COMPUTER VISION - LAB 7

M. Carraro, P. Zanuttigh, S. Ghidoni, G. Agresti

Topics: Machine Learning / Deep Learning

Goal: Train and test a CascadeClassifier to detect cars in a video. Compare the result with the YOLO Deep Learning based object detection algorithm.

In this lab, you have to train and test a CascadeClassifier (based on the Viola and Jones algorithm) to detect cars. Differently from the other lab assignments, an initial package is provided.

The code is written exploiting polymorphism, thus, you should just implement the derived classes extending a pure virtual base class. The instructions to follow to train and test it are in this PDF.

The source code package is composed of the following files:

- include folder
  - **classifier.h**: You should not modify this file (at most, you can add lines to it, defining more specialization classes). It contains the definition of the base class you are asked to extend (**ClassifierContainer),** the YOLO specialization (**YOLOClassifierContainer**) and the class you need to edit (**CascadeClassifierContainer**).
- src folder:
  - **lab7_test.cpp:** you should not modify this file. The only modification allowed are to add/change the input video (have a look at the video_caps variable) and to the array of ClassifierContainer objects (you should add the optional additional classes you define in classifier.h, if you do not add any, it should work as it is). Please, consider that adding classes is not required for the homework, it is an optional step if you want to try other classifiers. You can also modify the code for results visualization.
  - **classifier.cpp**: In this file you should put the code for the derived class
- data folder:
  - yolo-voc.cfg: file needed by the YOLO classifier containing its configuration , it must be passed as a command line argument
  - video.mp4: the input video

CascadeClassifier Training and Testing instructions:

- To train a classifier you first need a dataset of cars. The dataset can be downloaded from moodle or from \\nas2\datilab\ENI\lab7 . The dataset is composed of multiple folders, have a look at its contents. For this lab, the needed positive and negative files have already been computed (files PNGImages/negatives_all.txt and PNGImages/positives_all.vec). The positive file has been computed by using the `opencv_createsamples` binary.
- To actually call the training binary, you need to invoke the `opencv_traincascade` application. This application requires a bunch of parameters. To speed things up, we suggest

you to use the following parameters (go to the documentation and read the info about them!)

- -data <output folder>
- -vec <your positives_vec file>
- -bg <your_negative file>
- -numStages <number of stages of the cascade classifier. Since you do not have too much time (and hardware capabilities !), start with 8, then add new ones if needed (the first k stages are kept, so if need k+m, OpenCV will start to train directly from the (k+1)-th stage the next time you call the binary).
- -w (width rescaled sample, usually 24)
- -h (height of the rescaled sample, usually 24)
- -NumPos (positives extracted from the positives file each time, it should be a number less than the total number of positives, e.g., 1000)
- -NumNeg (negatives extracted from the negatives file each time, it should be a number less than the total number of negatives, e.g., 1370)

Once you trained the classifier, the result will be a `cascade.xml` file, which should be provided to the application by using the command line arguments.

After testing the cascade classifier, you can compare with the YOLO detection algorithm. The application needs 2 files to allow YOLO to work. One is the configuration file (*yolo-voc.cfg*) , the other is the *yolo-voc.weights* file which contains the weights and biases of the CNN (the network has already been trained). In order to filter the detections by YOLO you can apply a threshold on the confidence of the detection (notice that it can be passed as a command line argument but must be set also at the end of the yolo-voc.cfg file). This file can be downloaded from Moodle or from \\nas2\datilab\ENI\lab7 .

To test the classifier you'll need also to fill the TO-BE-COMPLETED comments in classifier.cpp

OUTPUT SAMPLE:

CASCADE YOLO