introducing one inequality constraint per pattern, yet with a single slack variable ξ_I . Hence the constraints on negative bag patterns, where $Y_I = 1$, read as $-\langle w, x_i \rangle b \geq 1\xi_I$, $\forall i \in I$.
For positive bags, we introduce a selector variable $s(I) \in I$ which denotes the pattern selected as the positive "witness" in B_I . This will result in constraints $-\langle w, x_{s(i)} \rangle - b \geq 1 - \xi_I$. Thus we arrive at the following equivalent formulation:

$$\begin{aligned} & \min_s \min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_I \xi_I \\ \text{s.t.} \forall I, \quad & Y_I = -1 \wedge -\langle w, x_i \rangle - b \geq 1 - \xi_I, \forall i \in I, \\ or \quad & \mathrm{Y}_I = 1 \wedge \langle w, x_{s(I)} \rangle + b \geq 1 - \xi_I, \, and_i \geq 0. \end{aligned} \tag{2.30}$$

In this formulation, every positive bag B_I is thus effectively represented by a single member pattern $x_I \equiv x_{s(I)}$. Notice that "non-witness" patterns (x_i, $i \in I$ with $i \neq s(I)$) have no impact on the objective.

2.4.4 Bag of Features

The Bag of Feature is a new approach in computer vision that has been risen since the past decade. The Bag of Feature approach has several applications such as Object detection, image classification and image retrieval. The specificity of this

approach is that it can use orderless collection of image features. Lacking any structure or spatial information, it is perhaps surprising that this choice of image representation would be powerful enough to match or exceed state-of-the-art performance in many of the applications to which it has been applied. Due to its simplicity and performance, the Bag of Features approach has become well-established in the field.

There are two common perspectives for explaining the BoF image representation. The first is by analogy to the Bag of Words representation. With Bag of Words, one represents a document as a normalized histogram of word counts. Commonly, one counts all the words from a dictionary that appear in the document. This dictionary may exclude certain non informative words such as articles (like "the"), and it may have a single term to represent a set of synonyms. The term vector that represents the document is a sparse vector where each element is a term in the dictionary and the value of that element is the number of times the term appears in the document divided by the total number of dictionary words in the document (and thus, it is also a normalized histogram over the terms). The term vector is the Bag of Words document representation â called a "bag" because all ordering of the words in the document have been lost. The Bag of Features image representation is analogous. [23]

A visual vocabulary is constructed to represent the dictionary by clustering features extracted from a set of training images. The image features represent local areas of the image, just as words are local features of a document. Clustering is required so that a discrete vocabulary can be generated from millions (or billions) of local features sampled from the training data. Each feature cluster is a visual word. Given a novel image, features are detected and assigned to their nearest matching terms (cluster centers) from the visual vocabulary. The term vector is then simply the normalized histogram of the quantized features detected in the image. [24]

The second way to explain the BoF image representation is from a codebook perspective. Features are extracted from training images and vector quantized to develop a visual codebook. A novel image's features are assigned the nearest code in the codebook. The image is reduced to the set of codes it contains, represented as a histogram. The normalized histogram of codes is exactly the same as the normalized histogram of visual words, yet is motivated from a different point of view. Both 'codebook' and 'visual vocabulary' terminology is present in the surveyed literature. [23] At a high level, the procedure for generating a Bag of Features image representation is shown in Figure (2.8):

It can be summarized as follows:
(1) Build Vocabulary: Extract features from all images in a training set. Vector

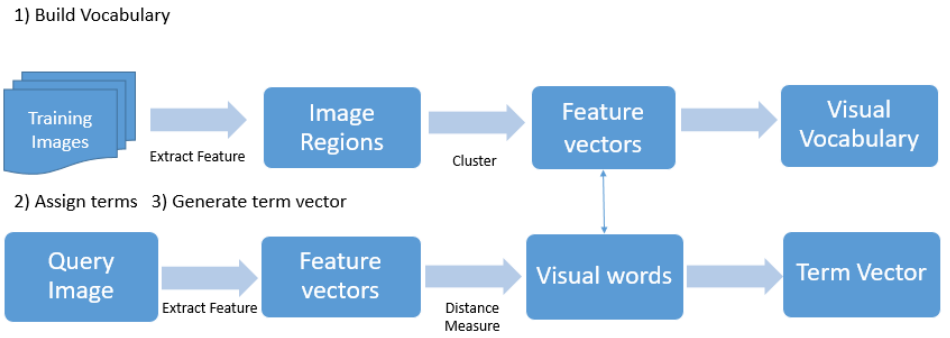


Figure 2.8: Bag of Feature Steps

quantize, or cluster, these features into a 'visual vocabulary,' where each cluster represents a 'visual word' or 'term.' In some works, the vocabulary is called the 'visual codebook.' Terms in the vocabulary are the codes in the codebook.
(2) Assign Terms: Extract features from a novel image. Use Nearest Neighbors or a related strategy to assign the features to the closest terms in the vocabulary.
(3) Generate Term Vector: Record the counts of each term that appears in the image to create a normalized histogram representing a "term vector." This term vector is the Bag of Features representation of the image. Term vectors may also be represented in ways other than simple term frequency.

3 Project Contribution

3.1 Dataset

Corel Corporation (from the abbreviation "Cowpland Research Laboratory") is a Canadian software company headquartered in Ottawa, Ontario, specializing in graphics processing. It is known for producing software titles such as CorelDRAW, and for acquiring PaintShop Pro, Video Studio and WordPerfect. Corel was founded by Michael Cowpland in 1985, as a research laboratory.
There exist two Corel Datasets. Corel- 5K Dataset and Corel- 10K Dataset. These datasets contain various categories of different subjects, each category contains 100 images. These datasets offer 5 resolutions from 128x192 up to 2048x3072 pixels.
In our work, we will be treating high resolution images (2048x3072 pixels). And we will be working on 15 different categories i.e. 1500 images. These categories are: "Beach", "Dogs", "Cars", "Birds", "Desert", "Elephant", "Waterfalls", "Ski", "Sunset", "Nature Scenes", "Lizards", "Glaciers", "Underwater", "Horses", "Flower".

Figure 3.1: Corel Dataset

3.2 Image Segmentation

The first goal of this internship is to provide a robust image segmentation and meaningful image extracted regions. For an efficient image segmentation, relevant features should be extracted. To do so, we have experienced with several feature extraction algorithms.

Image descriptors must possess certain properties to be practically effective for indexing and retrieval. Of prime importance is the ability of the descriptor to distinguish, with respect to some metric, visually different images. A second important, and competing, property is invariance to certain classes of image transformations such as rotation and translation. Other desirable properties include short descriptor length, noise insensitivity, computationally easy extraction, and descriptor scalability. The latter, in the context of MPEG-7 visual technology, means the ability to derive from a larger, more information-rich, higher performance descriptor a low performance one that requires less storage space.

Closely related to this is the notion of interoperability. Descriptors of the same kind but having, for example, differing lengths are said to be interoperable if such a mixed set of them can be intelligently handled by the query/retrieval mechanism. Mixed descriptors of this sort will be widely encountered in future internet search engine applications where a query descriptor may be applied to more than one database.

3.2.1 Feature Extraction

Color Features

First of all, we have experimented and analyzed several color features algorithms such as Color histogram, basic color features and Mean-sigma.
It is clear from the recipe for extraction that, neglecting image border effects, color histograms are invariant to translation and rotation. They are also relatively immune to low-level noise. Furthermore, schemes exist to make the color histogram both scalable and interoperable among descriptors having different numbers of bins and different amplitude quantizations. On the other hand, if two images have the same pixel count for each color, then they are indistinguishable based on this descriptor even when they are visually very different.
To overcome the insensitivity of the color histogram, we included in our analysis different methods such as color structure descriptor. We also experimented with combining different methods.
The color features can be very important in terms of detecting objects that have specific color such as detecting the sun or the sea. The color of these objects can

be a very good feature.

Texture and shape Features

In many cases, objects from the same category can have different colors. For instance, dogs can have different colors depending on the breed. In this case, color information is not sufficient. Texture features are thus necessary to learn the salient features of each category.
For texture features, we experimented with few different methods including Textons[14], Local Binary Patterns[19], Edge Histogram Descriptor[29] and Histogram of oriented Gradient[34].

For our work, we thought of combining color and texture features algorithms to get more meaningful features.
We also chose to window all the algorithms we are working on, which means, for a given window size w we apply the algorithms at each window at each pixel of the image with window size w. We did that to take into consideration the information given by each neighborhood of a given pixel and combine it with the information of the pixel so that the features become more coherent. We tried several combinations of feature extraction algorithms. We also added the position X and Y of all the pixels as features to help detect pixels corresponding to the same compact object.

3.2.2 Image Clustering

Clustering is the fact to divide the image into clusters or regions using the extracted features of the image. We have tried three clustering algorithms. K-means, Expectation maximization and Spectral Clustering. We have noticed that K-means is the fastest algorithm among these three but Spectral Clustering is the most efficient.

3.2.3 Image Segmentation based on Contour Detection

The contour detection based image segmentation consists in predicting the contours of an image by calculating the probability of boundaries at each pixel at a given orientation θ. The gradient is calculated by measuring local image brightness, color and texture channels. This gradient is computed by placing a circular disc split into two half discs by a diameter at angle θ and then we calculate the histogram of the values of the pixels covered by each half disc. Finally we calculate the distance between the two half discs[28].

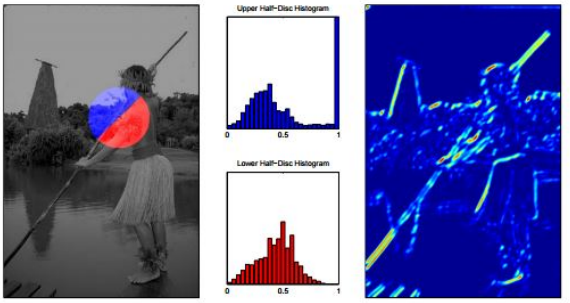


Figure 3.2: Gradient calculation using circular disc [28]

The next step is to apply the Watershed Transform (WT), to get image regions from contour detection. So we consider E(x,y,θ) the probability of an image boundary at position (x,y) and orientation θ . Then we determine the maximum E(x,y) over different orientations θ. We take the regional minima of E(x,y) as seed locations for homogeneous segments and apply the watershed transform used in mathematical morphology on the topographic surface defined by E(x, y). Then we build the Ultrametric Contour Map (UCM) from the boundaries of these initial regions. So we define G = (P0 , K0 , W (K0)), where the nodes are the regions P0 , the links are the arcs K0 separating adjacent regions, and the weights W (K0) are a measure of dissimilarity between regions. The algorithm proceeds by sorting the links by similarity and iteratively merging the most similar regions

This process produces a tree of regions, where the leaves are the initial elements of P 0, the root is the entire image, and the regions are ordered by the inclusion relation. We can also define H(R) = W (C) where C is the contour whose removal formed R and H(R) is the height of each region R is the height of each region R is the value of the dissimilarity at which it first appears. Then a dendrogram is defined as follow:

$$D(R1, R2) = min\{H(R) : R1, R2 \subseteq R\} \tag{3.1}$$

This is a convenient representation of the region tree since the segmentation at a scale k can be easily retrieved by thresholding the UCM at level k.

3.3 Experimental Results of Image Segmentation

3.3.1 Qualitative Evaluation

Image segmentation using combination of features

Here is a set of experimental results for different combinations. For the clustering in these experiments we used K-means.

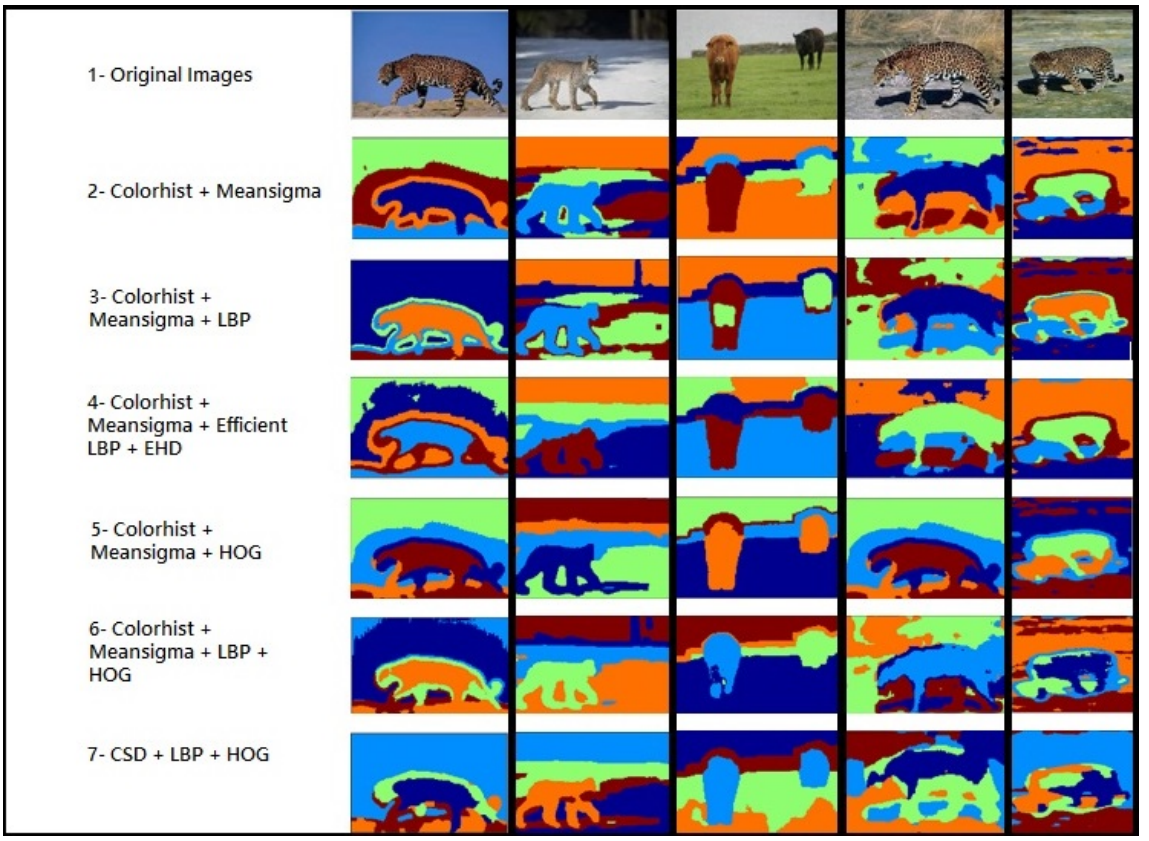


Figure 3.3: Results of the segmentation in arbitrary colors

This figure (3.3) shows the results of the segmentations with different feature combinations for five images in arbitrary colors.
After these experiments, we can notice that the combination of Colorhist and LBP algorithms are more likely to give good results. The use of Hog feature also has shown to be useful and so is the case of Mean-sigma.

One of the major difficulties of Image annotation is the choice of the features. In fact, the goal of this work is to handle a large database so the choice of features

should be as optimal as possible since it should be relevant and a good representative for the different classes and at the same time, not very large so not time consuming.That's why we have been trying several evaluation tests and methods.

We can notice that even the features Color-hist and Mean-sigma, HOG and LBP have very high correlations, they together give better results.

Image segmentation using Contour Detection Method

The second part of the project deals with the image segmentation based on contour detection. Here is some segmentation results using contour detection method (figure (3.4)). The first column is the original image and the rest are the extracted regions after segmentation.

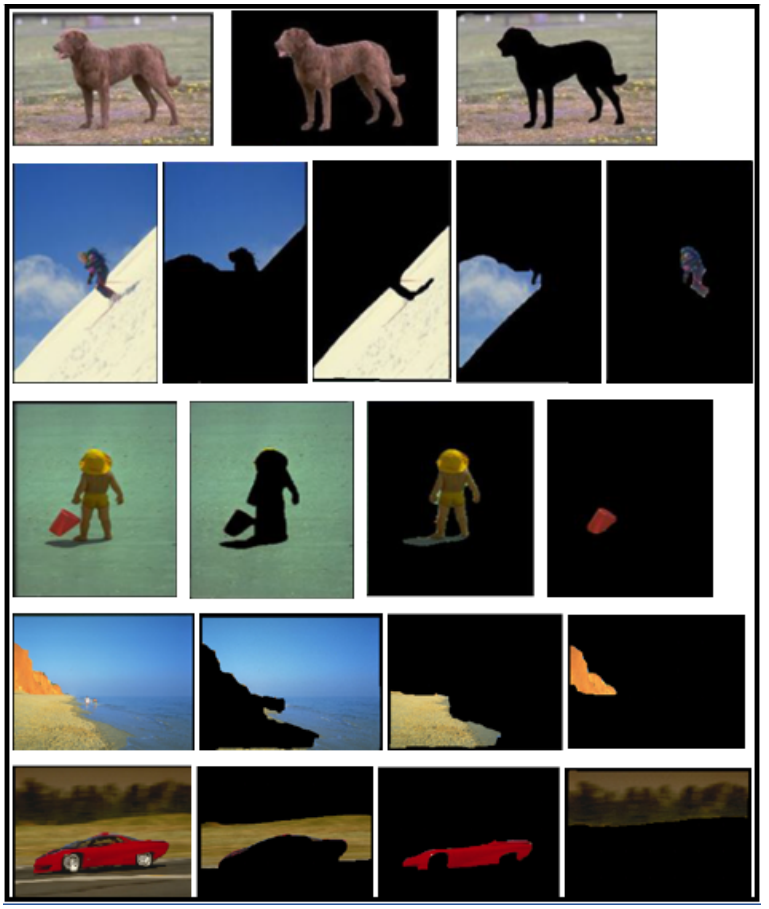


Figure 3.4: Results of the segmentation using Contour Detection algorithm

By comparing the segmentation results of the two methods, we can guess that the

second method gives better results. These results need to be verified by a quantitative study that we are going to treat in the following subsection.

3.3.2 Quantitative Evaluation

To make sure of the right combination, we have used an image segmentation evaluation algorithm proposed by Berkeley. This algorithm consists of comparing the similarity between the segmented image (by the tested segmentation algorithm) and the ground truth of the image which means a manually segmented image in which the objects are detected visually by a human being. The results are given in terms of Variation of Information VOI and Random Index RI. The VOI is generally used to compare clustering methods. It in fact computes the distance between two segmentations in terms of their average conditional entropy given by:

$$VI(S,S) = H(S) + H(S) - 2I(S,S) \tag{3.2}$$

where H and I represent respectively the entropies and mutual information between two clusterings of data S and S ′ In our case, these clusterings are test and ground truth segmentations. Obviously, the best the results are, the lower the value of VOI is .The Rand Index computes the compatibility and the similarity between two images. In our case it is given by the sum of the number of pairs of pixels that have the same label in the segmented image and the ground truth image and those that have different labels in both segmentations, divided by the total number of pairs of pixels. Given a set of ground truth segmentations G , k , the Probabilistic Rand Index is defined as:

$$PRI(S,\{Gk\}) = \frac{1}{T}\sum_{i\leq j}[c_{ij}p_{ij} + (1-c_{ij})(1-p_{ij})] \tag{3.3}$$

where c_{ij} is the event that pixels i and j have the same label and p ij its probability. T is the total number of pixel pairs.

So first of all we fixed the parameters which are the window size w=3 and the number of clusters C=5 and we chose K-means as the clustering algorithm. We tried different combination of features and got these results:

After experimenting several combination, we can conclude that combining Colorhist, Mean-sigma, LBP and HOG has shown to be the most efficient. Now we need to fix the parameters, so we varied the window size and the number of clusters and displayed the results:

Combination	RI	VOI
Colorhist + Meansigma	0.55	2.7
Colorhist + CSD	0.4	2.8
CSD + Meansigma	0.38	2.85
LBP + HOG	0.56	2.5
LBP + EHD	0.6	2.58
EHD + LBP	0.54	2.4
Colorhist + Meansigma+ LBP	0.72	2.58
Colorhist + Meansigma+ HOG	0.74	2.33
Colorhist + Meansigma + LBP + HOG	**0.77**	**2.24**
Colorhist + Meansigma + Efficient LBP + HOG	0.76	2.26

Figure 3.5: Different Combination RI and VOI Indices

Window size	RI	VOI
W=3	**0.77**	**2.24**
W=4	0.73	2.55
W=5	0.7	2.63
W=10	0.68	2.67

Figure 3.6: Varying the window size results

Number of Clusters	RI	VOI
C=4	0.75	2.4
C=5	**0.77**	**2.24**
C=6	0.76	2.32
C=10	0.52	2.46

Figure 3.7: Varying the number of clusters results

As shown in the previous two figures (figures (3.6) and (3.7)) the optimal parameters are: w=3 and C=5.
At this stage, we need to decide whether to do the segmentation with the combination of feature extraction algorithms that we explained earlier or with the contour detection algorithm. To choose one option, we opted for another evaluation test to compare the performance of the two codes (figure (3.9)). For this, we chose one image from each class (10 classes of Corel Dataset). We created the groundtruth of these images and applied the evaluation on both codes.

Groundtruth Images:

To calculate these indexes, we need handmade segmentation to use it as groundtruth. So we have worked on segmenting few pictures from all the classes using LabelMe annotation tool[35].
Below in figure (3.8) are examples of groundtruth images drawn by handmade segmentation using Labelme Segmentation Tool[35].

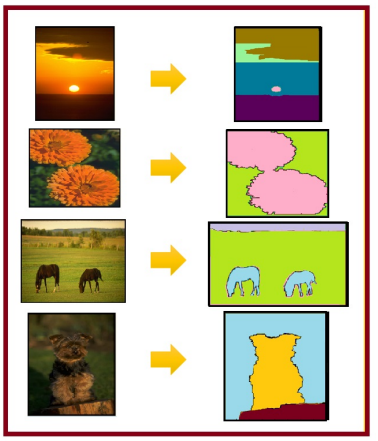


Figure 3.8: Example of Ground-truth images as well as the original images

Below in figure (3.9) are the results of the two methods evaluation:

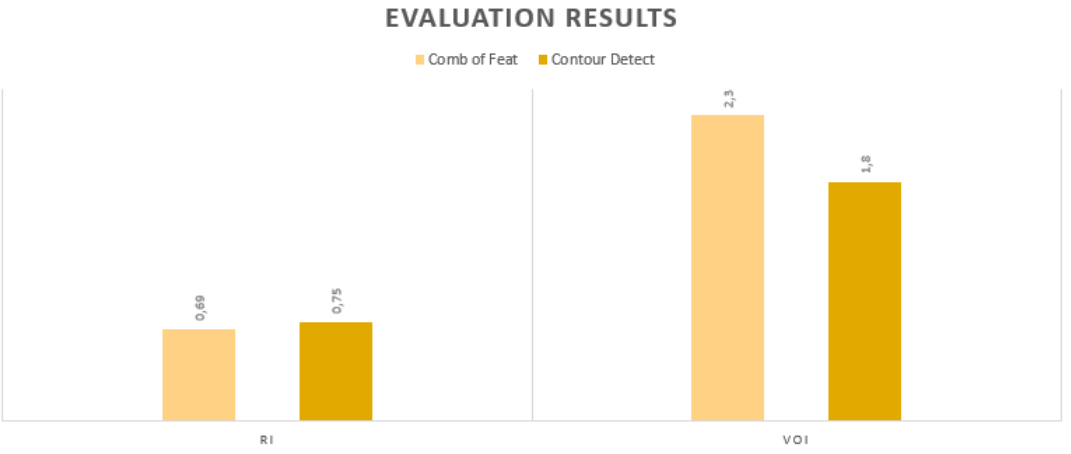


Figure 3.9: Comparison of the two methods results

It is clear that The Contour detection algorithm gave better results compared to the combination of features explained earlier.

So to sum up, the main goal of this work is to obtain meaningful image regions that we're going to use as instances and extract features from those instances to work on a Multiple Instance Learning problem. For the segmentation of the images, we decided to work with the contour detection algorithm since it has better results as mentioned above. The segmentation will produce image regions that we need to use as instances and to extract features from them. So, we decided to extract those features using the combination of feature extraction algorithm that we have been working on in the first part of the internship.

In our case, we chose the feature extraction results to be tested and judged good or bad, relevant or not by the human eye decision. That's why the choice of the features should be based on human decision since the combination of features resulted from the evaluation algorithms may not give good results from the point of view of the human. The main idea was to choose the combination of features that give more logical results from the point of view of the tester. To do so, we have tried several combinations of features. After segmentation, for a given image-region, and for each combination of features that we have tested in the previous section, we tried to see the image regions that are the closest to the reference image-region. For a given image region, we calculated the Euclidean distance of all the feature vectors of the image regions that we have and find the 20 image regions that give the minimal distances to the feature vector of the chosen image-region. Below in figures (3.10) and (3.11) are two examples of 20 image regions as well as their Euclidean distances to the chosen image-region.

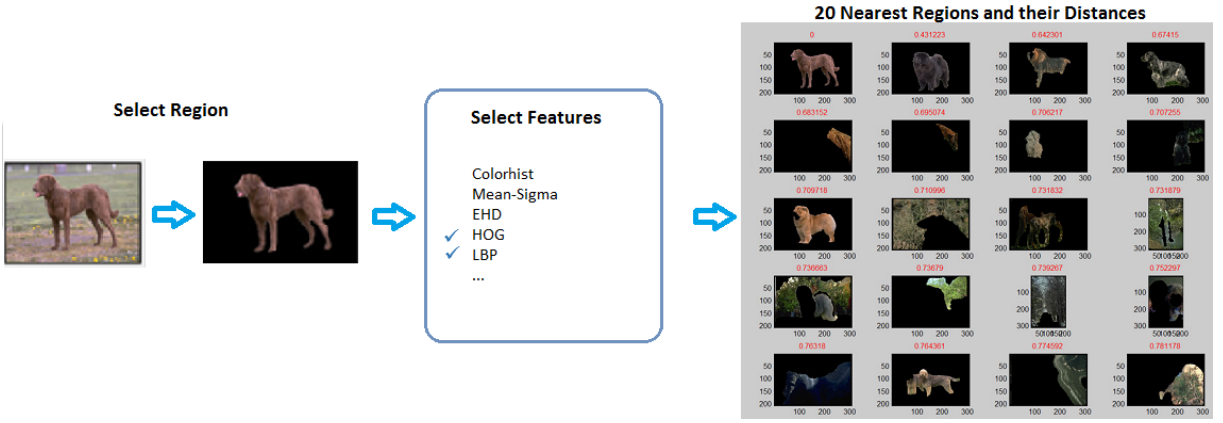


Figure 3.10: Example of Feature Testing

We can see that the combination of the HOG feature and the LBP feature is very performant in terms of shape and texture detection, we can see that the region tested in figure (3.10) consists of a 'dog'. Since dogs may have different colors the main feature that should be relied on is the texture and shape features that's why

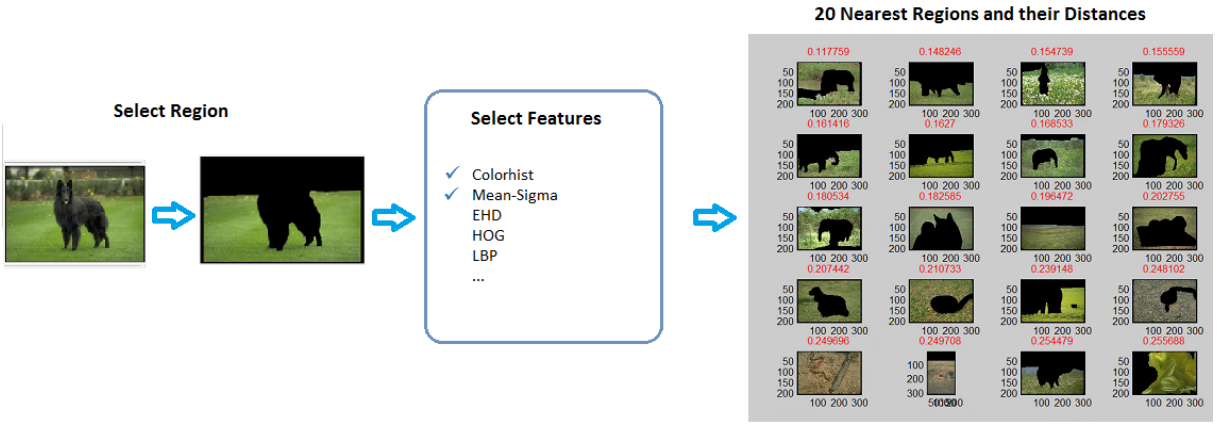


Figure 3.11: Example 2 of Feature Testing

the combination of HOG and LBP is very efficient for this type of regions.
In figure (3.11), the region consists of 'grass' that's why the color features should be relied on in this case. We can see through this test that the combination of Colorhist and Meansigma is very efficient for this image region as all the 20 nearest image regions detected represent 'grass'.
We can conclude that the combination of these four features should be very efficient as it covers the color, shape and texture features of a given image.

3.4 Image Annotation

3.4.1 Multiple Instance Learning

During our work, we have been working on a Multiple Instance Learning problem in which we defined each image as a bag containing instances that are in our case the regions that we extracted from the image segmentation that we have been treating in the previous sections. The bag labels are the classes of the images. So now we have a Multiple Instance Learning case in which we only know the labels of the bags in both training and testing data. And what we are trying to achieve is to learn models using the training data that will help us predict labels of both bags and instances of the testing data.

For the classifiers, we have tried several algorithms and then settled for mi-SVM. We opted for this choice because in later work we are trying to predict instance labels so we need instance space algorithm. With reference to several papers [36][37], mi-SVM seems to be one of the most robust and efficient algorithm that does not

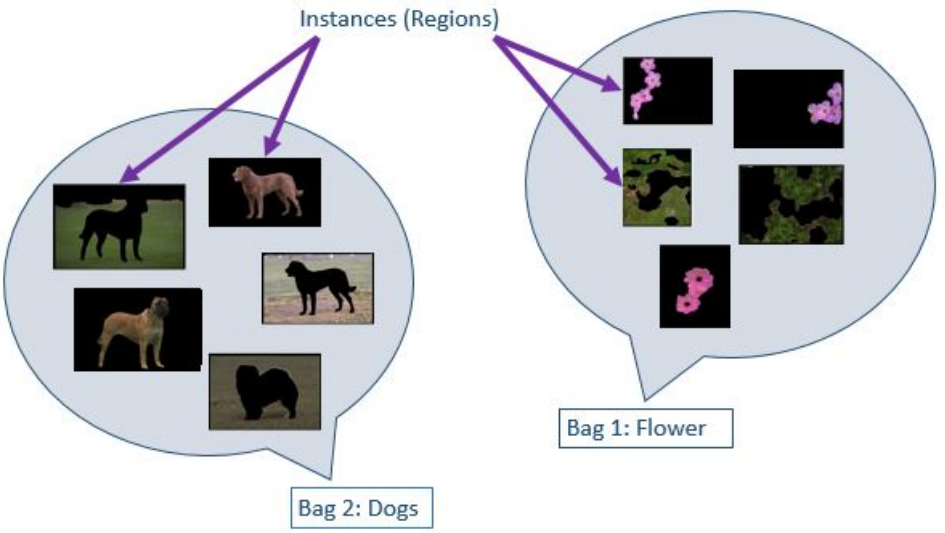


Figure 3.12: Multiple Instance Learning Example

take too much time processing .

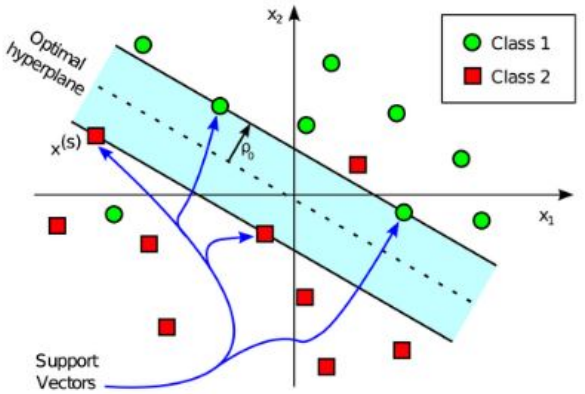


Figure 3.13: mi-SVM Classifier Building Model

The classifier is supposed to find a model that separates the relevant regions of each class. As shown in the figure above (3.13), if a region does not represent its class it will be considered as non representative region and it will get different label. In this problem in figure (3.13) the goal is to separate the two classes by a function that is induced from available examples. The goal is to produce a classifier that will work well on unseen examples, that is, it generalizes well.The blue area in the figure (3.13) is the margin defined by the Support Vector Machine. The main idea is that the decision boundary should be as far away as possible from the data points of both classes. There is only one that maximizes the blue margin (maximizes

the distance between it and the nearest data point of each class). Intuitively, the margin is defined as the amount of space, or separation between the two classes as defined by the hyperplane. Geometrically, the margin corresponds to the shortest distance between the closest data points to a point on the hyperplane.
We tried different parameters and get various results. The goal is to compare our results using our feature vectors to the results found by the method proposed in [22].

3.4.2 Dataset enlargement

So far we have been working on a dataset of 10 classes. Our goal is to enlarge the dataset so we will be working on 15 different classes which are : 'Beach', 'Cars', 'Birds' , 'Dogs', 'Desert', 'Elephant', 'Waterfalls', 'Ski', 'Sunset', 'Nature Scenes', 'Lizards', 'Glaciers', 'Underwater', 'Horses', ' Flower'.

Scale Invariant Feature Transform

Scale-invariant feature transform (or SIFT) is an algorithm in computer vision to detect and describe local features in images. It, in fact, transforms image data into scale-invariant coordinates relative to local features. The algorithm that detects key-points and calculate the SIFT descriptor can be summarized in these 4 steps [25]:
1. Scale-space extrema detection: The first stage of computation searches over all scales and image locations. It is implemented efficiently by using a difference-of-Gaussian function that is used to approximate the normalized Laplacian of Gaussian (LoG) filter to identify potential interest points that are invariant to scale and orientation.
2. Key point localization: At each candidate location, a detailed model is fit to determine location and scale. Key points are selected based on measures of their stability.
3. Orientation assignment: One or more orientations are assigned to each key point location based on local image gradient directions. All future operations are performed on image data that has been transformed relative to the assigned orientation, scale, and location for each feature, thereby providing invariance to these transformations.
4. Key point descriptor: The local image gradients are measured at the selected scale in the region around each key point. These are transformed into a representation that allows for significant levels of local shape distortion and change in illumination.

Bag of Features

At a high level, the procedure for generating a Bag of Features image representation can be summarized as follows:

(1) Build Vocabulary: Extract features from all images in a training set. Vector quantize, or cluster, these features into a ”visual vocabulary,” where each cluster represents a ”visual word” or ”term.” In some works, the vocabulary is called the ”visual codebook.” Terms in the vocabulary are the codes in the codebook.

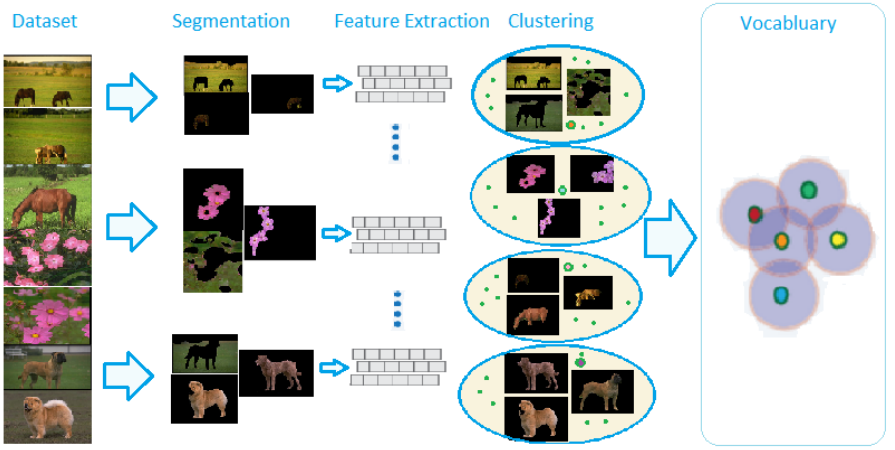


Figure 3.14: Vocabulary Creation Steps

The visual vocabulary, or bag of features, is created by extracting feature descriptors from representative images of each category. The Bag of Features object defines the features, or visual words, by using the k-means clustering algorithm on the feature descriptors extracted from training sets. The algorithm iteratively groups the descriptors into k mutually exclusive clusters. The resulting clusters are compact and separated by similar characteristics. Each cluster centroid represents a feature, or visual word. It is worth noticing that generating vocabulary is done off-line and may be time consuming since the amount of local descriptors could be huge.
(2) Assign Terms: Extract features from a novel image. Use Nearest Neighbors or a related strategy to assign the features to the closest terms in the vocabulary.
(3) Generate Term Vector: Record the counts of each term that appears in the image to create a normalized histogram representing a ”term vector.” This term vector is the Bag of Features representation of the image. Term vectors may also be represented in ways other than simple term frequency.

We use this method to encode each image from the training set. This function detects and extracts features from the image and then uses the approximate nearest

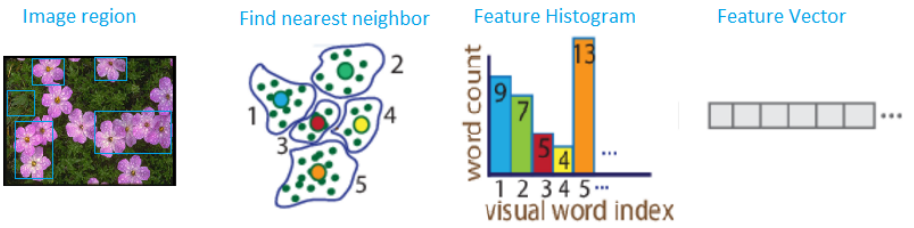


Figure 3.15: Bag of Feature Histogram Calculation steps

neighbor algorithm to construct a feature histogram for each image. The function then increments histogram bins based on the proximity of the descriptor to a particular cluster center. The histogram length corresponds to the number of visual words that the Bag of Features object constructed. The histogram becomes a feature vector for the image.

Bag of Feature Using Region-Level SIFT

In this part, we tried to apply the Bag of Features approach on region-level SIFT features. The method can be summarized in the following steps:

(1) Extract the different regions by segmenting the images by using the contour detection algorithm.
(2) Extract SIFT features from each image region. We obtain several feature vectors of the different interest points of the region by extrema detection, key point localization and orientation assignment.
(3) Use the Bag of Features object to define the visual words, by using the k-means clustering algorithm on the feature descriptors extracted from training sets to define the clusters centroids.
(4) Use the approximate nearest neighbor algorithm to construct feature histogram for each region.
Once we obtain the feature vectors for each region, we tried to combine them with the feature vectors that we obtained from the proposed method that we have been working on in the previous work in this project. We tried this method with both SIFT and Dense SIFT and compared the results.

SIFT vs Dense SIFT

The obvious difference between SIFT and Dense SIFT is that with dense SIFT you get a SIFT descriptor at every location, while with normal sift you get a SIFT

descriptions at the locations determined by Lowe's algorithm [25].
As a matter of fact, SIFT identifies interest points using Difference of Gaussian Filtering (DoG) before using Histogram of Oriented Gradients to describe these interest points, however Dense-SIFT does not identify interest points, it simply divides the image into overlapping cells before using HOG to describe them. Since they both use HOG they both produce 128 dimensional feature vectors.

3.5 Experimental Results of Image Annotation

To evaluate our method, we will try to calculate some evaluation scores for the label prediction mainly the accuracy. Since we know the labels of the bags but we ignore the true labels of the instances, we will create groundtruth images in which we will assign a label to each region. We created 20 groundtruth images per class. In figure (3.16) 3 examples of image groundtruth used to test the performance of our algorithm. The aim of this section is to compare the results of our methods

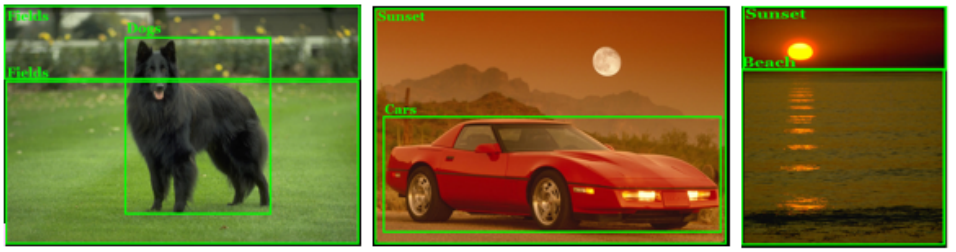


Figure 3.16: Groundtruth for MIL testing

with the results of the method proposed in [22].In fact, Chen et al. Describe in their paper a different combination of features that they used for the MIL problem that they have treated. So the first thing to do is to fix the parameters of SVM that we are going to work with. So we worked on one class and varied the parameters and get these results in figure (3.17).

Parameters	Kernel= RBF Cost Factor = 20	Kernel= RBF Cost Factor = 25	Kernel= Polynomial Degree=3 Cost Factor = 20	Kernel= Polynomial Degree=3 Cost Factor = 25
Bag Accuracy for the Class 'Beach'	0. 815	0.84	0.84	0.845

Figure 3.17: Bag accuracies for different parameters of SVM

Once we fixed the parameters, we tested our algorithm and found the results below(figure (3.18)) of Bag Accuracy and Instance Accuracy using our method with the parameters Kernel= Polynomial, Degree=3 and Cost Factor = 25:

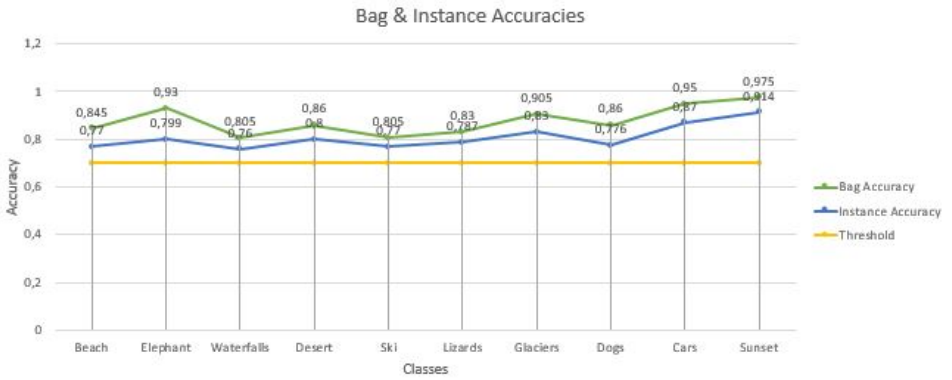


Figure 3.18: Bag/ Instance accuracies Results

We also visualized the results of the annotation by displaying the labels on the original images. Here are some results after annotation:

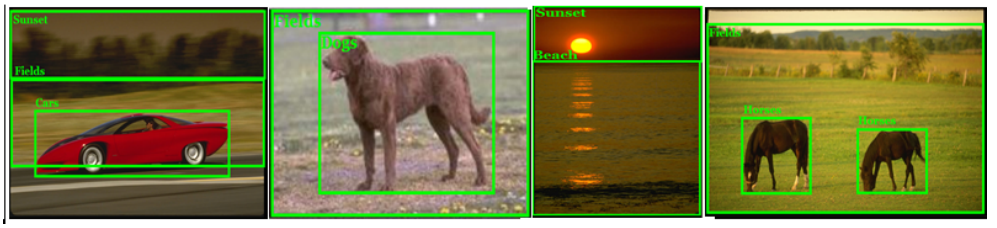


Figure 3.19: Annotation Results Examples

As shown in this figure (3.19), these images' annotation was accurate.
As for the computation time for testing one image that is new to the system, suppose the image is 3072x2048 pixels (as in Corel), the segmentation takes around 5 seconds, the feature extraction takes 1 second and the annotation process takes 0.1 seconds.

To make sure that our feature combination is the one that gives best results, we tried testing several combinations and looked at the bag accuracy and Instance Accuracy of the class Sunset (figure (3.20)):

So the best result was found by combining Colorhist, LBP, Meansigma and Hog which confirms the results found in our previous work of testing the features. For a fair comparison, we worked with the same classes as in [22] and we displayed

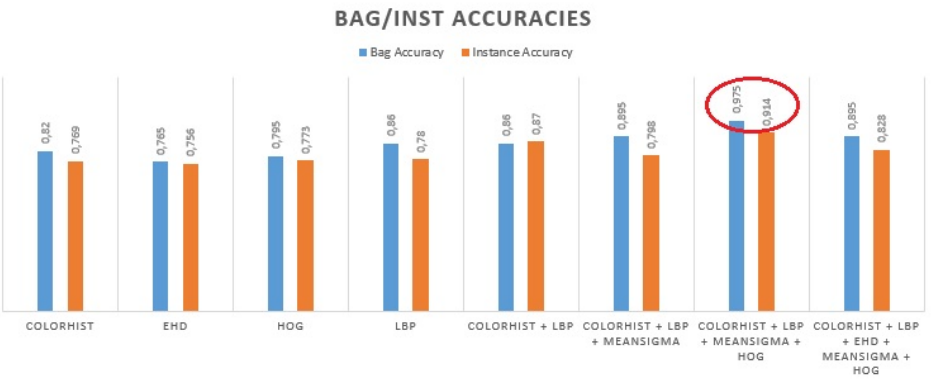
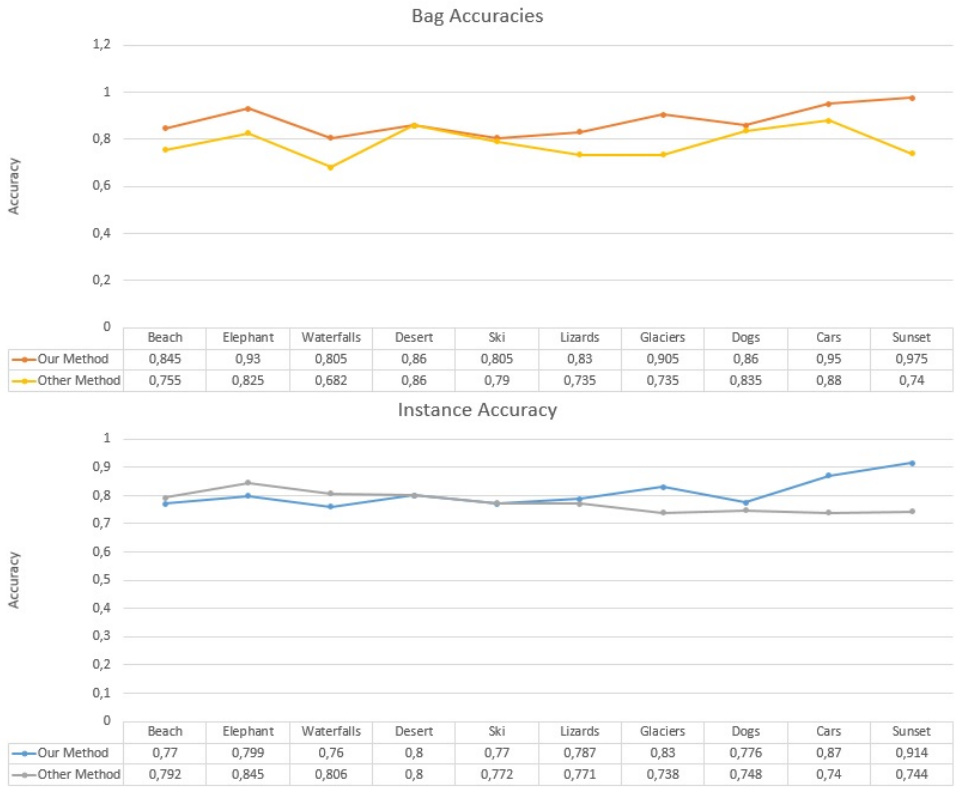


Figure 3.20: Bag/ Instance accuracies for various features for class Sunset

the results of the algorithm proposed in [22] with the same parameters and the same classes. Below are the results of the algorithm proposed in [22]. Clearly, the method proposed in this project gave better results than the one proposed in [22] and shown in figure (3.21) given a comparable computation time.



	Beach	Elephant	Waterfalls	Desert	Ski	Lizards	Glaciers	Dogs	Cars	Sunset
Our Method	0,845	0,93	0,805	0,86	0,805	0,83	0,905	0,86	0,95	0,975
Other Method	0,755	0,825	0,682	0,86	0,79	0,735	0,735	0,835	0,88	0,74

	Beach	Elephant	Waterfalls	Desert	Ski	Lizards	Glaciers	Dogs	Cars	Sunset
Our Method	0,77	0,799	0,76	0,8	0,77	0,787	0,83	0,776	0,87	0,914
Other Method	0,792	0,845	0,806	0,8	0,772	0,771	0,738	0,748	0,74	0,744

Figure 3.21: Comparison with the results of the features proposed in [22]

The next part of the work was about enlarging the dataset that we are working on. So after enlarging the database, we tried to calculate the accuracies of the label prediction using the algorithm that we have been working on. We also calculated the precision and recall for each class. The results were acceptable with a minimum accuracy of 0.8866 and a maximum accuracy of 0.9478(See figure (3.22)). The region labels were overall accurate.

Class	Accuracy	Precision	Recall
Beach	0.9223	0.719	0.9587
Cars	0.9191	0.850	0.9420
Desert	0.9040	0.528	0.9467
Dogs	0.8866	0.684	0.9235
Elephant	0.9019	0.778	0.9309
Flower	0.9321	0.978	0.9666
Glaciers	0.9191	0.819	0.9508
Horses	0.9261	0.796	0.9703
Lizards	0.9156	0.598	0.9552
Ski	0.9219	0.711	0.9621
Sunset	0.94785	0.938	0.9734
Waterfalls	0.9069	0.967	0.9554
Birds	0.9114	0.769	0.9692
Fields	0.9191	0.8102	0.9710
Nature	0.8928	0.508	0.9424

Figure 3.22: Evaluation of Instance Results for 15 classes

Then we tried with the second method that we explained earlier in the report which consists in combining the bag of feature vector of SIFT features with the obtained features of the first part of the report. In the instance level, we worked with SIFT and Dense SIFT. The processing time of SIFT features extraction and Bag of Features is around 1 second so The time difference between the first and the second method is insignificant. The results are as follow (figure (3.23)):

Here in figure (3.24) is a comparison between the two methods proposed. The annotation of the images from 15 classes with the combination of features before and after adding the Bag of Feature approach.

For the bag level, we tried the combination of the proposed feature with SIFT and with Dense SIFT (figure (3.25)).
In the bag level, Dense SIFT gave better results that SIFT. This can be explained by the fact that with the chosen threshold, Dense SIFT takes into consideration more keypoints than the SIFT and then gives more precision than SIFT.

Class	Accuracy	Precision	Recall
1-Beach	0.9427	0.92	0.9657
2-Cars	0.9293	0.76	0.9529
3-Desert	0.8973	0.68	0.9486
4-Dogs	0.9013	0.74	0.9371
5-Elephant	0.9053	0.82	0.9257
6-Flower	0.9427	0.86	0.9671
7-Glaciers	0.928	0.86	0.9514
8-Horses	0.9360	0.744	0.9714
9-Lizards	0.9347	0.868	0.9529
10-Ski	0.9347	0.948	0.9671
11-Sunset	0.9547	0.856	0.9829
12-Waterfalls	0.9187	0.82	0.9686
13-Birds	0.9227	0.812	0.98
14-Fields	0.912	0.812	0.9686
15-Nature	0.9013	0.92	0.95

Figure 3.23: Same Evaluation Results for the second method with BoF and SIFT

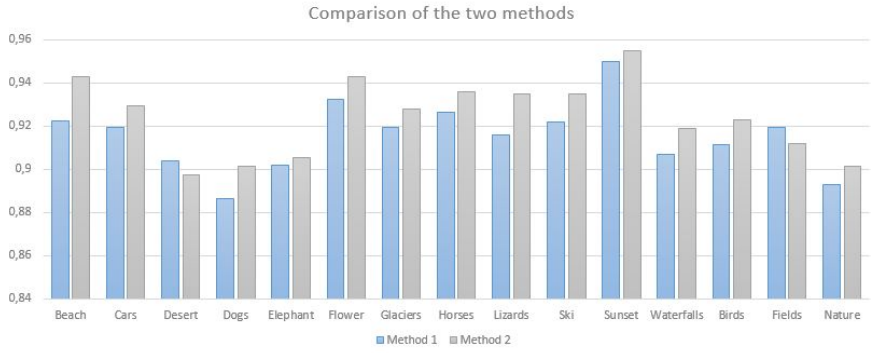


Figure 3.24: Comparison of the accuracies between the first and the second method with BoF and SIFT

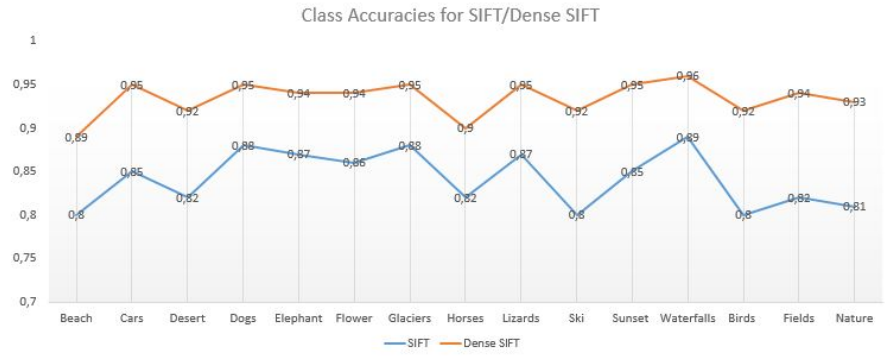


Figure 3.25: Accuracies for all classes in SIFT and Dense SIFT

3.6 Graphical User Interface

To sum up, our method gives satisfying accuracies for both bag level and instance level. To display and interpret our results at each step, we worked on different GUIs.

First, for the segmentation using a combination of features, we developed a GUI (figure (3.26)) in which you can specify the features you want to extract, the parameters such as the window size and the number of clusters as well as the specific parameters for each feature.
For each feature you have the possibility to assign a weight. You can also choose the clustering algorithm. Then, you simply choose the image you want to segment from the library of images and then you click on 'work' to display the segmentation.

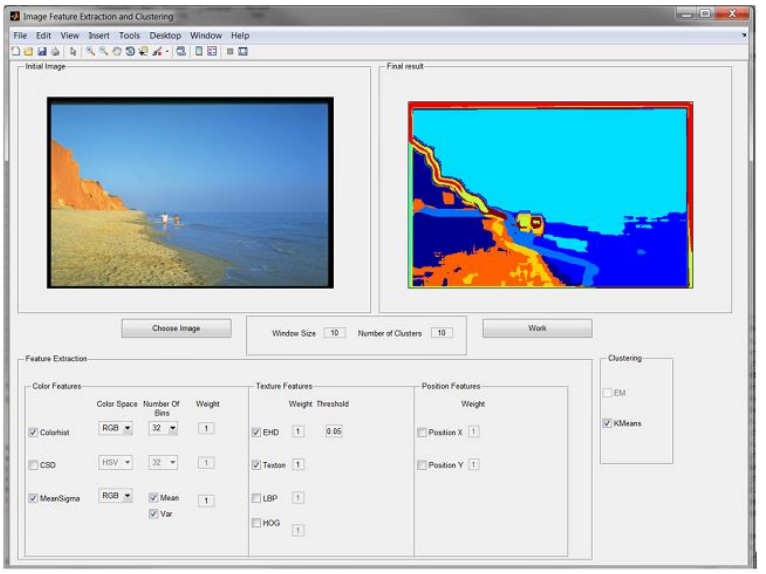


Figure 3.26: Gui for Image Segmentation

This GUI allows easier way to try different combinations of features and to display the results as well as the parameters to permit better visual interpretation.

The second GUI was developed after applying the Multiple Instance Learning algorithm. In fact, we calculate the probability of the instance labels and then display the image, its segmentation and the two regions that have the highest probability as well as the truth labels of these two regions in Figure (3.27).

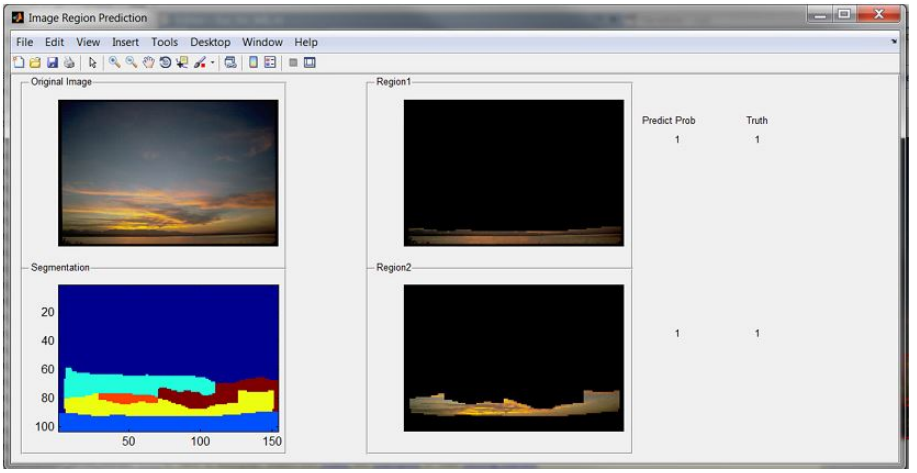


Figure 3.27: Gui 1 for Image Annotation

Then, we displayed the image and the regions that have the highest probabilities as well as their probabilities in Figure (3.28) left.

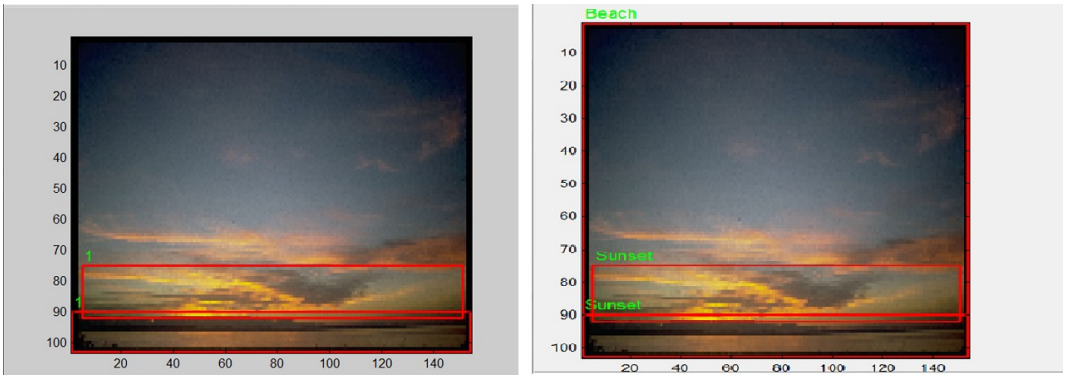


Figure 3.28: Image Annotation Results

As shown in the figure (3.28) three regions were detected. In the first image on the left, the probability of labeling is shown. As for the image on the right, the labels of the regions marked by the red rectangles are displayed on the original image. The three labels assigned for this image in the three regions displayed in the figure (3.28) are: "Beach", "Sunset" and "Sunset".
In fact, the way we assign the region labels is that we test each region in all the created models and we compare the probability of the instance, and then we display the label of the model that gave the highest probability.

Finally, we have worked on a GUI that summarizes all the steps in Figure (3.29)

. First you choose the image that you want to test, then you segment it using the contour detection algorithm. After that, you extract the features from the regions, load the models of the classes and then test the features and display the results.

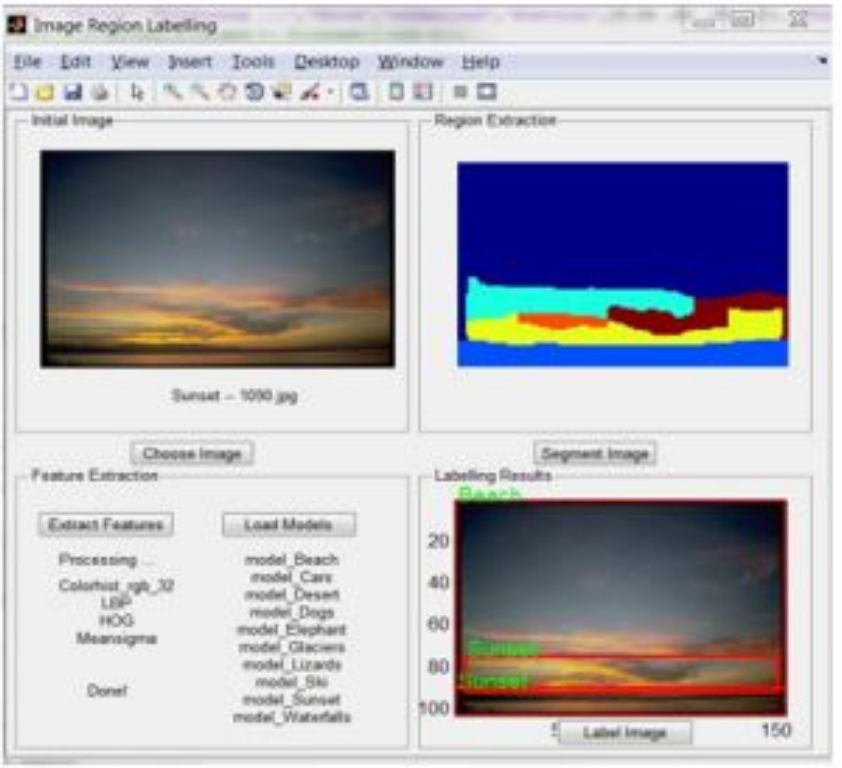


Figure 3.29: Gui 2 for Image Annotation

4 Conclusion

In this report, we covered the different methods that we have been working on to realize efficient image annotation algorithm based on class-level labeled data. In fact, we have applied Multiple Instance Learning techniques to learn models. these models were used to predict instance-level labels of unseen images.

We first started by the segmentation phase in which we tried several options and then opted for segmentation using image contour detection. After that, we dealt with the feature extraction phase in which we opted for a combination of features that we judged to be the most efficient among the tested other combinations. As for the classifier, we worked with Soft Margin Support Vector Machine.

We compared our results to other existent work and found satisfying results. We then worked on enlarging the dataset and measure the evaluation scores. We then applied a bag of feature approach to enhance the accuracies. We also included the Dense SIFT feature. We got better evaluation scores.
However, we noticed that the instance label are in some cases influenced by the class label. For example, an instance in a class labeled "Beach" can be sometimes more likely to be labeled as "Beach" which is not always the fact. In fact, the mi-SVM is efficient but in some cases it fails to discriminate some data points into the right class.
Such a problem is going to be our next challenge as we will try to improve the labeling performance as well as conserving the low correlation between class and instance labels. content

References

[1] K. Barnard, Q. Fan, R. Swaminathan, A. Hoogs, R. Collins, P. Rondot, and J. Kaufhold. "Evaluation of Localized Semantics: Data, Methodology, and Experiments", International Journal of Computer Vision, Vol.77 (1-3), pp. 199-217, 2008.

[2] H. J. Escalante, M. Montes, and L. E. Sucar. "Word Co-occurrence and Markov Random Fields for Improving Automatic Image Annotation", Proceedings of the 18th British Machine Vision Conference, Vol. 2, pp. 600-609, Warwick, UK, 2007.

[3] H. J. Escalante, M. Montes and L. E. Sucar. "An Energy-based Model for Region Labeling", Computer Vision and Image Understanding, in press, 2011.

[4] H. J. Escalante, M. Grubinger, C. A. Hernandez, J. A. Gonz ' alez, A. L ' opez, M. Montes, E. Morales, L. E. 'Sucar, and L. Villasenor. "The Segmented and Annotated IAPR TC-12 Benchmark", Computer Vision and Image Understanding, Vol. 114, pp. 419-428, 2010.

[5] J. Shi and J. Malik. "Normalized Cuts and Image Segmentation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22 (8), pp. 888-905, 2000.

[6] L. T. Lan and A. Boucher. "An Interactive Image Retrieval System: from Symbolic to Semantic" Proceedings of the International Conference on Electronics, Informations and Communications, Hanoi, Vietnam, 2004.

[7] J. Shi and J. Malik. "Normalized Cuts and Image Segmentation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22 (8), pp. 888-905, 2000.

[8] M. J. Swain, D. H. Ballard, "Color indexing, International Journal of Computer Vision", Vol. 7, No. 1, 1991, pp. 11-32.

[9] L. Tran, "Efficient Image Retrieval with Statistical Color Descriptors", Ph.D. Thesis, Department of Science and Technology, Linkoping University, Linkoping, Sweden, 2003.

[10] J. Sun, X. Zhang, J. Cui and L. Zhou, "Image retrieval based on color distribution entropy", Pattern Recognition Letters, Vol. 27, No. 10, 2006, pp. 1122-1126.

[11] Messing, D.S, van Beek. P, Errico. J. H., " The MPEG-7 colour structure descriptor: image description using colour and local spatial information", International Conference on Image Processing, Thessaloniki, Greece, 2001, ISBN: 0-7803-6725-1.

[12] ISO/IEC JTC1/SC29/WG11, Doc N3913: Study of CD 15938-3 MPEG-7 MUltimedia Content Description Interface - Part 3 Visual. ISO/IEC January 2001 (Pisa).

[13] R. Hansch, O. Hellwich, " Color Textons for Building Detection",Technische Universitat Berlin.

[14] T. Leung, J. Malik, Representing and recognizing the visual appearance of materials using three-dimensional textons, International Journal of Computer Vision. 43(1)(2001)29-44.23

[15] O. G. Cula, K. J. Dana, 3D texture recognition using bidirectional feature histograms, International Journal of Computer Vision. 59(1)(2004)33-60.

[16] M. Varma, A. Zisserman, A statistical approach to material classification from single images, International Journal of Computer Vision. 62(2)(2005)61-81.

[17] M. Varma, A. Zisserman, A statistical approach to material classification using image patch examplars, IEEE Trans. on Pattern Analysis and Machine Intelligence. 31(11)(2009)2032-2047.

[18] L. Liu, P. Fieguth, G. Y. Kuang, H. B. Zha, Sorted random projections for robust texture classification, in Proc. International Conference on Computer Vision, 2011, pp. 391-398.

[19] M. Hadid, A. Zhao, G. Ahonen, M. PietikÃ¤inen, "Computer Vision Using Local Binary Patterns", T. 2011, XVI, 212p., Hardcover.

[20] D. Robinson, "K-means clustering is not a free lunch".
http://varianceexplained.org/r/kmeans-free-lunch/ consulted on 5/4/2016.

[21] Q. Zhang and S.A. Goldman, "EM-DD: an improved Multiple Instance Learning Technique" Advances in Neural Information Processing Systems 14, pp. 1073-1080, 2002.

[22] Y. Chen, J. Bi and J. Z. Wang, "MILES: Multiple-Instance Learning via Embedded Instance Selection".

[23] S. O'HARA, B. A. DRAPER. "Introduction To The Bag Of Features Paradigm For Image Classification And Retrieval".

[24] Bag-of-words model. Wikipedia The Free Encyclopedia. 2016.
https://en.wikipedia.org/wiki/Bag-of-words model (Accessed 8-9-2016)

[25] D. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004).

[26] A. Vedaldi. "Scale Invariant Feature Transform (SIFT)".
http://www.vlfeat.org/api/sift.html (Accessed 8-9-2016).

[27] M.Huang, Z.Mu, H.Huang. " A Novel Approach for Interest Point Detection via Laplacian-of-Bilateral Filter". Journal of Sensors. Volume 2015 (2015), Article ID 685154, 9 pages.

[28]P.Arbelaez,C. Fowlkes. "Contour Detection and Hierarchical Image Segmentation".

[29]D.K.Park, Y.S.Jeon, C.S.Won. "Efficient Use of Local Edge Histogram Descriptor".

[30]J. Goldberger, S. Gordon and H. Greenspan. "Unsupervised image-set clustering using an information theoretic framework." IEEE Trans. on Image Processing, 15(2):449-458, 2006.

[31]N.Dhanachandra, K.Manglem,Y.J.Chanu. "Image Segmentation Using K - means Clustering Algorithm and Subtractive Clustering Algorithm "

[32]A.Tsai, J.Zhang, A.S.Willsky. "Expectation-maximization algorithms for image processing using multiscale models and mean- field theory, with applications to laser radar range profiling and segmentation".

[33] A.Y.Ng, M.I.Jordan, Y.Weiss. "On Spectral Clustering: Analysis and an Algorithm".

[34]N.Dalal, B.Triggs. "Histograms of Oriented Gradients for Human Detection".

[35] http://labelme.csail.mit.edu/Release3.0/

[36] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support Vector Machines for Multiple-Instance Learning," Advances in Neural Information Processing Systems 15, pp. 561-568, 2003.

[37] Y.Chen,J.Bi, J.Z. Wang. "MILES: Multiple-Instance Learning via Embedded Instance Selection". IEEE Transactions on pattern analysis and machine intelligence, VOL. 28, NO. 12, December 2006.

[38] S.Andrews, I.Tsochantaridis and T.Hofmann. "Support Vector Machines for Multiple-Instance Learning".Department of Computer Science, Brown University, Providence, RI 02912 stu,it,th@cs.brown.edu.