Updated Semester Project Page Plan

Selected Project Topic: Constrained Clustering of Images with textual (or visual) explanations

Selected Programming language: Python 3+

A. We have understood how the mandatory features work and are in the process of implementing them in python. We have chosen SIFT+RANSAC as the additional feature for our image data set.

MPEG-7 Color layout descriptor
It is used to capture the spatial distribution of color in an image.
We use OpenCV to read and perform operations on images.
The steps involved are as follows-

1. Read the image using OpenCV's imread method, which creates a numpy array from the image.
2. **Image partition** - Divide the image into 64 blocks, a grid of 8x8. The 8x8 grid ensures invariance to resolution or scale.
3. **Representative color selection** - A single color is selected as the representative of that block from each block. There are multiple methods to do this, such as applying k-means to the pixels from each block to get the most dominant colors. In our case, we have used a simple average of all the pixels as it is simpler and faster to compute. It also provides sufficient description accuracy.
4. The color space is converted from RGB to YCbCr using the cvtColor method of OpenCV, and we get a tiny image of 8x8 in YCbCr color space.
5. The Y, Cb and Cr components are extracted for each block, and we get 3 8x8 matrices.
6. **DCT transformation** is applied to each matrix using the DCT method of Opencv.
7. Each matrix is scanned in a zigzag fashion to group the low-frequency coefficients of the matrices.

Bag of Visual words: Framework to learn descriptors based on patches of data.
Libraries
imageio import imread
sklearn.feature_extraction.image import extract_patches_2d
sklearn.decomposition import PCA
sklearn.cluster import KMeans
joblib import Parallel, delayed
skimage import feature

Step 1: **Extracting features locally-** Extract subimages(patches) using function extract_patches_2d() from sklearn.feature_extraction.image We are defining the size of the patch and the number of patches per image. Using LBP features as a base descriptor- LBP operates in single-channel images so if RGB images are provided. It converts it to grayscale. We are obtaining features of LBP for each patch.
Step 2: **Cluster the local features**- Using the clustering algorithm, KMeans, define the number of visual words
Step 3: **Histograms from Bag of features-** We check the frequency of each visual word in each training image. First, computing features for each image and predicting the number of patches of images. Finally, computing histogram and appending in the array.

SIFT+RANSAC:

1. SIFT will give us the feature vector for an image.
2. It gets features and it is independent of scale, rotation, illumination, and perspective.
3. It finds and constructs a space to ensure scale invariance.
4. Then finds the difference of gaussians and find the important points inside an image.
5. Once the features are detected and extracted, we can apply the RANSAC algorithms.
6. Then we match the features.
7. We fit the transformation function fitting.
8. And we finally do the image registration using the RANSAC algorithm.

SURF (Speeded up robust features):
SURF descriptors have been used to locate and recognize objects, people or faces, to reconstruct 3D scenes, to track objects and to extract points of interest.
We use OpenCV to read and perform operations on images.
Step 1: Feature Extraction:
The approach for interest point detection uses a very basic Hessian matrix approximation. Surf uses the Hessian matrix because of its good performance in computation time and accuracy. SURF relies on the determinant of the Hessian Matrix for the measure of selecting both location and scale.
Fast-Hessian Detector
1. **Hessian matrix-based interest points**
   - Apply convolution with Gaussian kernel then the second-order derivative.
   - SURF performs the approximation (both convolution and second-order derivative) using box filters.
   - The approximated determinant of the Hessian is calculated.
2. **Scale-space representation**
   - The images are repeatedly smoothed with a Gaussian and subsequently sub-sampled in order to achieve a higher level of the pyramid.
   - The filter responses are normalised with respect to the mask size.This guarantees a constant Frobenius norm for any filter size.
   - The scale space is analysed by up-scaling the filter size. The layers are obtained by filtering the image with gradually bigger masks, taking into account the discrete nature of integral images and the specific structure of our filters.

Step 2: Feature Description:
The first step consists of fixing a reproducible orientation based on information from a circular region around the interest point. Then, we construct a square region aligned to the selected orientation, and extract
the SURF descriptor from it.
1. Orientation Assignment
   - We first calculate the Haar-wavelet responses in x and y direction, which is in a circular neighbourhood of radius 6s around the interest point(key point), with s the scale at which the interest point was detected.
   - Once the wavelet responses are calculated and weighted with a Gaussian (σ =2.5s) cantered at the interest point, the responses are represented as vectors in a space with the horizontal response strength along the abscissa and the vertical response strength along the ordinate. We then calculate the sum of vertical and horizontal wavelet responses in a scanning area, then change the scanning orientation (add

π/3), and re-calculate, until we find the orientation with largest sum value, this orientation is the main orientation of feature descriptor.

2. Descriptor Components

- Constructing a square region cantered around the key point and oriented along the orientation we already got above. The size of this window is 20s.
- Then the region is split up regularly into smaller 4 × 4 square sub-regions. For each sub-region, we compute a few simple features at 5×5 regularly spaced sample points.
- Each sub-region has a four-dimensional descriptor vector v for its underlying intensity structure **V = (∑ dx, ∑ dy, ∑|dx|, ∑|dy|)**. This results in a descriptor vector for all 4×4 sub-regions of **length 64**. (i.e. **dx** is the Haar wavelet response in the horizontal direction and **dy** the Haar wavelet response in the vertical direction)

B. Generation of constraints

1. Domain Knowledge

For the image dataset in *"VOCtrainval_06-Nov-2007\VOCdevkit\VOC2007\ImageSets\Main"* we have been provided with the list of different class of images.

The images have been classified as either an airplane, bicycle, cat, bird and etc.

- We will use this domain knowledge to create must-link constraints within the selected candidates of the images after performing knn.
- We will create the cannot-link constraints by using the transitivity rule. The final result of constraints would be a dictionary visualizing a connection graph between the images.
- The current problem is that there are images which belong to multiple classifications and we will have to find a way to ensure constraints accordingly.
- We are also considering restricting constraints so it avoids too many constraints and thus hampering the efficiency.

2. Clustering parameters

We are also considering constraints such as min and max data points in a cluster.

C. Selecting the constraint clustering algorithm

We are levitating towards PCK Means since we have to consider the aspect of expressibility and explanations of the clusters.