Assignment #03 and Final Report

(Document Updated on April 26, 2018)

IT University of Copenhagen (ITU)
Introduction to Image Analysis and Machine Learning (IAML)
(Spring 2018)

Deadlines: May 08, 2018 and May 22, 2018

Purpose The last course activity is about creating a car detector using binary classifier. You will create the classifier using a set of image analysis and machine learning methods presented in the course. It is essential to present how you: (i) preprocessing the images/videos; (ii) increased the dataset; (iii) labelled the features as "non-car" or "car"; (iv) created the classifier model; (v) trained and tested the classifier; and (vi) evaluated your car detector.

Groups You have to use the same group of students as in the previous assignments. *You can NOT mix bachelor and master students in a group.* As any other workgroup, it is essential to distribute the workload among the members. Notice we expect that master students present better performance and robustness of your code and quality of your reports than bachelor students. Your group ID is available on learnIT.

Reports You have to write a two pages report (for Assignment #03) with the planning of your classifier/detector, and a maximum fifteen pages Final Report with the results of your implementations. You have to describe what the overall method is, what your assumptions are, and why your method fails/works? Your Final Report must have information in which other students can reproduce your results, but not necessarily in chronological order. *Your reports should contain ONLY images that you have generated with the results of your methods.* Add as appendix the reports you have written for Assignments #01, #02 and #03. In the following links, you can find information on how to write a good report: link 1 and link 2.

Your Results *The results of your experiments will be saved and shared with your reports.* You will compact all generated files in two zip files called `Assignment03_Group_ID.zip` and `FinalReport_Group_ID.zip`. Make sure you have submitted your reports, Python project, source code, output videos and exported Numpy files in a zip file.

Tasks, Exercises and Extra This document contains information that will guide you through the assignment. *There are questions that you have to answer in your reports.* All exercises are mandatory in this assignment, except the ones indicated with $^{(Extra)}$. Extra exercises are optional activities that you help you to get a more in-depth insight into the task, and they will also help you to achieve a better grade.

Deadline This activity spans three weeks. Only ONE member of your group should hand the reports, code, and videos for Assignment #03 on May 08, 2018, and for Final Report on May 22, 2018 via learnIT.

Assignment 3.01. In this exercise, you will plan a car detector using a binary classifier. You will present the planning of your classifier as Assignment #03 (deadline May 08, 2018).

Feel free to propose to use any machine learn approach (e.g., logistic regression, k-nearest neighbors, support vector machine, neural network) presented during the course to create the best car detector as possible. It is essential to explain why you have decided to use an approach instead of another one (e.g., SVM vs. NN). In this document there are some hints (based on the developed exercises) that will help you to choose a specific method.

We have provided a small dataset divided in two subsets, namely: (i) *positive*, a subset with 375 images (i.e., 64×64 resolution) with cars; and (ii) *negative*, a subset with 5 images (i.e., 3456×2304 resolution) without cars. We also provided five videos in the folder `inputs` that you will use to train and test your classifier. Figure 1 shows five positive images with cars and Figure 2 shows three negative images.

Figure 1: True positive images with cars from our dataset.

Figure 2: Images without cars from our dataset.

In the first report, you have to describe the classifier model(s) used to detect cars based on the dataset provided in supporting material.

(a) **Define the classifier model:** At this step, you have to decide what classification model you will use to classifier images as "non-car" or "car". You need to extract features from

images to define a descriptor. You can use HOG features (see Exercises 9.03 and 10.01) or you can extract features using an image analysis approach (see Assignment 2.01). You also have to decide what machine learning approach you will use in your classifier:

- **Logistic regression:** See Exercises 9.01 and 9.02;
- **Support Vector Machine:** See Exercises 10.1 and 10.2;
- **Neural Network:** See Exercise 11.01; or
- **K-Nearest Neighbors:** Lecture slides.

Hint: You can increase the chance of getting a better grade (for Final Report) if you implement more than one classifier model and compare their performances.

Assignment 3.02. In this exercise, you will implement the binary classifier proposed in Assignment #03. You will also implement a detector to detect cars in the videos available in supporting material (see folder `inputs/videos`), as shown in Figure 3. Use the files `As301_classifier.py` and `As301_detector.py` to answer this exercise.

Figure 3: (a) Example of one frame from the video "Cars_01.mov" and (b) two cars detected using a binary classifier.

You will present the development and evaluation of your "*binary classifier*" and "*car detector*" in the Final Report (deadline May 22, 2018). In the second report, you have to describe all steps done to implement and evaluate your classifier and detector in details.

(a) **Load the dataset:** You have to load both subsets (*positive* and *negative*) using the function `loadDataset()`. Notice you will use 64×64 positive and negative images to train your binary classifier. All positive images are 64×64 resolution in our dataset. However, the negative images are of different resolutions. In the following, you will generate randomly the negative training windows from the initial negative training dataset.

(b) **Generate a sample of negative images:** In this exercise, you have to make a function `sampleNegativeImages()` to generate a fixed set of $N > 200$ patches sampled randomly from 5 car-free training photos provided in the initial negative set. For example, Figure 4 shows two examples of squared windows selected from a negative image.

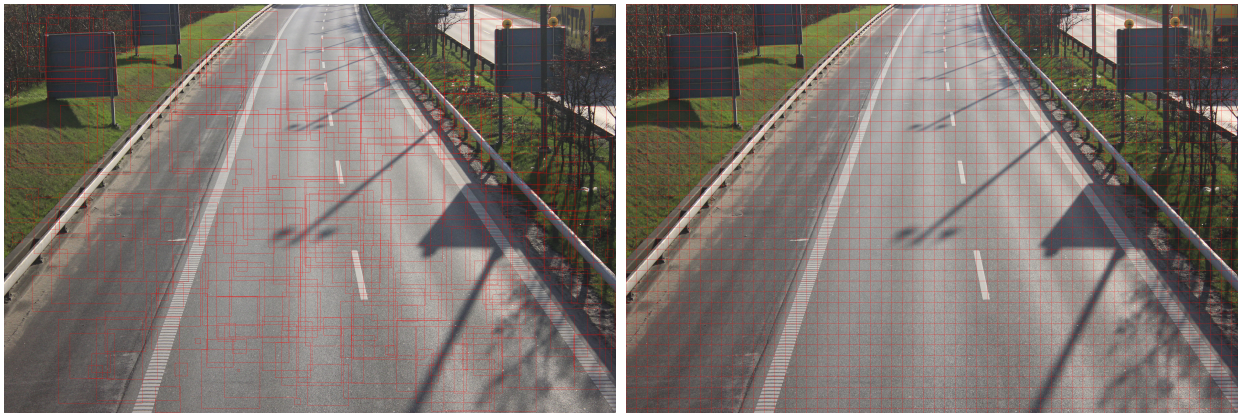

Figure 4: (a) Example of 250 windows with different resolutions selected randomly from a negative image and (b) Example of 2166 windows with 60×60 selected in sequence from a negative image.

(c) **Increase the dataset:**$^{(Extra)}$ You can increase the car dataset by generating new positive images by:

- Using linear transformation (e.g., rotation, shear, scale, reflection) on the available positive subset;
- Using any open-source car dataset available on the Internet. *Hint:* It is necessary to resize the images to 64×64 resolution; or
- Using the script `warmUpAnnotations.py` to create select new regions of interest (i.e., annotations) from the videos available in our dataset.

□ **Output:** `Cars_XYZ_Extra.png`.

Hint: In Final Report, describe the differences in terms of performance and accuracy when you train your classifier using the original dataset and using the extended dataset.

(d) **Label the dataset:** You have to label each sample of both subsets with a distinct number. For example: **+1** for positive samples; and **-1** for negative samples.

(e) **Create the classifier model:** Feel free to create your classifier model manually or using a Python Framework (e.g. scikit-learn, scikit-image, keras, tensorflow, among others).

(f) **Train the classifier model:** Use the dataset (positive and negative samples) and the labels (+1 and -1) to train your binary classifier. The training process depends on the classifier model that you have selected in the previous steps. Consult the supporting material of Exercises and Assignments to understand how to train your classifier model.

(g) **Detect cars in a video:** You have to detect cars of different sizes in all videos available in the folder `inputs/videos`. You can use any preprocessing approach to improve the car detection. For example, you can use both "*image pyramids*" or "*sliding window*" to detect cars in images at various scales and locations. Remember to describe in the

Final Report how your car detector works. Use the file `As301_detector.py` to answer this exercise.

- □ **Output:** `Detection01.mov`, `Detection02.mov`, `Detection03.mov`, `Detection04.mov` and `Detection05.mov`.

(h) **Evaluate your classifier model:** Develop a function which print confusion matrix (see Exercise 9.01) of your car classifier. You have to calculate the error (RMS) based on the ground truth data ($N > 20$ frames extracted from the videos) with the classified data. You have to measure the number of *false positives*, *false negatives*, *true positives*, and *true negatives* of your classifier.

Your function also have to estimate the accuracy through:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}.$$

(i) **Plot graphics:** In your report, you have to include several plots to represent your dataset, training process, test process and detection. In the following, there are some suggestions of plots, but you don't need to generate all of them for your report:

- A plot to show the features of positive and negative samples in two different colors;
- A plot with the dataset and the decision boundary in which split the sample into two classes (i.e., non-cars and cars);
- The correlation matrix of the features vectors (see Exercise 3.04).

(j) **Define the region of interest:**$^{(Extra)}$ Define a Region of Interest (ROI) to avoid processing the entire image and, at the same time, improve the car detection and speed up the time processing. As the cars ONLY pass through the road, your detector should search for cars on it. Figure 5 shows two examples of regions of interest in red.

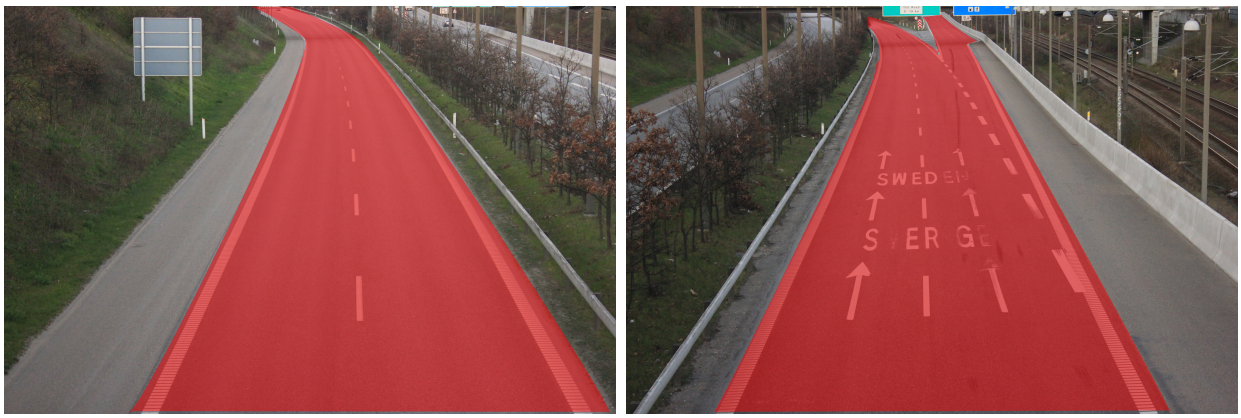

Figure 5: Regions of interest over the road.

(k) **Overlapping bounding box:**$^{(Extra)}$ Your car detector can detect more than one bounding box for a single car, as shown in Figure 6. Improve your car detector to remove the overlapping bounding box from the detected cars.

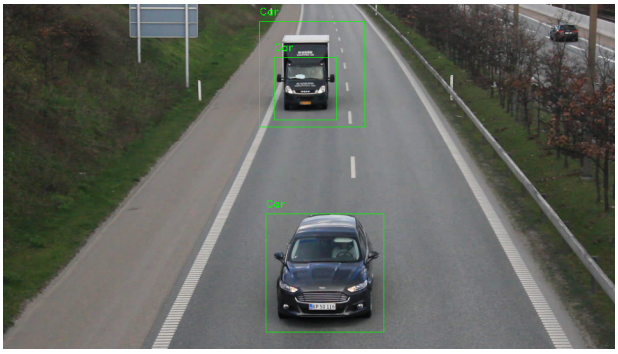

Figure 6: Example of overlapping bounding box.

(l) **Use a training and test subsets:**$^{(Extra)}$ You have used the entire dataset to train and test your car detector. Now, you will split the dataset into the training and test datasets (e.g., 70% training and 30% test). Calculate and compare the accuracy of your car detector for the training data, test data and the entire dataset.

(m) **Predict the destination of the car:**$^{(Extra)}$ The video `Cars_05.mov` has three different routes. You have to predict if the detected car is going to: (i) *Copenhagen* [left road]; (ii) *Malmö* [right road]; or (iii) the *Airport* [turn right]. For example, you can use different colors to draw the rectangle around the detect car.

- **Output:** `Destination_Extra.mov`.

Checklist #01. Before you submit the Assignment #03, be sure the you have include the following items in your report.

- ☐ **Report Front Page:** First page of your report must contain your names and emails
- ☐ **Report Size:** Be sure that you have written a concise report (maximum 2 pages)
- ☐ **Images:** You have used ONLY the images that you generated from your experiments
- ☐ **Zip File:** Zip file is named as `Assignment03_Group_ID.zip` (replace by your ID)

Checklist #02. Before you submit the Final Report, be sure the you have make all mandatory exercises and generated all output files (i.e. videos, images, and Numpy files).

- □ **Report Front Page:** First page of your report must contain your names and emails
- □ **Report Size:** Be sure that you have written a concise report (maximum 15 pages)
- □ **Images:** You have used ONLY the images that you generated from your experiments
- □ **Source Code:** Only the Python scripts you have created or changed
- □ **Appendix:** Include as appendixes the reports submitted for Assignment #01, #02 and #03
- □ **Files Generated During the Exercises:**
  - □ **Assignment 3.01. (c):** `Cars_XYZ_Extra.png`
  - □ **Assignment 3.01. (g):** `Detection01.mov`
  - □ **Assignment 3.01. (g):** `Detection02.mov`
  - □ **Assignment 3.01. (g):** `Detection03.mov`
  - □ **Assignment 3.01. (g):** `Detection04.mov`
  - □ **Assignment 3.01. (g):** `Detection05.mov`
  - □ **Assignment 3.01. (m):** `Destination_Extra.mov`
- □ **Zip File:** Zip file is named as `FinalReport_Group_ID.zip` (replace by your ID)