Introduction to Computer Vision Assignment #4 (Final Project)

Due 6/9(Wed), 2021

Sliding Window-based Object Detection

The goal of the project is to learn and implement a sliding window-based object detection algorithm. This is a competition. So, you have to design your system carefully to make it perform in the best mode.

- Read the following materials:
  [1] Paul Viola and Michael Jones, Rapid Object Detection using a Boosted Cascade of Simple Features, CVPR 2001
  [2] Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR 2005
  [3] Qiang Zhu et al., Fast Human Detection Using a Cascade of Histograms of Oriented Gradients, CVPR2006
  [4] Y. Mu, S. Yan, Y. Liu, T. Huang, B. Zhou, Discriminative local binary patterns for human detection in personal album, CVPR 2008
  [5] Ch. 17 of *Computer Vision: A Modern Approach*, 2nd Ed., by Forsyth and Ponce, Pearson, 2012
  [6] Ch 14.1 of *Computer Vision: Algorithms and Applications* by R. Szeliski, Springer, 2011

- Your goal is to implement a window-based Human detection algorithms using HOG feature and evaluate them.
- Dataset and code: will be provided separately

Implementation detail

1. Implement your own Human detection algorithm using HOG and SVM [2]:
   a. Use HOG feature (MATLAB: `extractHOGFeatures(I)`, VLfeat: `vl_hog(im, cellSize, 'verbose')`.
   b. Train a linear SVM using the INRIA person dataset (positive samples and negative human samples).
   c. Implement the overall sliding window-based detection pipeline using the classifier you have trained. You can start with the single-scale detector and then extend it to full multi-scale detector. You may need to apply the non-maximum suppression step to remove duplicated detections for the final output.
   d. Test your detector on the test dataset of the INRIA person dataset.
   e. Evaluate the performance using ROC curve, precision-recall curve, and average precision (use `evaluate_detections.m`).
   f. You can visualize the detection results in each image (use `visualize_detections_by_image.m` or `visualize_detections_by_image_no_gt.m`).

2. Make your algorithm fast by a cascade architecture [1].
   a. Implement the cascade of HOG algorithm described in [3].
   b. Test your new algorithm on the test dataset and evaluate the performance as in 1-e.
   c. Compare and discuss the results with the previous ones in terms of accuracy and speed.

3. (Option) Try to use the LBP (Local Binary Pattern) feature [4] (Matlab: `extractLBPFeatures(I)`, `VLFeat: lbp.h`) instead of HOG. Compare the results with those

you obtained in 1 & 2.

4. How can you improve the performance of your algorithm in terms of either accuracy or speed? Propose your own idea and show it.

Warning: Do not use the publicly available code of whole detection system, and do not copy codes of your colleagues including who have taken this course before. Some codes of individual parts can be downloaded from the web, but you have to state clearly where you get them and how you use in your system.

Submission instructions: what to hand in

- Upload the electronic file that includes the report and source code in a single zip format with the name "ICV21_Assignment #4_yourname.zip" on the ETL class homepage.
- The report that includes the brief description of your algorithms, results, and discussions
- Do not use existing codes unless explicitly allowed to do so.