Image Retrieval with Bag of Words and SIFT

Project Report #1

Mariateresa Nardoni

Abstract

If a set of images is given, Image Recognition is a typical problem to solve. This can be coniugated in different ways, in particular we can use Image Retrieval. We developed an Image Retrieval system using SIFT keypoints, BoVWs, and the INRIA Holidays dataset. We utilized SIFT descriptors and BoVWs to represent images, employing an euclidean metric to assign SIFT descriptors to the nearest BoVW centroid. Then we separated query BoWs from gallery BoWs and computed distances to construct a distance matrix for Image Retrieval. Finally we evaluated the system's performance using for qualitative analysis.

1. Introduction

In this project an Image Retrieval problem was tackled and discussed with the following specific aim: given a query image, find within the gallery set the images closest to the query image.

In particular to achieve this, we used SIFT (Scale-Invariant Feature Transform) keypoints, BoVWs (Bag of Visual Words) and the INRIA Holidays dataset. Each image in the dataset has its own number of descriptors, each of which is represented as a histogram. For each SIFT descriptor, we calculated its distance from each centroid to see which bin it falls into, assigning the SIFT to the nearest BoVW via an appropriate Euclidean metric. We then created an array of Bag of Visual Words for all the images.

Once this pre-processing is over, the central phase of our project can finally begins: we can start the Image Retrieval phase.

First we separated the query bows from the gallery bows, then we calculated all the distances between query bows and data in order to construct the distance matrix with the aim of visualizing which images were retrieved closest to the query bows.

Finally, qualitative analyses of the results obtained were made through the use of the mean Average Precision (mAP) function

1.1. Motivation

Given a specific query image, Image retrieval is the process of finding the same (or the most similar) image(s) in the dataset: a user typically provides an image query, and the system searches in the database for images that are similar to the query image based on visual appearance, features, or content. The primary goal is to find a set of images that match the desired concept or appearance.

This project analysed in class brings an example of practical application, using images (taken from both outdoor and indoor environments) with the goal of testing what we studied and discussed.

The goal of this project is to show how it is possible to do Image Retrieval by exploiting SIFTs and BoVWs of images, in order to achieve correct (or not) retrieval of our query image. In addiction, performance measures will also be provided, which they will change according to different parameters that characterize our project, with the aim of being able to analyze broader cases.

2. Method

The project was written in Python language. All calculations and results obtained were processed on Google Colab.

The following resources were used in the realisation of this project, a brief explanation follows:

2.1. INRIA Holidays Dataset

The INRIA Holidays Dataset is a computer vision dataset that serves as a benchmark for retrieval tasks. It is a collection of images used for evaluating and testing image search and retrieval algorithms. This dataset is particularly popular in the field of computer vision for evaluating image similarity and content-based image retrieval techniques.

This dataset is a set of images which mainly contains holidays photos. The remaining ones were taken on purpose to test the robustness to various attacks such as rotations, viewpoint and illumination changes.

The dataset includes a very large variety of scene types and images are in high resolution. This dataset contains an annotated set of 1491 images with 500 query images.

2.2. SIFT

A SIFT keypoint is a keypoint detected by a Scale-Invariant Feature Transform (SIFT) algorithm. These keypoints are identified in an image in a way that uniquely and robustly represents local image features, making them invariant to scale, rotation, and deformations.

It's a circular image region with an orientation and it is described by a geometric frame of four parameters:

- ▷ the *keypointer center coordinates* (x, y)
- ▷ its *scale* (radius of the region)
- ▷ its *orientation* (angle expressed in radians)

These keypoints were invented by David Lowe in 1999 and are used in computer vision applications for tasks such as object recognition, motion tracking, and finding correspondences between images.

Due to their invariance to geometric transformations, SIFT keypoints are extremely useful for image analysis and processing in contexts where objects may appear in different conditions and orientations.

2.3. Bag of Visual Words (BoVW)

"Bag of Visual Words" (BOVW) is a concept used in the field of image recognition and digital image processing, it's an extension of the "Bag of Words" (BoW) concept used in text recognition. Essentially, BoVW is a representation model for images based on the idea of breaking down an image into "visual words" or "visual descriptors" and then creating a "bag" of these words to represent the image as a whole.

A visual vocabulary is created by clustering the descriptors from many images into groups or "visual words." The image is then represented by counting how often each "visual word" from the visual vocabulary appears in the image. These frequencies are typically organized into a vector, which represents the image.

BoVW has been applied in various computer vision applications, particularly in object recognition and image retrieval.

3. Work done

The first phase of the project is the pre-processing one, in which the resources that we will need in the realisation of the Image Retrieval were downloaded and loaded. First, the already precomputed SIFTs were downloaded. SIFT is a 128 dimensional histogram of orientations and it is extracted using scale and orientation invariant operators. Next, the dictionary of visual words was downloaded. This dictionary was learned on Flickr60K and has the following sizes: 100, 200, 500, 1K, 2K, 5K, 10K, 20K, 50K, 100K, 200K. The size of the dictionary is set at the beginning of the code, this will be varied to study different performances. All files ending with *.siftgeo* are then loaded, these are the 1491 images that we are going to work on.

As a first step, what has been done is to construct a dictionary of SIFTs containing the informations we are interested in, i.e. the informations on the descriptors, eliminating the superfluous ones. The SIFTs were then processed to create a dictionary called img_descs, which associates each image id with the array of SIFTs. This dictionary was created both to maintain the relationship between the image name and the descriptors and because the number of SIFTs for an image may vary. Furthermore it should also be taken into account that an image may not have any associated SIFTs. Each image, therefore, will have its own number of histogram descriptors, each with 128 elements (bins).

Once the SIFTs have been defined in this way, we move on to prepare the visual words. First, the vocabulary size will be chosen, which will make the representation more or less fine depending on the value entered.

Defined the SIFTs and the visual words, what we have to do is to calculate the distance between them using an Euclidean metric, with the aim of associating to each SIFT the nearest centroid (a visual word), to verify in which bin it falls. Cases involving the 'Euclidean' and 'Cosine Similarity' metrics were studied, comparing them and examining them for the various dimensions of the dictionary of visual words. This calculation was done both by writing the code manually and by exploiting scikit-learn's NearestNeighbors function in order to speed up the computation considerably. The results obtained are the same.

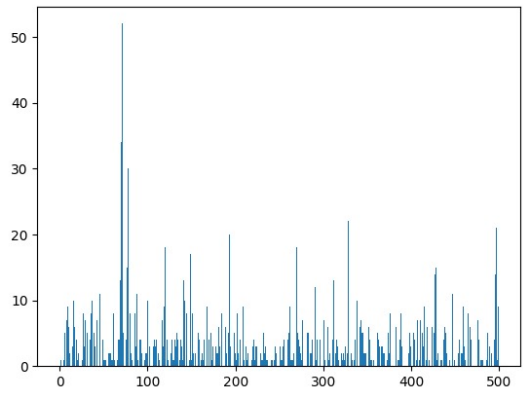

Figure 1. Example of a histogram for a single image with a dictionary size of 500

This calculation to obtain the BoWs was done for each image in the dataset. Once we have obtained all the BoWs, and thus defined an array of BoWs called bow_array, we can finally begin the main phase of the project: the Image Retrieval one.

First we have to prepare the data: to do this, we have separated the query images from the gallery images, so that we can do the experiments. Query images are those that end with '00' or have their id divisible by 100. The query images in the INRIA Holidays dataset are in total 500 and their size is the vocabulary size of the visual words defined at the beginning of the project. We then calculated all the distances between the queries and the data in order to construct the distance matrix, so that the element retrieved closest to our query is displayed in the first position. A similarity ranking is realised. The metrics "Euclidean" and "Cosine Similarity" were examined in the construction of the matrix, so that the results could be compared. This calculation was done again by exploiting scikit-learn's NearestNeighbors function and changing the metrics used within it.

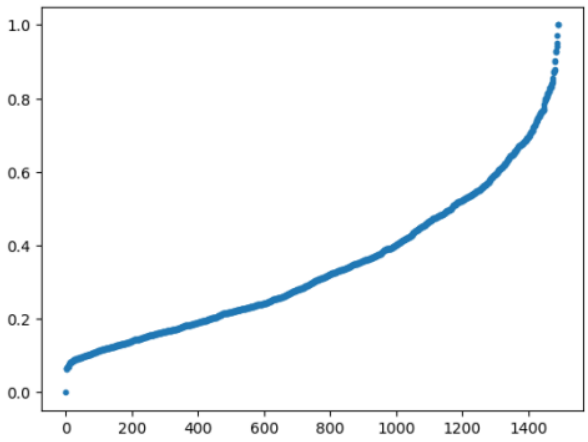

Figure 2. Example of plotting the distances of all images in the dataset from a single query

Based on the indices of the retrieved images, we want to analyse the qualitative performance of the results obtained. To do this, the mean Average Precision (mAP) function was examined so that we can test how the various results change depending on the chosen metrics and the size of the vocabulary used. Depending on these values, Image Retrieval will be more or less precise in its intent. We also want to test the robustness of our method by analysing when a query image is retrieved correctly, especially if it has undergone a geometric transformation, such as a rotation or a translation. This will allow us to provide a qualitative analysis on the performance of the methods used, in particular this allows us to understand how the robustness and invariance of SIFTs features can be useful in Image Retrieval operations.

Obviously, the performance comparison was also made taking the ground truth into account, in order to analyse how far the results deviated from the real ones. The results and graphs obtained are analysed and discussed in the next section.

4. Results

In the following, the results obtained with the mean Average Precision (mAP) function through the use of a different metric, the "Euclidean" and the "Cosine Similarity" one, in calculating the distance between SIFTs and centroids (**the first distance in the tables**) and in calculating the distance between Query Bows and data (**the second distance in the tables**) will be analyzed.

Size	mAP: Cosine Similarity - Cosine Similarity
100	0.381
200	0.398
500	0.407
1000	0.401
2000	0.396
5000	0.387
10000	0.401

Size	mAP: Euclidean - Euclidean
100	0.332
200	0.349
500	0.393
1000	0.384
2000	0.337
5000	0.214
10000	0.105

Size	mAP: Cosine Similarity - Euclidean
100	0.336
200	0.354
500	0.393
1000	0.381
2000	0.340
5000	0.218
10000	0.108

Size	mAP: Euclidean - Cosine Similarity
100	0.379
200	0.394
500	0.404
1000	0.403
2000	0.398
5000	0.389
10000	0.403

Table 1. mAP results obtained with different metrics

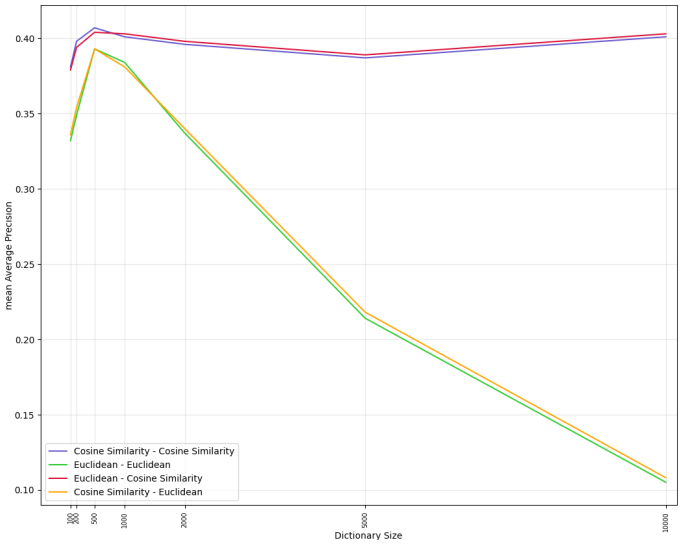

Figure 3. Comparison of proposed metrics

The graph represents the trend of mean Average Precision as the vocabulary size grows. Mean Average Precision evaluate how good a ranking is, this means that a high mAP corresponds to a ranking with the few relevant images in the first positions.

Studying the trend of the graphs, it can be seen that Cosine Similarity - Cosine Similarity and Euclidean - Cosine Similarity have a rather similar trend: both reach their peak performance with a vocabulary size of 500, and then both get worse but improve again with a size of 10000. Cosine Similarity - Cosine Similarity has the best performance recorded until it reaches the size of 500, after which it is beaten by Euclidean - Cosine Similarity which has better performance when the dictionary exceeds the size of 500. Worst performances, on the other hand, are recorded by Euclidean - Euclidean and Cosine Similarity - Euclidean metrics. Both again reach peak performance at a dictionary size of 500, and, thereafter, will experience a decline in perfomances as the size increases. The best metric analyzed therefore turns out to be Cosine Similarity - Cosine Similarity with a size of 500.

Once the performances were tested, we tested the Image Retrieval of the query image.

In the first case analysed (Cosine Similarity - Cosine Similarity, size = 500) it can be seen that there are many retrieved elements close to the query itself and few are far away. The results obtained in the Image Retrieval were retrieved correctly: in the first case the landscape image is retrieved correctly despite a translation, in the second case the food is retrieved correctly even though it has undergone a scale transformation, but when analysing the ground truth, the rotated image itself is not retrieved. However, it is possible to see, despite the errors, a good retrieving of the query

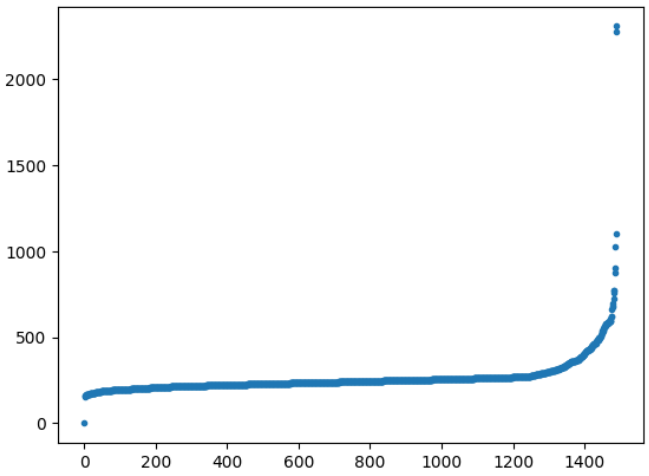

Figure 4. Cosine Similarity - Cosine Similarity, mAP = 0.407, size = 500

Figure 5. Results, Cosine Similarity - Cosine Similarity, mAP = 0.407, size = 500

Figure 6. Ground truth, Cosine Similarity - Cosine Similarity, mAP = 0.407, size = 500

image. Using the Cosine Similarity metric - Cosine Similarity with a dictionary size of 500 allows us to obtain good results that are fairly resistant to geometric transformations of the image, albeit with some errors.

In the second case analysed (Euclidean - Euclidean, size = 10000), the worst case was examined. No images could be found close to the query, as can be seen from the graph and the results. The results do not allow all the images that are closest to the query to be retrieved correctly, in particular it is interesting to see how, in the first result proposed, the method is so bad that it does not allow the mountain to be retrieved correctly, but assigns a completely black image as the closest image. With these metrics and this vocabulary size, moreover, robustness to geometric transformations is completely lost, as an image close to the query subjected to such a deformation is never found.

As the size of the dictionary increases, performance drops noticeably, so we can say that good results are ob-

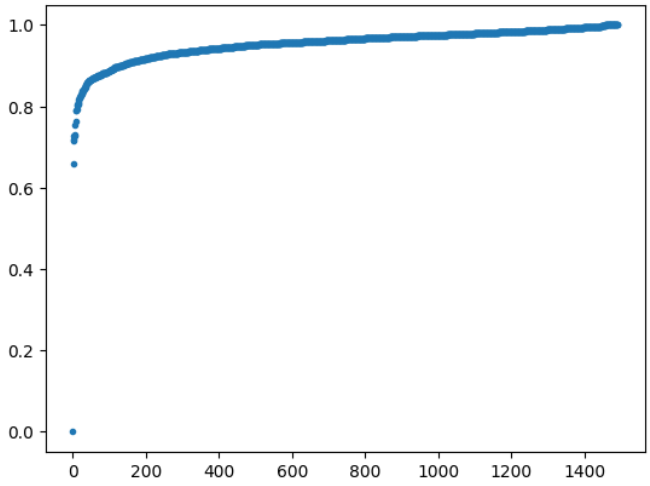

Figure 7. Euclidean - Euclidean, mAP = 0.108, size = 10000

Figure 8. Results, Euclidean - Euclidean, mAP = 0.108, size = 10000

Figure 9. Ground truth, Euclidean - Euclidean, mAP = 0.108, size = 10000

tained with a dictionary size of 500. Using dictionaries with a size greater than 500 significantly lowers the quality and performance of Image Retrival, and also does not allow for good query image retrieval by losing robustness to geometric transformations and illumination changes.

Appendix - Images

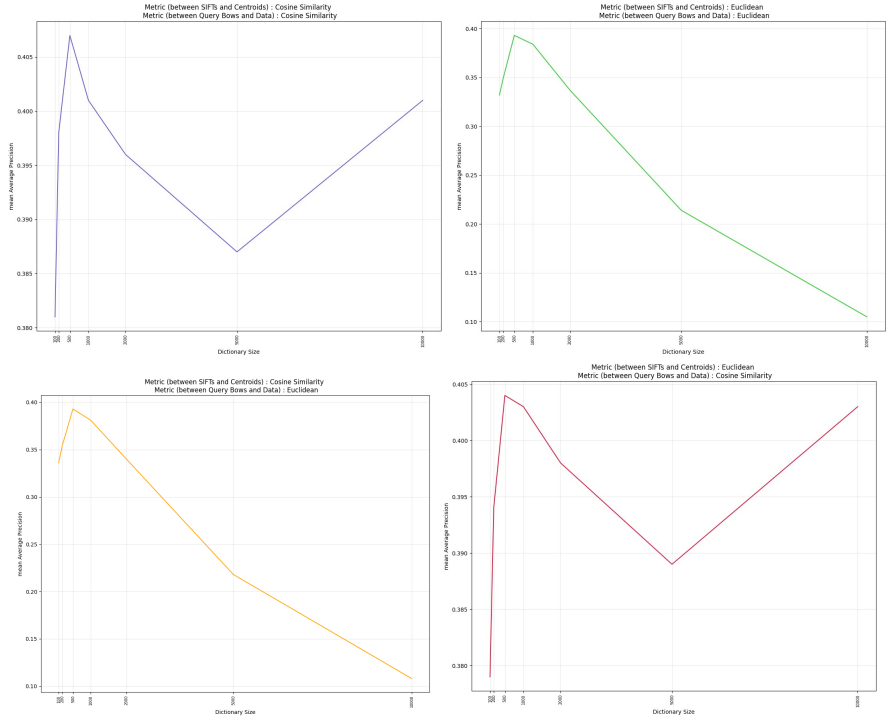

Figure 1: Plots of the various metrics analyzed

Appendix - Images

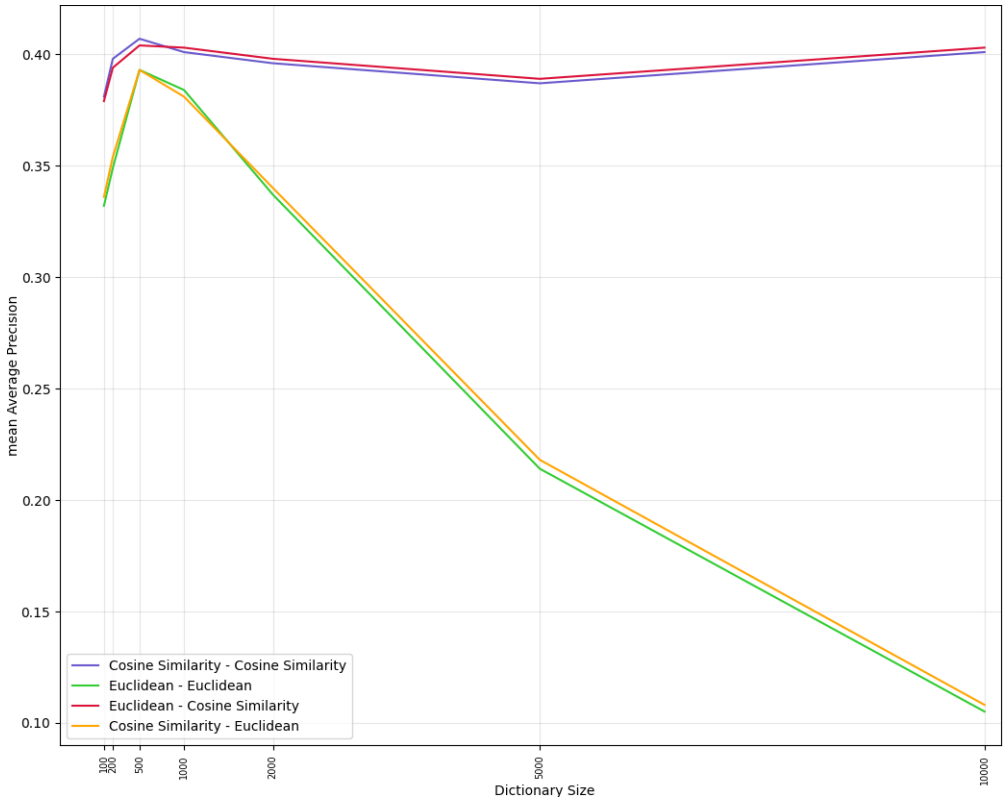

Figure 2: Comparison of proposed metrics