Counting Number of Cars in Parking Lot

Siddharth Banra
2016csb1061

Department of Computer Science,
IIT Ropar

1 Introduction

Parking is becoming a major problem especially with the increasing number of vehicles in the metropolitan areas. Even in malls and busy roads it very difficult to find parking spots.Big Malls use workers who guide the customers to the vacant spots or they have to find it themselves. So finding a good spot to park a car is very tedious job .Many existent systems are using sensor-based techniques such as ultrasound and infra-red-light sensors. But this type of systems may requires high costs for installation and maintenance. Therefore, there is a growing interest in the use of vision-based systems .This may solve the problem of parking cars in very low cost.
Our aim is to provide a vision based solution to provide total number of cars in the parking lot so that we can find the number of slots free in the parking . Our system require less number of cameras which helps to find the desired output in very low cost.

2 Brief Idea

The aim of the project is to count number of cars in a given image of a parking lot.So, the to detect a car in an image ,I have used a liner support vector machine(SVM) and trained it with HOG features extracted from patch images. I have used CNRpark and GTI cars data set. I have trained the SVM with cars and non cars patch images. After training the svm ,to detect cars , I have used sliding window technique to detect cars in the parking lot image as we do not know the position of the car in an image ,I am moving the window by certain value and checking the window with the svm if the window contains a car or not, and to remove the overlapping bounding boxes I have used Non-maximal-suppression. This concept requires less budget as it requires only the image of the car.

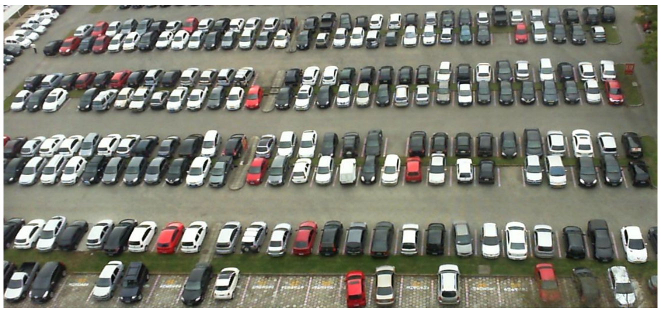
Figure 1: parking lot image

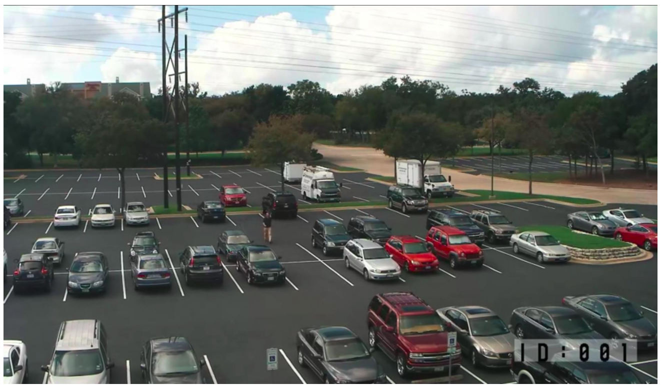

Figure 2: Parking lot image

3 Algorithm

In this section I have described the every method that I have used in the project.

3.1 Features Extraction

3.1.1 HOG Features

The idea of HOG is,instead of using each individual gradient direction of each individual pixel of an image, I have grouped the pixels into small cells of 16 x 16 pixels. For each cell, we compute all the gradient directions and grouped into a number of orientation bins. We sum up the gradient magnitude in each sample. So stronger gradients contribute more weight to their bins, and effects of small random orientations due to noise is reduced. This histogram gives us a picture of the dominant orientation of that cell. Doing this for all cells gives us a representation of the structure of the image. The HOG features keep the representation of an object distinct but also allows for some variations in shape. We can specify the number of orientations, pixels per cell, and cells per block to computer the hog features of a single channel of an image. The number of orientations is the number of orientation bins that the gradients of the pixels of each cell will be split up in the histogram. The pixels per cells is the number of pixels of each row and column per cell over each gradient the histogram is computed. The cells per block specifies the local area over which the histogram counts in a given cell will be normalized.

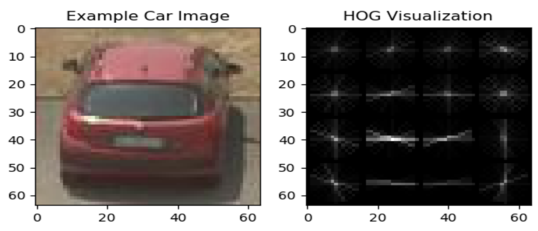

Figure 3: HOG features of car

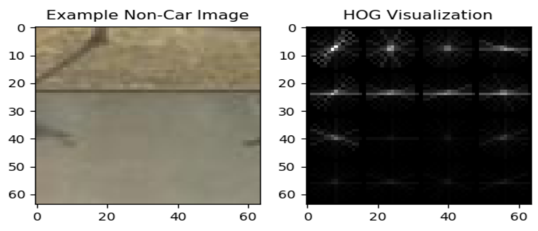

Figure 4: HOG features of non car

3.1.2 Colour Histogram

As many cars have similar colour but different shape so I have used another feature That I have used to train my classifier is a histogram that captures the general distribution of colours within the image,irrespective of the actual location of the image.

3.1.3 Spacial Features

I have also used spacial features of the image, this helps the classifier to sense where the different colours are within the image.

3.2 Training SVM

I have used 17,208 car images and 13,153 non car image to train the SVM classifier. The dataset has 64X64 pixel size image patches of cars and non cars. I have trained the classifier with the hog features of the cars and non cars images so that it can differentiate between cars and non car images. I have used Linear SVM classifier from SKlearn library.I have used the Standard Scaler from SKlearn library to make a scaler based on the mean and variance of all the features in the data set. The StandardScaler normalizes features by removing the mean and scaling it to unit variance. I used 80 percent of the dataset to train the classifier. The remaining 20 percent is used to determine the accuracy of the classifier.

3.3 Sliding window

Sliding window is a technique in which we slide windows of a particular size across the image with certain increment in x and y direction. As we do not know the position of the cars in the image ,so sliding window technique will help us to search cars in the image . The size of the sliding window is fixed and is set by the user according to the average size of the cars in the image. Now all these window are passed to a function called search window which checks each and ever window with the svm classifier to check if it is a car or not. This function returns windows which are approved by the classifier to have cars in them. As we are incrementing the windows with length less than the width and height of the window ,therefore, these windows tend to overlap. So ,there are many bounding boxes around the cars which have been approved by the classifier. These windows are will be removed by Non maximal suppression.

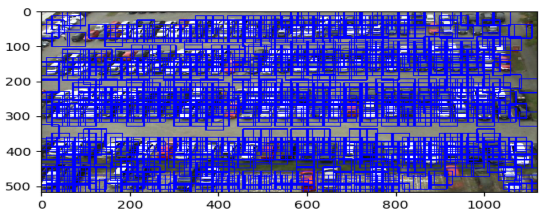

Figure 5: image before nms

3.4 Non maximal suppression

There are multiple sliding windows around the detected cars sue to to the fashion in which we are incrementing the windows, we can also not ignore the overlapping technique of the sliding window as we do not know the position of the the cars in the image. So , to remove the windows which are overlapping on the same car image we will remove them by non maximal suppression. This function will take input the bounding boxes which got approved by the svm and bounding box area overlap threshold. We will remove the multiple overlapping boxes and replace it with one bounding box. This will help us to identify the cars with more ease. This function returns the actual boxes which have cars. The count of the boxes gives the number of cars in the image.

4 Results

Training Classifier with cars and no cars images gave 97 percent accuracy with the following parameters.
orientation bins =30
pix per cell=16
cell per block =2

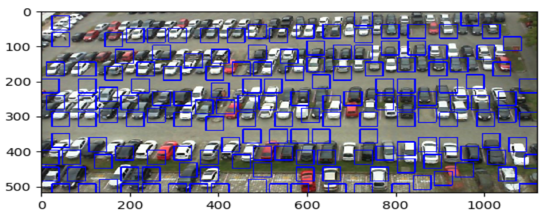

Figure 6: Output image after nms

Including the spacial and colour histogram features.
the out.p file is trained with these parameters.
As my data set was not labelled I have manually counted these values.
The number of cars in the image were - 210
Total cars detected - 185
True positives - 157
False positive - 28
False negative - 8

5 Conclusion

In this project of counting cars in an image , the out put result is good but it can be made better. Due to the small size of the data set the trained svm gave decent results most of the time but there were few cars which were not detected and there were few non car objects which were detected as cars. The results would have been great if we could extract more features from the images. But the overall output was good. Svm could have been trained better if I had a larger dataset. Using spacial features and colour histogram along with the HOG features boosted the results of the project. It gave good results in particular images. The sliding window technique gave decent results but was not accurate in finding the exact location of the cars in the image. As the size of the bounding boxes were fixed for a particular image there fore if the sizes of the cars in the image vary too much then the results may come out to be bad .Non maximal suppression reduces the false positives but while refining the boxes it would some times removes the wrong boxes.

6 References

[1] Nicholas True *"Vacant Parking Space Detection in Static Images"*.

[2] Imen Masmoudi,Ali Wali,Anis Jamoussi, Adel M. Alimi *"Vision based System for Vacant Parking Lot Detection"*.

[3] Websites reffered
`https://scikit-learn.org/stable/modules/generated/sk`
`https://www.learnopencv.com/histogram-of-oriented-gr`
`https://www.Wikipedia.com`