ECSE 415 Course Project - Counting People in a Shopping Mall

November 17, 2020

The goal of this project is to create an algorithm to count the number of people in a surveillance image of a mall. The algorithm must be based on a Support Vector Machine person detection technique.

The dataset used in this competition is the "mall" dataset, described in the paper by Loy, Chen Change, Ke Chen, Shaogang Gong, and Tao Xiang entitled "Crowd counting and profiling: Methodology and evaluation."which was published in "Modeling, simulation and visual analysis of crowds", pp. 347-382. Springer, New York, NY, 2013. Students should read this paper to learn about different approaches to crowd counting. The paper mainly covers the use of regression based methods (whereas we will be using detection based methods), but many of the ideas in the paper are still relevant to this competition. An example of the images in this dataset is shown in the figure below. There are 2000 images in the dataset, each of size 640x480 pixels.

The approach taken to do the person counting will be to use a person detector to produce bounding boxes around each person in the image. This is followed by a process to remove duplicate bounding boxes (where multiple detections are made of the same person). Then the number of bounding boxes detected is determined.

There are two parts to the project:

- 1. Choose an existing person detection technique, such as YOLO, Faster-RCNN, where

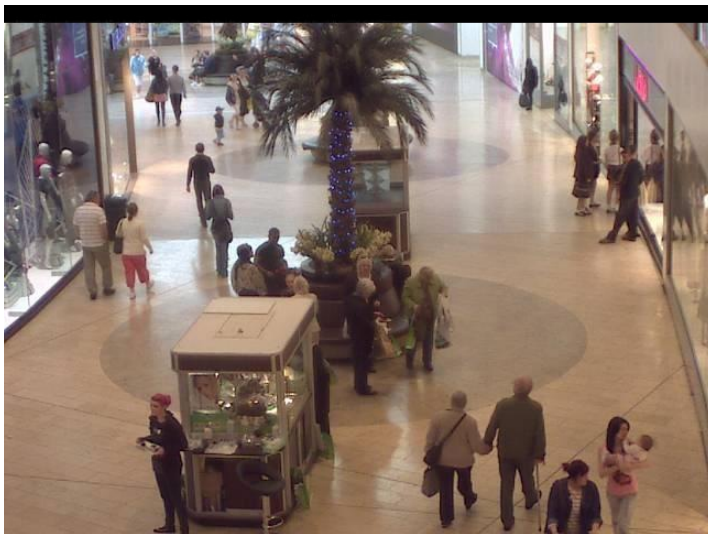

Figure 1: An example of an image frame from the "mall" dataset.

code is already available (e.g. you can find useful code on the Facebook Detectron2 site). Use this to obtain bounding boxes around people in the mall images. Then, write python code to eliminate duplicate detections (if any) and to count the bounding boxes.

- 2. Implement your own person detection algorithm, based on a Support Vector Machine (such as that used in Assignment 3). Get the positive examples by cropping out image patches containing people from the mall images. Negative examples can be obtained by extracting randomly located patches from the images (that do not intersect person patches). You can use the person detector implemented in part 1 to identify the person patches in each image. From these patches you should compute suitable features to feed into the SVM. Typically the raw intensity values will not work too well. You could try HoG or Haar features. You could also try Local Binary Pattern (LBP) features. These were not covered in class, but are fairly simple and fast. Train your SVM using the extracted positive and negative examples. It is up to you to determine how many examples you need to do the training. Once you have the SVM-based person detector working, you can use it in place of the pre-made detector in part 1, to make a new version of the complete person counting system.

You should write a report describing the systems you implemented. The report should also provide an evaluation of the person detectors that you implemented. Since we do not have ground truth person bounding boxes for this dataset, you can use the bounding boxes provided by the pre-made detector of part 1 as the ground truth (after removing duplicate detections). Use the "Intersection-over-union" (IoU) metric to score how well your detector works compared to the ground truth. The report should be accompanied by the source file (Jupyter notebook preferred) so that the grader can run your implementation.

The evaluation just mentioned only considers the quality of your SVM person detector. It does not evaluate the person counting performance. To evaluate the person counting, we will use a "Kaggle Competition". This is an on-line competition where you submit a file containing your estimates of the person counting for each image in the dataset. This is compared against the ground truth values and the accuracy of your estimate is ranked relative to other group's submitted estimates. This will give you a feeling for how well your method is working.

The URL for the Kaggle competition is:
`https://www.kaggle.com/t/31ea21c9c6af41e2ae80eb66f696418b`

Go to the competition web site and download the dataset, and get some details on how to submit your estimate file. The data zip file includes a sample submission file (containing random answers) that you should use as a template for making your own submissions. You can make as many submissions as you wish, but only the top-ranked one will appear in the final ranking. Note that you will need to create an account on the Kaggle site. Each group member should make an account, and then merge with the other members to form a single team. Submissions should only be done by a single team rather by each group member.

The report should include the performance of the person counting as expressed by the Kaggle competition's evaluation and ranking.

The grade for the project will be based on the following items:

- Demo - the grader will check and run the provided code to verify that it (more-or-less) detects people, eliminates duplicate detections, and counts the detections.

- Report - the grader will look for the required report elements: description of algorithms used (with reference to code), evaluation (using IoU metric and the Kaggle performance), discussion of issues arising from the design (e.g. difficulties encountered, effectiveness of different types of features, etc).

The project grade will not depend on the Kaggle ranking. Getting a high ranking will give your group bragging rights among your classmates, however!

The project must be done in groups of 3 or 4 students. Each student must do a significant part of the project, and the role of each student should be stated in the report. All group members will receive the same grade.