Final Project Part 1
Bag-of-Words based Image Classification

Computer Vision 1
University of Amsterdam

Due 11:59pm, October 27, 2019

General guidelines

Your source code and report must be handed in together in a zip file (`ID1_ID2_ID3_ID4.zip`) before the deadline by sending it to **computervision1.uva(at)gmail.com**. For full credit, make sure your report follows these guidelines:

- Include an introduction and a conclusion to your report.
- The maximum number of pages is 10 (single-column, including tables and figures). Please express your thoughts concisely. The number of words does not necessarily correlate with how well you understand the concepts.
- Answer all given questions (in green boxes). Briefly describe what you implemented. Blue boxes are there to give you hints to answer questions.
- Try to understand the problem as much as you can. When answering a question, give evidences (qualitative and/or quantitative results, references to papers, etc.) to support your arguments.
- Analyze your results and discuss them, e.g. why algorithm A works better than algorithm B in a certain problem.
- Tables and figures must be accompanied by a brief description. Do not forget to add a number, a title, and if applicable name and unit of variables in a table, name and unit of axes and legends in a figure.

Late submissions are not allowed. Assignments that are submitted after the strict deadline will not be graded. In case of submission conflicts, TAs' system clock is taken as reference. We strongly recommend submitting well in advance, to avoid last minute system failure issues.
Plagiarism note: Keep in mind that plagiarism (submitted materials which are not your work) is a serious crime and any misconduct shall be punished with the university regulations.

1 Introduction

The goal of the assignment is to implement a system for image classification; in other words, this system should tell if there is an object of given class in an image. You will perform 5-class (airplanes, birds, ships, horses and cars) image classification based on bag-of-words approach[1] using SIFT features. STL-10 dataset[2] will be used for the task. For each class, test sub-directories contain 800 images, and training sub-directories contain 500 images. Images are represented as (RGB) 96x96 pixels.

Download the dataset from `http://ai.stanford.edu/~acoates/stl10/stl10_matlab.tar.gz`. There are three files: *train.mat*, *test.mat* and *unlabeled.mat*. For the project, you will just use the train and test partitions. Download the dataset and make yourself familiar with it by figuring out which images and labels you need for the aforementioned 5 classes. Note that you do not need *fold_indices* variable.

1.1 Training Phase

Training must be conducted over the training set. Keep in mind that using more samples in training will likely result in better performance. However, if your computational resources are limited and/or your system is slow, it's OK to use less number of training data to save time.

Hint

In order to debug your code use a small amount of input images/descriptors. Once you are sure everything works properly, you can run your code for the experiment using all the data points.

Hint

You are not allowed to use the test images for training purpose.

1.2 Testing Phase

You have to test your system using the specified subset of test images. All 800 test images should be used at once for testing to observe the full performance. Remember to exclude them from training for fair comparison.

[1]`http://www.robots.ox.ac.uk/~az/icvss08_az_bow.pdf`
[2]`https://cs.stanford.edu/~acoates/stl10/`

2 Bag-of-Words based Image Classification

Bag-of-Words based Image Classification system contains the following steps:

1. Feature extraction and description
2. Building a visual vocabulary
3. Quantize features using visual dictionary (encoding)
4. Representing images by frequencies of visual words
5. Classification

We will consider each step in detail.

2.1 Feature Extraction and Description

SIFT descriptors can be extracted from either (1) densely sampled regions or (2) key points. You can use VLFeat functions for dense SIFT (e.g. `vl_dsift`) and key points SIFT descriptor extraction (e.g. `vl_sift`). Moreover, it is expected that you implement not only for grayscale SIFT, but also for RGB-SIFT and opponent-SIFT.

Hint

Check out the MATLAB API of VLFeat for further information in the following link: `http://www.vlfeat.org/matlab/matlab.html`

Use VLFeat functions in order to extract SIFT descriptors while working on RGBSIFT and etc.

2.2 Building Visual Vocabulary

Here, we will obtain visual words by clustering feature descriptors, so each cluster center is a visual word, as shown in Figure 1. Take a subset (maximum half) of all training images (this subset should contain images from ALL categories), extract SIFT descriptors from all of these images, and run k-means clustering (you can use your favourite k-means implementation) on these SIFT descriptors to build visual vocabulary. Then, take the rest of the training images to calculate visual dictionary. Nonetheless, you can also use less images, say 100 from each class (exclusive from the previous subset) if your computational resources are limited. Pre-defined cluster numbers will be the size of your vocabulary. Set it to different sizes (400, 1000 and 4000).

Hint

Remember first to debug all the code with a small amount of input images and only when you are sure that code functions correctly run it for training over the larger data.

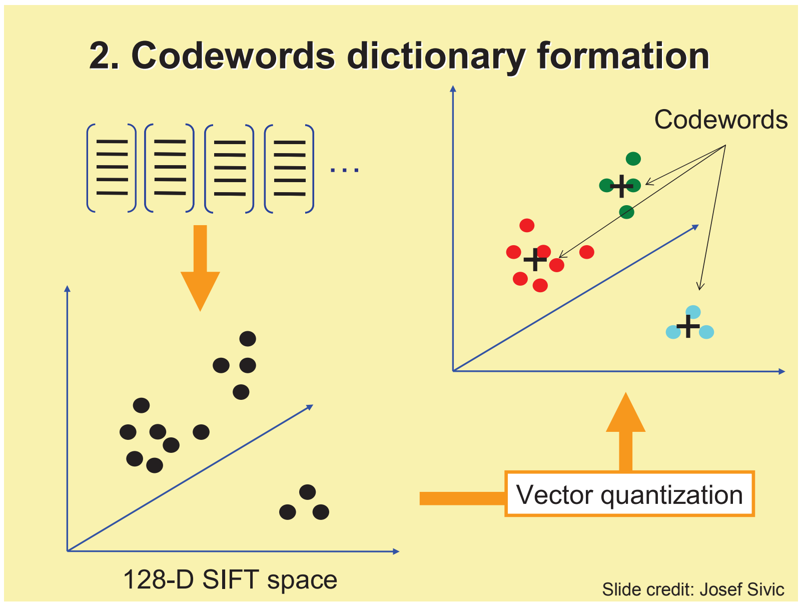

Figure 1: Learning Visual Dictionary. Code-words is another term for visual words.

2.3 Encoding Features Using Visual Vocabulary

Once we have a visual vocabulary, we can represent each image as a collection of visual words. For this purpose, we need to extract feature descriptors (SIFT) and then assign each descriptor to the closest visual word from the vocabulary.

2.4 Representing images by frequencies of visual words

The next step is the quantization, the idea is to represent each image by a histogram of its visual words, see Figure 2 for overview. Check out MATLAB's `hist` function. Since different images can have different numbers of features, histograms should be normalized.

2.5 Classification

We will train a Support Vector Machine (SVM) classifier per each object class. As a result, we will have 5 binary classifiers. Take images from the training set of the related class (should be the ones which you did not use for dictionary calculation). Represent them with histograms of visual words as discussed in the previous section. Use at least 50 training images per class or more, but remember to debug your code first! If you use the default setting, you should have 50 histograms of size 400. These will be your positive examples. Then, you will obtain histograms of visual words for images from other classes, again about 50 images per class, as negative examples. Therefore, you will have 200 negative examples. Now, you are ready to train a classifier. You should repeat it for each class. To classify a new image, you should calculate its visual words histogram as described in Section 2.4 and use the trained SVM classifier to assign it to the most probable object class. (Note that

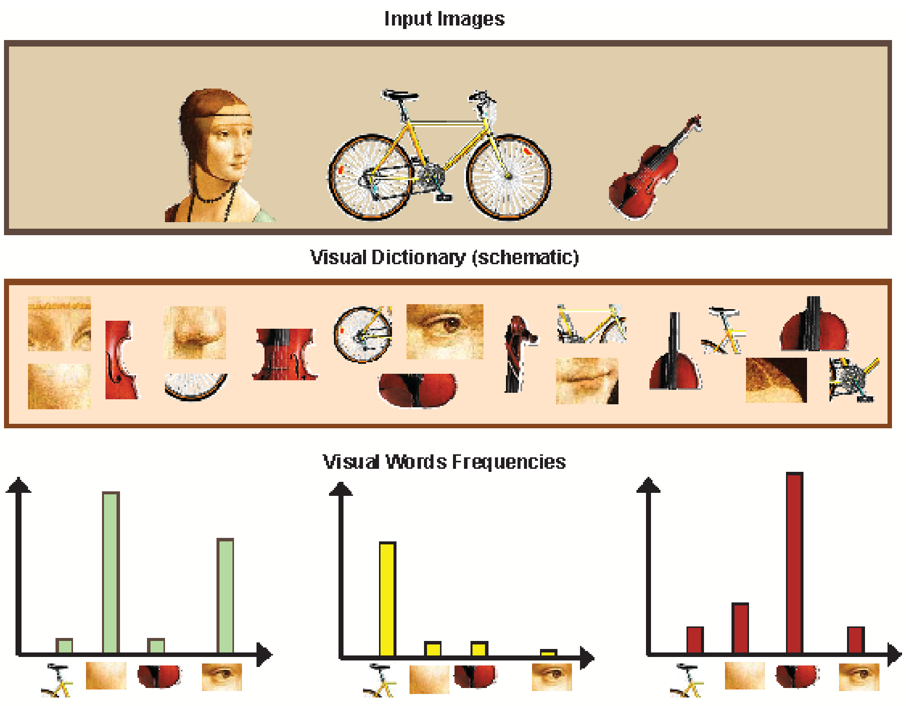

Figure 2: Schematic representation of Bag-Of-Words system.

for proper SVM scores you need to use cross-validation to get a proper estimate of the SVM parameters. In this assignment, you do not have to experiment with this cross-validation step).

Hint

You can use Libsvm library that provides a functionality (-b) to obtain class probabilities per image (`https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/#multi_class_classification_and_probability_output_via_error_correcting_codes`), or you can use your favorite SVM library that supports to use of confidence values for classifiers.

2.6 Evaluation

To evaluate your system, you should take all the test images from all classes and rank them based on each binary classifier. In other words, you should classify each test image with each classifier and then sort them based on the classification score. As a result, you will have five lists of test images. Ideally, you would have images with airplanes on the top of your list which is created based on your airplane classifier, and images with cars on the top of your list which is created based on your car classifier, and so on.

In addition to the qualitative analysis, you should measure the performance of the system quantitatively with the Mean Average Precision over all classes. The Average

Precision for a single class c is defines as

$$\frac{1}{m_c}\sum_{i=1}^{n}\frac{f_c(x_i)}{i}\ , \tag{1}$$

where n is the number of images ($n = 50 \times 5 = 250$), m is the number of images of class c ($m_c = 50$), x_i is the i^{th} image in the ranked list $X = \{x_1, x_2, \ldots, x_n\}$, and finally, f_c is a function which returns the number of images of class c in the first i images if x_i is of class c, and 0 otherwise. To illustrate, if we want to retrieve R and we get the following sequence: $[R, R, T, R, T, T, R]$, then $n = 7$, $m = 4$, and $AP(R, R, T, R, T, T, R) = \frac{1}{4}\left(\frac{1}{1} + \frac{2}{2} + \frac{0}{3} + \frac{3}{4} + \frac{0}{5} + \frac{0}{6} + \frac{4}{7}\right)$.

3 Deliverables (60 pts.)

1. Students are expected to prepare a report for this project. The report should include the analysis of the results for different settings such as:

   - key points vs dense sampling
   - mAP based on vocabulary size
   - mAP based on SIFT descriptor used (grayscale SIFT, RGB-SIFT and opponent-SIFT)

   Do not just provide numbers, remember to follow the general guidelines and discuss different settings.

2. For qualitative evaluation, you are expected to visualize the top-5 and the bottom-5 ranked test images (based on the classifier confidence for the target class) per setup. That means you are supposed to provide a figure for each experimental setup (2 for SIFT sampling variants x 3 for vocabulary size x 3 for SIFT color variants = 18 experimental setups in total), as discussed in Section 2.6.

3. A demo function which runs the whole system should be prepared and submitted with all other implemented functions (excluding VLFeat and LIBSVM functions).

Hint

Having visual elements such as charts, graphs and plots are always useful for everyone. Keep this in mind while writing your reports.

4 Bonus (Max. 5 pts.)

We will award groups that prove to put extra efforts into their project. By *extra* we mean one or more of the following:

- **[1 pts.]** Using another classification approach (such as k-Nearest Neighbor). You can simply use the features you extracted in this project.
- **[1 pts.]** Experimenting with different number of training images (i.e., only using quarter of the whole training set, only using half of the training set as opposed to full training set to observe the possible performance drop).
- **[2 pts.]** Using another type of features other than SIFT.
- **[Up to 5 pts.]** Any other thing that you can think of or find from the literature that can boost the results.

In order to get the bonus, your submission should first satisfy the previous steps. You need to implement and explain the method and justify your reasoning. You can get at most 5 points from bonus part.