Experiment 10: Image Classification

(Credit: Ivan Laptev, Jean Ponce, Cordelia Schmid and Josef Sivic, Object recognition and computer vision 2016, http://www.di.ens.fr/willow/teaching/recvis16/assignment2/)

Theory:

Classifier is a procedure that accepts a set of features and produces a class label from them. A classifier is anything that takes a feature set as an input and produces a class label. How to build a classifier? Take a set of labelled examples and use them to come up with rule that assigns a label to any NEW example. In general: We have training dataset (xi,yi), where xi = each of feature vector xi consists of measurements of the properties of the different types of objects, and yi = labels representing the type of objects that generated the example. Classifier is an important tool in high level vision. Many problems can be abstracted to form that looks like classification.

You should think classifier as a rule: We pass some feature vector and the rule returns a class label. We know the relative cost of mislabeling each class : Must come up with a rule that can take any plausible x and assign a class to it, in such a way that the expected mislabeling cost is as small as possible, or at least tolerable.

Goal

In image classification, an image is classified according to its visual content. For example, does it contain an airplane or not. An important application is image retrieval - searching through an image dataset to obtain (or retrieve) those images with particular visual content.

The goal of this exercise is to get basic practical experience with image classification. It includes: (i) training a visual classifier for five different image classes (aeroplanes, motorbikes, people, horses and cars); (ii) assessing the performance of the classifier by computing a precision-recall curve.

Getting started

- Download the code and data ~914MB (http://www.di.ens.fr/willow/teaching/recvis16/assignment2/practical-category-recognition-2015a.tar.gz)
- You can also download the code and data separately:
    - Code Only (12 MB): http://www.di.ens.fr/willow/teaching/recvis16/assignment2/practical-category-recognition-2015a-code-only.tar.gz
    - Data Only (902 MB): http://www.di.ens.fr/willow/teaching/recvis16/assignment2/practical-category-recognition-2015a-data-only.tar.gz
- Unpack the code archive. This will make a directory called practical-category-recognition-2015a.
- Unpack the data archive into practical-category-recognition-2015a/data/. This should create several sub-directories in the data directory such as practical-category-recognition-2015a/data/images/
- Start your MATLAB in the directory practical-category-recognition-2015a.
- Try running setup.m command (type setup without the .m suffix). If all goes well, you should obtain a greeting message. Note: this exercise is using several functions from the open source Matlab package vlfeat by A. Vedaldi.
- Note: If you have trouble running pre-compiled libraries in VLFeat, you can try to download separately this version of VLfeat: http://www.vlfeat.org/download/vlfeat-0.9.20-bin.tar.gz

As you progress in the exercises you can use MATLAB help command to display the help of the MATLAB functions that you need to use. For example, try typing help setup.

Exercise description

The image-classification directory contains Matlab scripts exercise1.m and exercise2.m, which contain a commented skeleton of the code**. You will need to (1) complete these scripts by writing your code, and (2) produce a written report containing answers to questions given below (shown in red).** Your answers should include results, graphs or images as specified in each question. Start by opening the script excercise1.m. You can modify this script and add your code to it. Follow the steps below:

Part 1: Training and testing an Image Classifier

Stage A: Data preparation and feature extraction

The data provided in the directory data consists of images and pre-computed descriptors for each image. The JPEG images are contained in data/images. The data consists of three image classes (containing aeroplanes, motorbikes or persons) and `background' images (i.e. images that do not contain these three classes). This data is divided as:

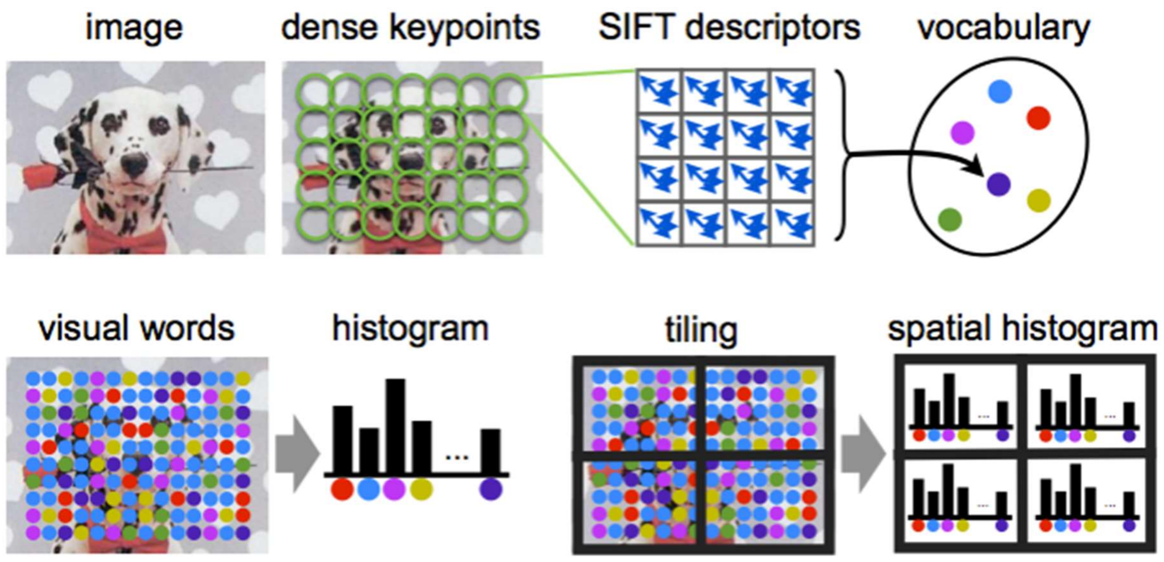
	aeroplane	motorbike	person	background
Training	112	120	1025	1019
Test	126	125	983	1077

Total	238	245	2008	2096

The list of training images for each class is given in text file "data/*_train.txt", where * is the class name. The list of testing images for each class is given in text file "data/*_val.txt", where * is the class name.

A descriptor for an image is obtained as a statistics of local image features, which in turn capture the appearance of a large number of local image regions. Mapping local features to a single descriptor vector is often regarded as an encoding step, and the resulting descriptor is sometimes called a code. Compared to sets of local features, a main benefit of working with codes is that codes can be compared by simple vectorial metrics such as the Euclidean distance. For the same reason, they are also much easier to use in learning an image classifier.

In this part we will use the Bag of Visual Words (BoVW) encoding. The process of constructing a BoVW descriptor starting from an image is summarized next:



First, SIFT features are computed on a regular grid across the image ("dense SIFT") and vector quantized into visual words. The frequency of each visual word is then recorded in a histogram for each tile of a spatial tiling as shown in the figure above. The final feature vector for the image is a concatenation of these histograms.

QA1: Why is the spatial tiling used in the histogram image representation?

Stage B: Train a classifier for images containing aeroplanes

We will start by training a classifier for images that contain aeroplanes. The files data/aeroplane_train.txt and data/aeroplane_val.txt list images that contain aeroplanes. Look through example images of the aeroplane class and the background images by browsing the image files in the data/images directory.

The aeroplane training images will be used as the positives, and the background images as the negatives. The classifier is a linear Support Vector Machine (SVM). Train the classifier using the function trainLinearSVM by following the steps in excercise1.m.

We will first assess qualitatively how well the classifier works by using it to rank all the training images. View the ranked list using the provided function displayRankedImageList as shown in excercise1.m.

QB1: Show the ranked images in your report.

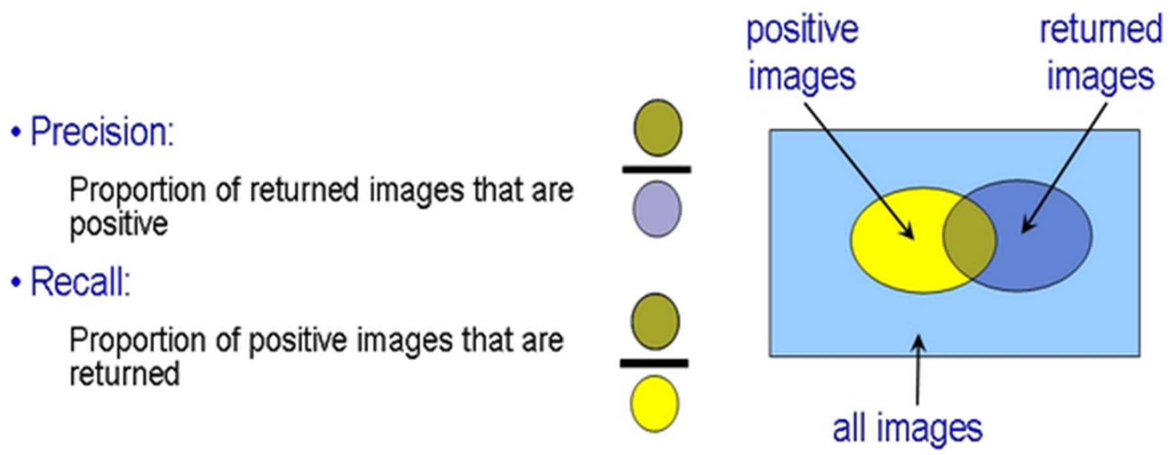
Use function displayRelevantVisualWords to display the image patches that correspond to the visual words which the classifier thinks are most related to the class (see the example embedded in excercise1.m).

QB2: In your report, show relevant patches for the three most relevant visual words (in three separate figures) for the top ranked image. Are the most relevant visual words on the airplane or also appear on background?

Stage C: Classify the test images and assess the performance

Now apply the learnt classifier to the test images. Again, you can look at the qualitative performance by using the classifier score to rank all the test images. Note the bias term is not needed for this ranking, only the classification vector w.

Now we will measure the retrieval performance quantitatively by computing a Precision-Recall curve. Recall the definitions of Precision and Recall:



positive images
returned images
• Precision:
Proportion of returned images that are positive
• Recall:
Proportion of positive images that are returned
all images

Stage D: Learn a classifier for the other classes and assess its performance

Now repeat stages (B) and (C) for each of the other two classes: motorbikes and persons. Re-use your code after changing the dataset loaded at the beginning of stage (B). Remember to change both the training and test data.

QD1: In your report, show the top ranked images, precision-recall curves and APs for the test data of all the three classes (aeroplanes, motorbikes, and persons). Does the AP performance for the different classes match your expectations based on the variation of the class images?

QD2: For the motorbike class, give the rank of the first false positive image. What point on the precision-recall curve corresponds to this first false positive image? Give in your report the value of precision and recall for that point on the precision-recall curve.

The Precision-Recall curve is computed by varying the threshold on the classifier (from high to low) and plotting the values of precision against recall for each threshold value. In order to assess the retrieval performance by a single number (rather than a curve), the Average Precision (AP, the area under the curve) is often computed. Make sure you understand how the precision values in the Precision-Recall curve correspond to the ranking of the positives and negatives in the retrieved results