Image Retrieval - Practical lab

S.Nicolas & L. Heutte

January, 2023

1 Download `SmallHolidays` dataset

The Holidays dataset is a public[1] set of images which mainly contains some personal holidays photos. The dataset contains 1491 images in total : 500 queries and 991 corresponding relevant images. For this lab, we have resized the images to a smaller resolution. Download the archive `smallholidays.zip` at `https://sites.google.com/view/imageretrievalm2sd/`.

2 Implement a basic image retrieval system

Compute image descriptors. Using image pixels directly as features is the simplest idea to compute the similarity between two images. We will use a distance between the raw pixels between the two images. Resize the images to 16x16x3 and flatten the pixels as a vector.

What interpolation method is used when you resize the images ? Comment on what properties of the image are preserved at this low resolution of 16x16 ? Why would it be a good idea to compare images pixel by pixel at this resolution rather than at full resolution ?

Retrieve similar images to a query and compute a score :

1. Take one image as the image query, among the 500 queries. Note that all query names end with ”00”.
2. Compute a similarity score with the euclidean distance with all remaining images.
3. Display the 10 highest score images with associated rank. Check the retrieved images and comment on them.
4. Plot the precision and recall curve for this query. To compute recall, all the relevant images for a given query have names that start with the same numbers, but end with 1, 2, etc. For example, relevant images to query `105100.jpg` are : `105104.jpg`, `105101.jpg`, `105103.jpg` and `105102.jpg`.

3 Compute global evaluation on 500 queries

Instructions for the evaluation. Now you will compute global performance for the entire set of the 500 queries. Open the `holidays_images.dat` file and read the image queries names as follows (the names all end with ”00”) :

```python
for line in open("holidays_images.dat","r"):
  imname=line.strip()
  imno=int(imname[:-len(".jpg")])
  if imno%100==0:
        #read query image & retrieved similar images
```

Generate the output file as follows : each line is a query image with associated results, following the rule :

`result_line = query_image_name rank0 result_image_name0 rank1 result_image_name1 rank2 result_image_name2`

example : `101800.jpg 0 101800.jpg 1 101801.jpg 2 138008.jpg 3 138007.jpg`

An example file is `myresults.dat`.

Note : the order of queries is not relevant. If the query image is ranked, it must be ignored in the scoring.

Write a script `my_holidays_map.py` that takes your `.dat` file as input argument and returns a sample mAP computation.

Question. Compute mAP if you keep 1, 5, 10 of the best retrieved images for each of the 500 query. Comment on the results.

[1] Source : `http://lear.inrialpes.fr/~jegou/data.php`

4 Improving the image descriptors and similarity metrics

Implement other descriptors than just the raw pixels : gray level vs color histogram, Haralick parameters of GLCM vs Local Binary Patterns (LBP), HOG descriptors, etc... and see how performance improves. You can chose features from : `http://scikit-image.org/docs/0.13.x/api/skimage.feature.html`.

For the similarity metric, you can use from now on the `scipy.spatial.distance.cdist` function to compute distances. Apart from euclidean, you can use cosine distance etc.

Bag of Visual Words. Implement a simple version of the Bag of Visual Words model :

— divide images into patches (you can use `sklearn.feature_extraction.image.extract_patches_2d`)
— compute a dictionary of visual words (VW) with k-means
— assign each patch to its closest VW
— describe each image with a histogram of the assigned VW

5 Using a pre-trained model to extract features

Use the pre-trained model ResNet50 (trained on ImageNet dataset) to extract deep features and see how performance improve.

```
from keras import applications
model = applications.resnet50.ResNet50(weights='imagenet', include_top=False, pooling='avg')
# load image setting the image size to 224 x 224
img = image.load_img(img_path, target_size=(224, 224))
# convert image to numpy array
x = image.img_to_array(img))
# the image is now in an array of shape (3, 224, 224)
# need to expand it to (1, 3, 224, 224) as it's expecting a list
x = np.expand_dims(x, axis=0)
x = preprocess_input(x)
# extract the features
features = model.predict(x)[0]
# convert from Numpy to a list of values
features_arr = np.char.mod('%f', features)
```