Vehicle detection

The goals / steps of this project are the following:

- Perform a Histogram of Oriented Gradients (HOG) to extract features and also use color features on a labeled training set of images and train a classifier Linear SVM classifier
- Data is normalized and selected for training and testing
- A sliding-window technique is implemented and used on trained classifier to search for vehicles in images.
- Pipeline is ran on a video which creates a heat map of recurring detections frame by frame to reject outliers and follow detected vehicles.
- Estimate a bounding box for vehicles detected.

Here I will consider the rubric points individually and describe how I addressed each point in my implementation.

Writeup / README

Histogram of Oriented Gradients (HOG)

1. Explain how (and identify where in your code) you extracted features from the training images.

Features were extracted in the second cell block. I decided to use spatial, histogram and HOG features. A lot of the code is from the Udacity course.

Function for getting spatial features bin_spatial is in cell 2 lines 4-8, histogram features function color_hist is in cell 2 lines 11-19 and hog features are extracted with a function get_hog_features in cell 2 lines 21-25. Function called extract features in cell 2 lines 27-52 put these three feature groups together.

Images are read in cell 1 lines 58-59. In cell 2 lines 62-70 features are extracted from the images. In cell 3 lines 7-9 the normalization is done. Here is an example of one of each of the `vehicle` and `non-vehicle` classes:

I then explored different color spaces and different `skimage.hog()` parameters (`orientations`, `pixels_per_cell`, and `cells_per_block`). After trying various parameters, the default ones gave quite good results which can be seen in cell 2 lines 55-60.

2. Explain how you settled on your final choice of parameters.

I tried various combinations of parameters and after trying various parameters, the default ones gave quite good results which can be seen in cell 2 lines 55-60, but only when features were combined together. If only HOG would have been used, then I would have chosen different parameters. The classifier got a result of 98.8% which seemed good enough for me to proceed. The color space was kept rgb as it good results when combined with spatial and histogram features.

3. Describe how (and identify where in your code) you trained a classifier using your selected features.

I trained a linear SVM using spatial, histogram and HOG features. Data was normalized in cell 3 lines 7-9. The data was randomized and split into training and test set in cell 3 lines 16-17. The training command itself is in cell 4 line 8. The accuracy is tested on the testset in cell 4 line 12.

Sliding Window Search

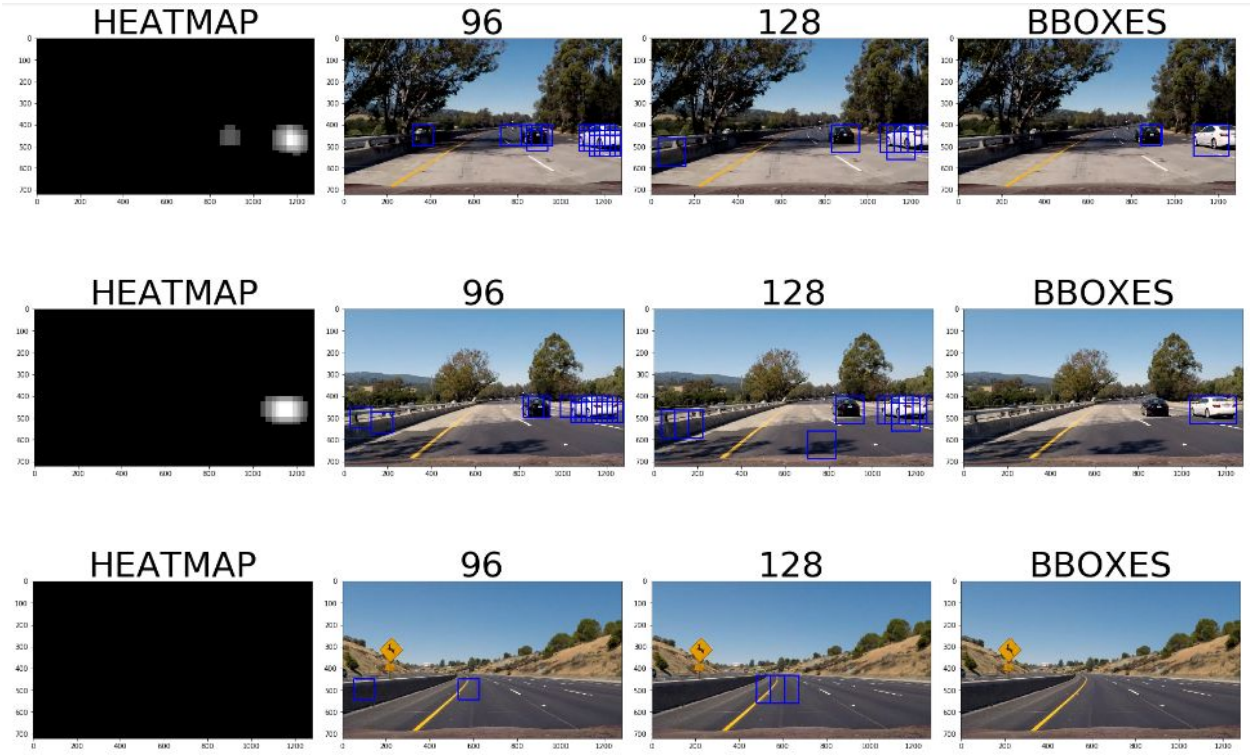
1. Describe how (and identify where in your code) you implemented a sliding window search. How did you decide what scales to search and how much to overlap windows?

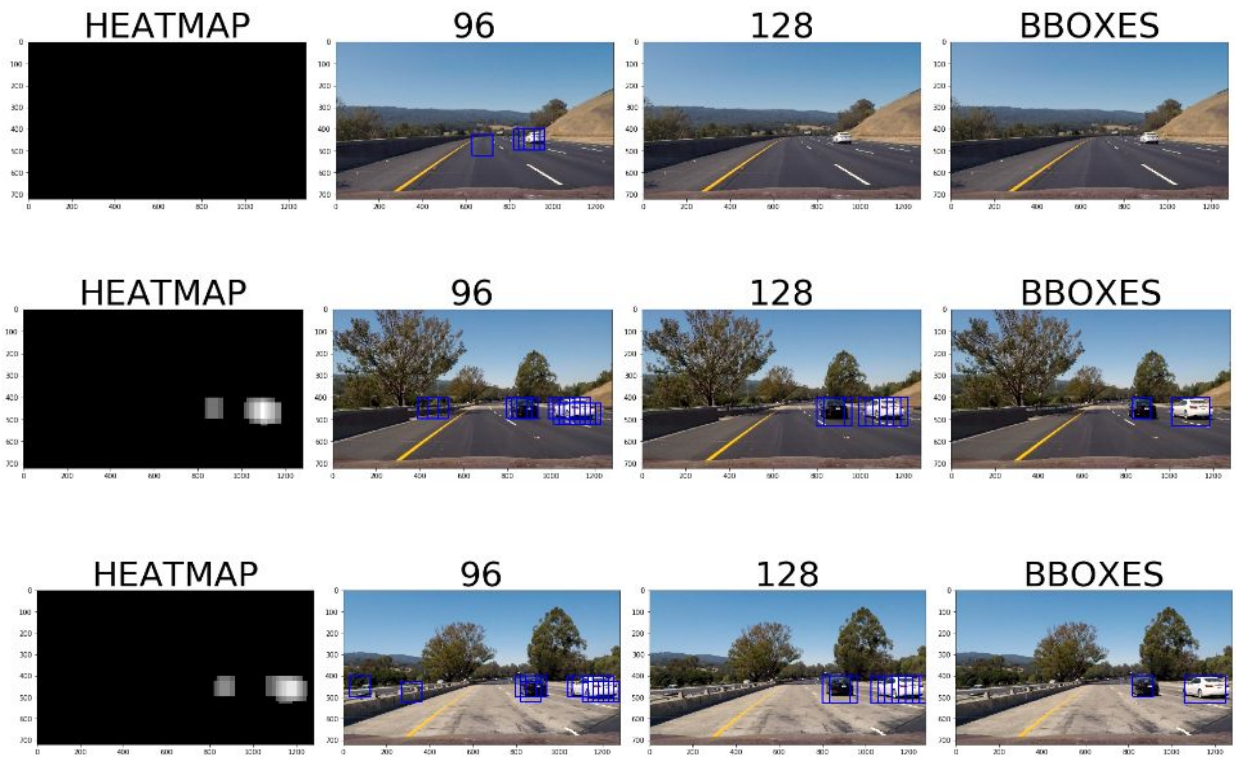
One part of sliding windows was done in cell 5 lines 7-46. This returns the windows from which to search for vehicles. I got good results when overlap was 75%, but this depends on a lot of variables and may be different with a different setup.

Since I wanted to speed up the process I decided to extract HOG features only once for the image. This is done in cell 5 function search_windows lines 101-139.

2. Show some examples of test images to demonstrate how your pipeline is working. What did you do to optimize the performance of your classifier?

Finally I searched on two scales - (96, 96) and (128, 128) sized windows with the setup described above. Smaller windows resulted in more false positives and bigger windows rarely detected anything, so I did not use them to improve the speed of the algorithm. The pipeline without centroid tracking can be seen in cell 6.Here are some example images:





Finally, I also created two classes - Centroid in cell 7 lines 64-117 and TrackImage in cell 7 lines 120-198. Centroid is used to keep track of the labels which are appearing thanks to heatmaps and TrackImage is used to keep track of heatmaps over multiple frames.

Video Implementation

1. Provide a link to your final video output. Your pipeline should perform reasonably well on the entire project video (somewhat wobbly or unstable bounding boxes are ok as long as you are identifying the vehicles most of the time with minimal false positives.)

Here's a link to video.

2. Describe how (and identify where in your code) you implemented some kind of filter for false positives and some method for combining overlapping bounding boxes.

I added the heatmaps over multiple frames and I kept track of the centroids of the labeled boxes. If the boxes appeared over multiple frames their confidence increased and they were displayed. If they were not present in the succeeding frames, their

confidence decreased so that they disappeared or were deleted completely. This approach helped to get rid of false positives.

Discussion

1. Briefly discuss any problems / issues you faced in your implementation of this project. Where will your pipeline likely fail? What could you do to make it more robust?

The classifier accuracy was 98.8%, but there were still a lot false positives which messed up the results quite a lot, meaning I had to build quite a lot of filtering into TrackImage and centroid. It will probably fail in different lighting conditions. The Classifier accuracy should be better which could probably be easily improved with more data and a more powerful classifier. I wanted to focus on the sliding-window, heatmap and tracking approach as I had used a good classifier for a similar task in the past already and that would not have been such a challenge.

I could have used the false positives received during testing on video to add them to my training data for the algorithm to become more robust. This method is called hard negative mining if I am not mistaken.