Computervision Lab 6
Classification

David Van Hamme, Sanne Roegiers

March 5, 2019

1 Feature vectors

Many applications require observations to be assigned to categories or classes. For example, a doctor may look at a magnetic resonance image (MRI) of a patient's brain and classify the brain as healthy or pathological based on the prevalence or absence of certain distinctive qualities or features. In computer vision, such classification problems are very common. In order to automate the assignment of a class label to an observation, typically a series of numbers is first calculated that describe the presence of features in the observation. This set of descriptive numbers is called a *feature vector*. Features may relate to average color values, presence of edges, strength of texture, ...

2 Exercise 14

In this exercise you will make a set of DoG (differential of Gaussian) filters, using the code from Lab 5 exercise 10. You will use it to construct a feature vector for each 16x16 pixel block in an image.

- Make DoG filters in 2 scales and 6 orientations.
- Filter *road1.png* with each of the filters. This gives you 12 response values per pixel.
- Divide the image into 16 by 16 pixel blocks and compute for every filter the maximum absolute value of the response within each block. This gives you 12 numbers per block.
- Using the block labels in *road1_blocks.png*, compare the mean value of each feature for blocks containing road markings, and blocks containing no road markings. This should result in a plot similar to the one in Figure 1.

3 Random forest

A random forest classifier is a classifier that consists of a number of decision trees. Each decision tree makes a hierarchical set of decisions, each decision based on a simple criterion on only a few feature values, to arrive at a class label for the full feature vector. The random forest introduces random variations in the construction of the trees and takes a majority vote of their outcomes. This solves a major drawback of decision trees: their poor generalization to data slightly dissimilar from the training data.

4 Exercise 15

Using the feature vectors as calculated in Exercise 14, you will train and test a random forest classifier to classify blocks as containing road marks or not containing road marks.

- Calculate feature vectors for each 16 by 16 block of images in *road1.png* through *road4.png*.

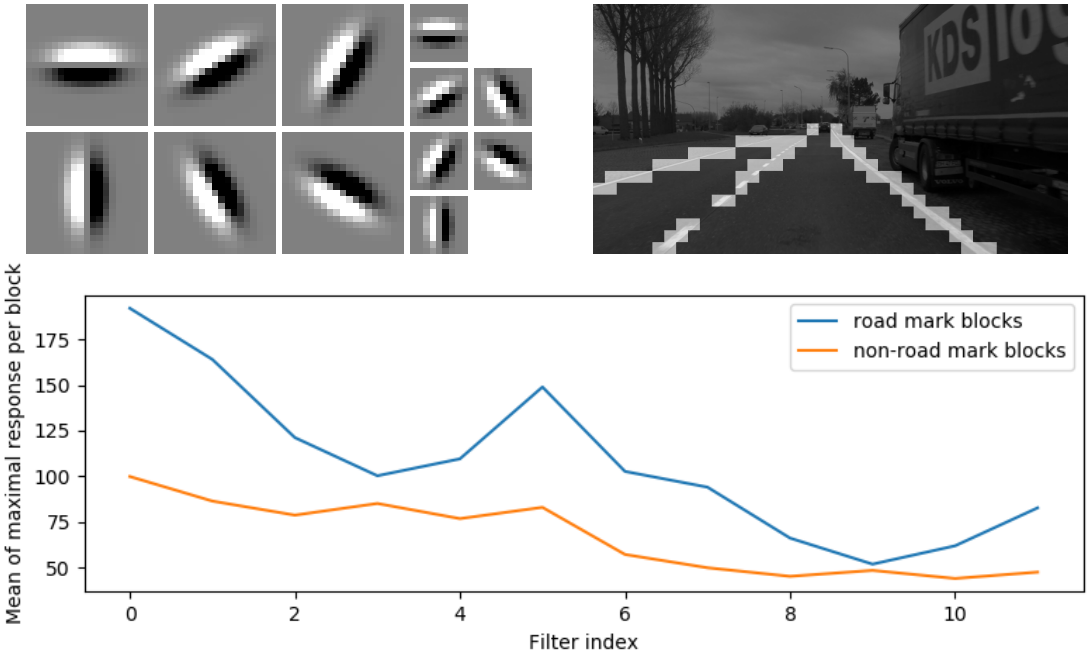

Figure 1: Filter bank (top, left), applied on labelled image (top, right), and mean feature vector values (bottom).

- Using the labels in *roadN_blocks.png*, construct a **balanced** training set consisting of equally many marked blocks as unmarked blocks.
- Train a random forest classifier with 10 trees. Avoid overfitting to this small dataset by choosing a suitable minimum leaf node size (or size fraction).
- Use the trained classifier to predict a label for each block, and visualize the labels in an overlay.

In python, you may choose to use **sklearn.ensemble.RandomForestClassifier** as it is easier to use than the opencv implementation. In C++, you can use classes **RTrees** and **StatModel** from the machine learning (**ml**) library.

The classifier should work reasonably well in the lower half of the images, but generate a lot of false positives in the upper half of the images. In practice this would not be a great limitation as it is pointless to run the classifier above the horizon level.

5 Cross-validation

Running the classifier on the data it was trained on usually gives unrepresentatively high performance figures. In order to mitigate this, a common practice is to split the training set into multiple complementary subsets, train the classifier on only some of the subsets and evaluate it on the remainder. For example, in five-fold cross validation, the dataset will be split into five parts of roughly equal size. For each part, the labels will be predicted by a classifier trained only on the four other parts. The average performance of the five classifiers evaluated on the four different unseen parts then gives a much more representative number for the expected performance on new data.

6 Exercise 16

Use four-fold cross validation with the random forest of Exercise 15 by training on only three of the images and evaluating on the fourth, for all four combinations. Use the ground truth block labels to obtain average performance figures, and compare this to the performance of the classifier trained and tested on all four images. Split your evaluation into precision (percentage of predicted blocks with road marks that actually contain road marks) and recall (percentage of the actual blocks with road marks that are predicted as such).

7 Report

Write a short report about how you solved the exercises. Include in this report your input and output images. Describe every new function that you used. Explain the basic algorithm and purpose of the functions and clarify the parameters and how you selected them. Upload your report in the form of **LabX_name.pdf** to the dropbox on Minerva.