BBM 418 - Computer Vision Laboratory

HACETTEPE UNIVERSITY

Department of Computer Engineering

Programming Assignment-1
Spring 2018-2019
Dr. Nazli Ikizler Cinbis
TA. Ozge Yalcinkaya

Due Date: 23:59 pm on Monday, April 8th, 2019

Image Retrieval Basics

In this assignment, you will get familiar with basic image representation and retrieval methods. First of all, you will experiment with simple image representations and use k-NN classification method to obtain categories of the query images. You will compare the results in terms of the classification accuracy. Then, in the second part, you will use a more advanced method to represent the images and you will repeat the classification part. At the end, you will compare all the results and make comments on them. You will use a subset of Caltech 256 [1] dataset.

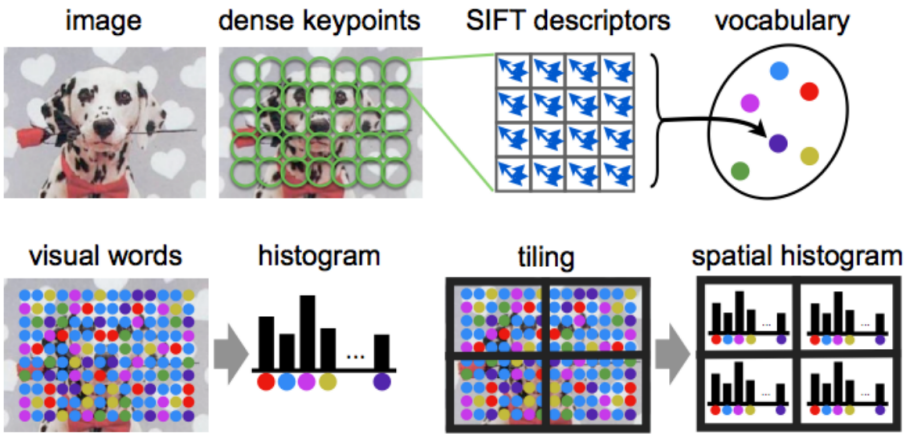

Figure 1: Bag of Visual Words

Background

k-NN Classification: k-Nearest Neighbor is a simple classifier. Consider you have all image representations both for training and query images. What you are going to do is:

1. Finding the most similar training images for each query image. You will use Euclidean distance to calculate the similarity between two image feature vectors.
2. In this case, your classifier will be 1-NN: you will assign a category to each query image by considering only the most similar train image's category.
3. After that, you will calculate the average and class based accuracy of the query images by using ground-truth labels of the query images. Average accuracy is the ratio of correctly predicted query images and class-based accuracy is the ratio of correctly predicted query images for each category separately.

Dataset: There are 10 object categories that is selected from [1]. In the training set, you have 30 images whereas in the query set you have 5 images for each object category. Therefore, you will calculate the average and class based accuracies over 50 query images. Note that, all the images are named with a related category number.

PART 1 - Simple Image Representations

a) Gabor Filter Bank

You will create different gabor filters by using a built-in gabor filter function from python libraries. Then, you will apply your gabor filters to each image to get different filter responses. Gabor filters detect different orientations in

the images. You will calculate the mean amplitude of each response to obtain a gabor filter bank feature vector for each image. Basically, you will take the mean of responses for each filter and put them to a vector. At the end, if you have 40 filters, your image vector should be in the length of $1x40$ where each element includes a mean value from the related response map. You may normalize your vectors between [0, 1].

You should experiment with different number of gabor filters in different orientations and show the effect of using different filters on accuracy. For each number of filters, you should calculate the average and the class-based accuracies.

b) Average SIFT

Extract SIFT feature vectors by using a built-in function and then take the average of the descriptors so that you will have a vector with length $1x128$ for each image. Then, similar to part 1.a, you should calculate the accuracies.

PART 2 - Bag of Visual Words

Bag of visual words (BoW) is a strategy that draws inspiration from the text retrieval community and enables efficient indexing for local image features. In document classification, a bag of words is a sparse vector of occurrence counts of words; that is, a sparse histogram over the vocabulary. In computer vision, a bag of visual words is a sparse vector of occurrence counts of a vocabulary of local image features. BoW descriptor extraction consists of 3 main parts. As the illustration at Figure 1 shows, you have to i) extract SIFT features, ii) create a code book, iii) quantize each feature and calculate a histogram from it, with or without a spatial tiling.

1. *Codebook*

   Combine only training set's SIFT descriptors each being $Nx128$ vectors and use k-means to extract a codebook over them with k=500.

2. *Quantization*

   Use your codebook to get visual word representation of each image (both train and query) by finding the closest codebook word of each SIFT descriptor of the images. In the end, each image will be represented as $1xN$ where each descriptor in the image is described by a cluster centroid id.

3. *Histogram*

   Represent each image with a histogram of codebook that is in the length of $1x500$.

   - *Spatial tiling*
     You have to split image into nxn tiles and calculate a histogram from each part separately. Since you have 500 words in your codebook, one histogram from one tile's length has to be 500. After calculating one histogram from each tile, concatenate all. This way you will have one histogram matrix in the size of $nx500$. You have to try different number of tilings and calculate accuracies. Explain the effect of spatial tiling. In your report, discuss the rationale behind the spatial tiling.

Experiments

Compare the four representations described above (Gabor filter bank, Average SIFT, BoW, BoW with spatial tiling) in terms of class-based accuracy and average accuracy and discuss about your observations. After the gabor filter experiments, choose the best fixed gabor filter bank and give the results using this best one for this comparison. Similarly, for spatial tiling , you can take the best performing spatial tiling number after your detailed experiments with different n.

To show the performance of different options (Gabor filter bank, Average SIFT, BoW, BoW with spatial tiling), first, select 3 example query images from 3 different categories. Then, for these query images, show the 5 most similar training images found for each of the four options described above.

Make your comments in detail for all the results, explore the reasoning of why the results are different etc.

Bonus: Think about the drawbacks of the Euclidean distance. Propose a different distance measure to find the similarity between two feature vectors. Show experimental results, explain why it worked/didn't work.

The Implementation Details

1. You are only allowed to use built-in functions for creating a gabor filter, doing convolution, extracting SIFT descriptors, k-means clustering and distance calculation. For other steps, you should implement your own functions.
2. You are expected to write functions for each step and give a main script to run all results.
3. You should pay attention to code readability such as comments, function/variable names and your code quality: 1) no hard-coding 2) no repeated code 3) cleanly separate and organize your code 4) use consistent style, indentation
4. Your code should read all images from a folder named "dataset" and write results to the console.
5. Implement your code with Python 2.7 and use libraries from Anaconda. You can install any library that is not in Anaconda as well, such as OpenCV.

What should you write in the report?

- Explain all steps in detail such as why we are doing this, what is a Gabor filter bank, what it provides etc. Do this explanations for each step.
- Give experimental results, used parameters and comments on the results in detail.
- Explain your implementation and your approach to the problem.
- A basic structure might be: 1) Introduction (what is the problem, how do you approach to this problem, what is the content of your report) 2) Implementation Details (the method you followed and details of your solution) 3) Experimental Results (all results for separate parts with different parameters and your comments on the results) 4) Conclusion (what are the results and what are the weaknesses of your implementation, in which parts you have failed and why, possible future solutions)
- You should write your report in LaTeX
- You should give visual results by using a table structure.

What to Hand In

Your submission format will be:

- README.txt *(give a text file containing the details about your implementation, how to run your code, the organization of your code, functions etc.)*
- code/ *(directory containing all your code)*
- report.pdf

Archieve this folder as **b<studentNumber>.zip** and submit to `https://submit.cs.hacettepe.edu.tr`.

Grading

The assignment will be graded out of 100:

- **30 (part 1):** CODE: 0 (no implementation), 5 (a partially correct solution), 10 (a correct solution) and REPORT: 20
- **70 (part 2):** CODE: 0 (no implementation), 15 (a partially correct solution), 30 (a correct solution) and REPORT: 40

Academic Integrity

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in the pseudocode) will not be tolerated. In short, turning in someone else's work, in whole or in part, as your own will be considered as a violation of academic integrity. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else.

References

[1] http://www.vision.caltech.edu/Image_Datasets/Caltech256/