UDACITY

< Back to Self-Driving Car Engineer

Vehicle Detection and Tracking

REVIEW

CODE REVIEW

HISTORY

Requires Changes

1 SPECIFICATION REQUIRES CHANGES

Writeup / README

The writeup / README should include a statement and supporting figures / images that explain how each rubric item was addressed, and specifically where in the code each step was handled.

Good job addressing the rubric items in the 'Project 5 Term 1 Write up _ Evernote Web.pdf' file with supporting images.

Histogram of Oriented Gradients (HOG)

Explanation given for methods used to extract HOG features, including which color space was chosen, which HOG parameters (orientations, pixels_per_cell, cells_per_block), and why.

Good job choosing the 'YCrCb' color space. Good work computing the HOG values for all three color channels. Well done using a binned histogram of the three color channels and a downsampled (16x16) pixel image as the input features for the classifier.

Well done mentioning in the 'Project 5 Term 1 Write up _ Evernote Web.pdf' file, " At first I train SVC with the provided dataset and the Test Accuracy of SVC is not great. No matter how I twist the HOG parameters (change color space, change , the Test Accuracy of SVC is around 0.97 area. To improve the result, I augment the data set by flip the image and rotate them vertically to generate more data. The result is still the same (around 0.97)."

The HOG features extracted from the training data have been used to train a classifier, could be SVM, Decision Tree or other. Features should be scaled to zero mean and unit variance before training the classifier.

Nice job using StandardScaler() to normalize the dataset by features. Nice job splitting the dataset into a training and holdout testing set with train_test_split. Good work using a support vector machine classifier to train the data and test the accuracy of the model on the holdout test data.

Some students have found success trying a 'rbf' kernel instead of a linear kernel for the svm classifier.

Sliding Window Search

A sliding window approach has been implemented, where overlapping tiles in each test image are classified as vehicle or non-vehicle. Some justification has been given for the particular implementation chosen.

Good work using 3 different window sizes to try to detect the vehicles in the image. Good job limiting the sliding windows to only the area of the image where cars are most likely to occur for the image.

Some discussion is given around how you improved the reliability of the classifier i.e., fewer false positives and more reliable car detections (this could be things like choice of feature vector, thresholding the decision function, hard negative mining etc.)

Good work trying to use the svm `decision_function()` method to try to reduce the number of outliers for the sliding window classification.

Well done also mentioning in the 'Project 5 Term 1 Write up _ Evernote Web.pdf' file, "At first I train SVC with the provided dataset and the Test Accuracy of SVC is not great. No matter how I twist the HOG parameters (change color space, change , the Test Accuracy of SVC is around 0.97 area. To improve the result, I augment the data set by flip the image and rotate them vertically to generate more data. The result is still the same (around 0.97)."

Hard negative mining and using RandomSearchCV() or GridSearchCV() to try to tune the hyperparamters are some things that could be tried to improve the accuracy of the pipeline.

Hard negative mining is when we take the false positives from the output and insert it back into the training data and retrain the model. One way to do this is to limit the window search to the area(s) of the image where no expected vehicles are to occur. Then all of the classified vehicles would be false positives, and all of the detected sub images could be outputted and inserted into the training set to have the model retrained.
GridSearchCV()` will perform an exhaustive search over the specified parameter values given to try to find the best set of parameters that will give the highest test accuracy score for the dataset. RandomSearchCV() is a similar method that will search over multiple hyperparameter values but will randomly set the hyperparameter vales within a range, instead of using a fixed interval.

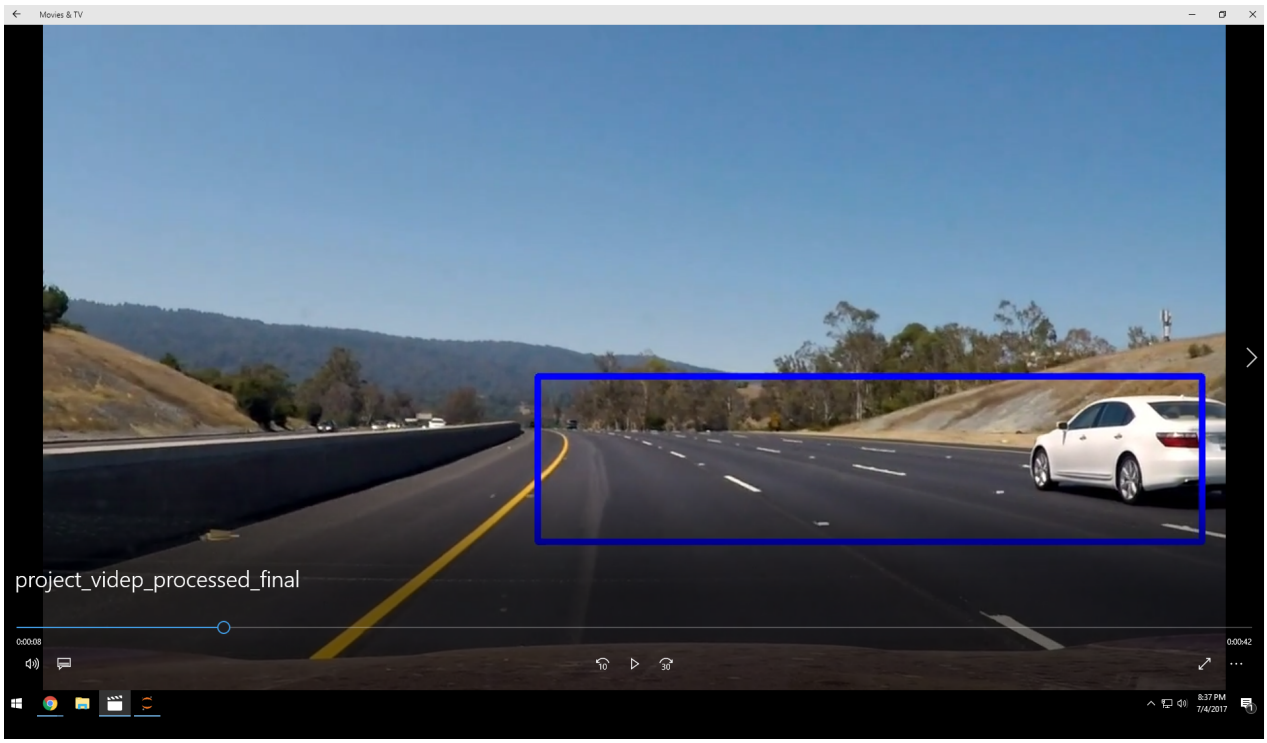
Video Implementation

The sliding-window search plus classifier has been used to search for and identify vehicles in the videos provided. Video output has been generated with detected vehicle positions drawn (bounding boxes, circles, cubes, etc.) on each frame of video.

Good work outputting the annotated project_video. There are still too many false positives in the output video as shown by one frame below.

The code looks fine and good so I'm not sure where the problem is for the large amounts of false detections. As mentioned below, one thing that could be tried is to decrease the 'n' number of frames used for the heat map from 30 frames to something like 5-10, for example, to see if that could help improve the accuracy of the classifier. The heat map threshold value could also be tuned as well to see if a more accurate output could be produced.

Another thing that could be tried as mentioned above is trying to use a `rbf` kernel for the svm classifier instead of a linear kernel to see if that could improve the test accuracy of the classifier. Some students have found success using the `rbf` kernel.



Please decrease the amount of false positive labels in the output video before re submission. Thank you.

A method, such as requiring that a detection be found at or near the same position in several subsequent frames, (could be a heat map showing the location of repeat detections) is implemented as a means of rejecting false positives, and this demonstrably reduces the number of false positives. Same or similar method used to draw bounding boxes (or circles, cubes, etc.) around high-confidence detections where multiple overlapping detections occur.

Good work incorporating a thresholded heatmap across multiple frames to try to reduce the number of false positives, increase the number of true positives and smooth the bounding boxes.

The output video still has too many false positives. We could try to reduce the number of consecutive frames used for the heat map to see if that could improve the output accuarcy of the output video. We could try to use the heat map consecutive frames number to between 5 - 10 to see if that can help improve the performance of the output video.

Discussion

Discussion includes some consideration of problems/issues faced, what could be improved about their algorithm/pipeline, and what hypothetical cases would cause their pipeline to fail.

Nice job mentioning in the 'Project 5 Term 1 Write up _ Evernote Web.pdf' file, "There are some false positive frames that seems to be appear consistently. Hence even after I apply heatmap, they still there. Hence the bounding box sometime cover a larger area that I wanted."

We should expand on this section. Please review the rubirc item above to see what should be written for this rubric item.

RESUBMIT

DOWNLOAD PROJECT

Best practices for your project resubmission

Ben shares 5 helpful tips to get you through revising and resubmitting your project.

Watch Video (3:01)

RETURN TO PATH