Vehicle Detection Using Machine Learning

Uma Kamatchi Kesava Pillai
Ingram school of Engineering
Texas state of university
uma.kamatchi@gmail.com

Abstract— **This project is aimed for accurate vehicle detection for the application of self-driving cars to avoid possible accidents. Machine learning has revolutionized computer vision, and it is the core technology behind capabilities for improving the task of object detection. Image features such as color and shape are extracted using Color Histogram, Spatial Binning and Histogram of oriented gradients methods. A linear support vector machine (SVM), Decision tree and random forest classifier is experimented with these features using scikit-learn library in python. Linear Support Vector Machine classifier shows the highest accuracy of 0. 9782 and minimum time taken to train the model. Tested model performance with standard machine learning performance metrices such as classification accuracy, precision, recall and F1 score. Implemented Sliding window technique for searching vehicles in an image and applied heatmaps to reduce false positive. The trained SVM classifier is used to predict vehicle in an image with bounding boxes.**

Keywords—component, formatting, style, styling, insert (*key words****)***

I. Introduction

According to the statistics of Texas motor vehicle traffic crash facts, the fatality rate on Texas roadways for 2017 was 1.37 death per hundred million vehicle miles travelled which was death toll of 3727 number of motor vehicle traffic fatalities [1]. More often these accidents happen due to lack of driver's attention and adverse weather. So, this can be overcome by alerting the driver while driving the vehicle. Accurate vehicle detection is the basic requirement.

Object detection methods have a wide range of application includes robotics, medical image analysis, surveillance and human computer interaction [2].Computer vision enables computer to recognize the pattern, identify and process object in images in the similar fashion as humans do. One way to train a computer is to understand visual data is to feed it lots of images that have been labelled and then subject those to algorithms or software techniques, that allow the computer to hunt down patterns in all the elements that relate to those labels[3].

Computer vision focuses on replicating parts of the complexity of the human vision system and enabling computers to identify and process objects in images and videos in the same way that humans do. Until recently, computer vision only worked in limited capacity. It is all about pattern recognition. So one way to train a computer to understand visual data is to feed it images, lots of images thousands, that will label, and then subject those to various software techniques, or algorithms, that allow the computer to hunt down patterns in all the elements that relate to those labels[4]. The three most common tasks in computer vision are image classification, classification with localization and object detection.

The goal is to determine whether there is a vehicle in the image but also where exactly the vehicle is? Object localization algorithms not only label the class of an object but also draw a bounding box around the position of an object in the image [5]. Machine learning provided a different approach to solving computer vision problems. With machine learning, developers no longer needed to manually code every single rule into their vision applications. Instead they programmed "features, "smaller applications that could detect specific patterns in images. They then used a statistical learning algorithm such as decision trees or support vector machines (SVM) to detect patterns and classify images and detect objects in them [4].

scikit-learn. is a Simple and efficient tool for data mining and data analysis, accessible to everybody, and reusable in various contexts, Built on NumPy, SciPy, and matplotlib and Open source, commercially usable

II. Background

A. *Feature Extraction*

In machine learning and image processing, feature extraction starts from an initial set of measured data and builds derived values (features) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps, and in some cases leading to better human interpretations. Feature extraction is related to reduction. When the input data to an algorithm is too large to be processed and it is suspected to be redundant (the repetitiveness of images presented as pixels), then it can be transformed into a reduced set of features as a feature vector. Determining a subset of the initial features is called feature selection. The selected features are expected to contain the relevant information from the input data, so that the desired task can be performed by using this reduced representation instead of the complete initial data [5].

The different methods to extract color features from the image are color histogram, Spatial Binning and Histogram of oriented gradients. Color histogram is a representation of the distribution of colors in an image. A color histogram focuses only on the proportion of the number of different types of colors, regardless of the spatial location of the colors. The values of a color histogram are from statistics. They show the statistical distribution of colors and the essential tone of an image [8].

a)Spatial binning

Binning is the process of combining a cluster of pixels into a single pixel which reduces the overall number of pixels.

b)Histogram of gradient

In the HOG feature descriptor, the histograms of directions of gradients are used as features. Gradients of an image are useful because the magnitude of gradients is large around edges and corners (regions of abrupt intensity changes) and knowing that edges and corners pack in a lot more information about object shape than flat regions [9].

B. Machine learning

Vehicle detection is a type of supervised learning method of machine learning where the categories are predefined such as vehicle and non-vehicle and is used to categorize new probabilistic observations into said categories. Training a machine learning involves five main steps as follows:

1.Selecting features and collecting training samples.

2.Choosing a performance metric

3.Choosing a classifier and optimization algorithm.

4. Evaluating the performance of the model.

5. Tuning the algorithm [8].

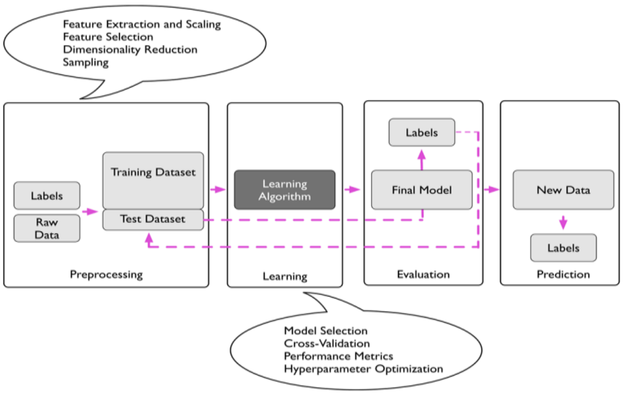


Fig 1: Workflow model

Fig:1 explains the typical workflow for machine learning in predicting vehicle detection model. It includes preprocessing-getting data into shape, Training and selecting a predictive model, evaluating models and predicting unseen data instances using python Scikit-learn library. The scikit-learn library offers not only a large variety of learning algorithms, but also many convenient functions to preprocess data and to fine-tune and evaluate models. Fig 2 shows the block diagram of vehicle detection.

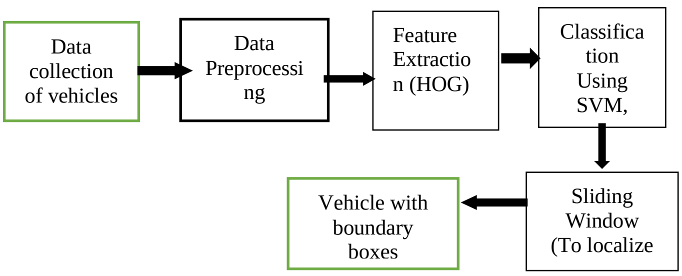


Fig 2 Block diagram of vehicle detection

C. Machine learning classifiers

Selected machine learning classifier according to the binary classification and supervised learning algorithms. Implement different machine learning algorithm to the vehicle detection model to determine the best classification accuracy using scikit-learn python library. The classifier which produces higher accuracy will help to Evaluate the model. The goal of classifier is to identify the object in the image is vehicle or not. The different type of classifier is SVM, Decision tree and Random forest.

a) Support vector Machine

Support Vector Machine is one of the most popular and widely used classifier in the world. The python community has created many SVM libraries that support all types of classification. It determines many parts of a classification includes Identifying the vectors, Drawing the hyperplanes, calculate the magnitude and margins etc. It gives best optimized boundary will help the model not to overfit [6].

b) Decision Tree

Decision tree is a type of supervised learning algorithm (having a pre-defined target variable) that is mostly used in classification problems. Decision tree classifier breaks down a decision of which class a new sample belongs to through questions and answers. It works for both categorical and continuous input and output variables. This model is developed by deducting the class labels of the samples. The splits are based on features with the largest information gain. Avoid overfitting problem by setting the hyperparameter as a maximum depth. Train a Decision trees classifier with different binary decision trees such as Gini, Entropy and classification error. set the hyperparameters as maximum depth to avoid overfitting problem [6].

c) Random Forest

Random forest is a collection of decision trees which are randomly trained by nature and easy to implement. The goal is to aggregate the accuracy of all the trees. Implementing the best classifier to Identify if an image contains a car

D. Experimentation setup

The algorithms were compared based on classification accuracy and time taken for training the model. Support Vector Machine classifier shows the highest training accuracy when compared to decision tree and random forest. The best performing model of will consider and implemented a sliding window technique and use the trained classifier to search for vehicles in images.

Vehicle detection model is implemented using scikit-learn machine learning python library in Spyder, python development environment.

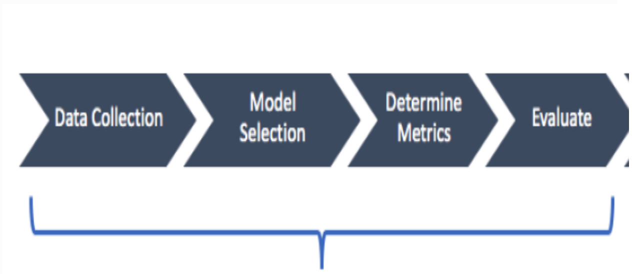


Fig 3: Experimental setup [9]

III. Procedure and Results

A. Data preprocessing

a) Datasets

The Vehicle Image database is collected from Grupo de Tratamiento de Imágenes (GTI).The dataset comprises 3900 images of vehicles (975 images x4 different regions) are in different poses and 3900 non-vehicle. The four different regions are selected according to the pose: middle/close range in front of the camera, middle/close range in the left, close/middle range in the right, and far range. In addition, the images are extracted in such a way that they do not perfectly fit the contour of the vehicle in order to make the classifier more robust to offsets in the hypothesis generation stage. Instead, some images contain the vehicle loosely (some background is also included in the image), while others only contain the vehicle partially. Several instances of a same vehicle are included with different bounding hypotheses. The images are cropped down to 64x64 pixels from sequences of 360x360 pixels recorded in highways [7]. Fig below shows the sample vehicle and Nonvehicle image from dataset.

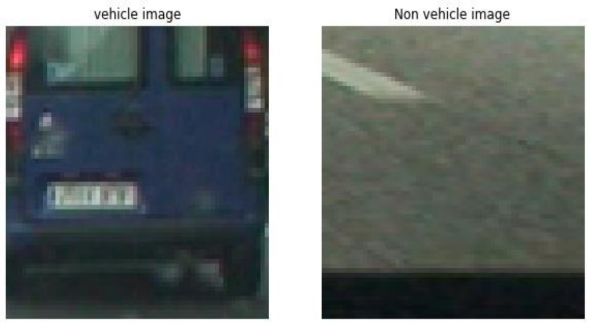


Fig 4:Vehicle and non vehicle image from Dataset

b). Extracting Color features

The following steps are followed for extracting color features from an image.

1.Color spaces extraction includes extract color histogram of three channels and the parameters are image, number of bins, range of bins (0,255).

2.Center of bin edges on histogram channel

3.Extracting color features from bin length includes concatenate all the three channels and give number of color features.

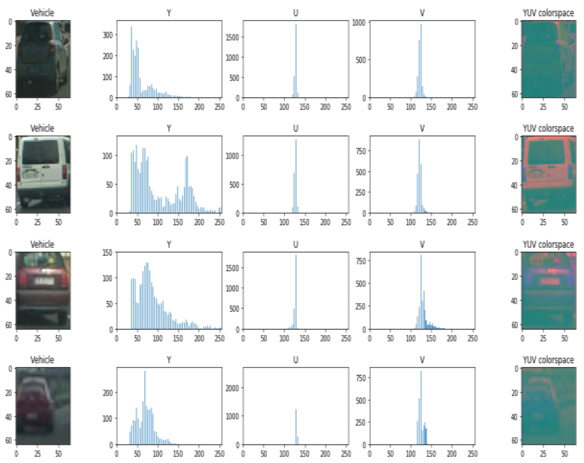


Fig 5: YUV color histogram from random 4vehicle images

c). Histogram of Gradient

The Histogram of Oriented Gradient (HOG) feature descriptor is popular for object detection and computed HOG using scikit-image image processing python library. The idea of HOG is to first group the pixel into small cells nxn pixels instead of using each individual pixel of an image then compute all the gradient directions and grouped into number of orientation bins and last sum up all the gradient magnitude in each image, so stronger gradients contribute more weight to their bins and effect of small random orientation is reduced.

As vehicles varies so much in color, measuring the gradients of specific direction captures some notion of shapes. The HOG features keep the representation of an object distinct but also shows variation in shape. The hyper tuning parameters for feature extraction are number of orientations, cells per block, pixels per block and color space.

The **number of orientations** is the number of orientation bins that the gradients of the pixel of each cell will be split up in the histogram. **pixels_per_cells** are the number of pixels of each row and column per cell over each gradient the histogram is computed. **cells_per_block** specifies the local area over which the histogram counts in a given cell will be normalized. Having this parameter is said to generally lead to a more robust feature set. The normalization across blocks called **transform_sqrt** which is to reduce the effects of shadows and illumination variations.

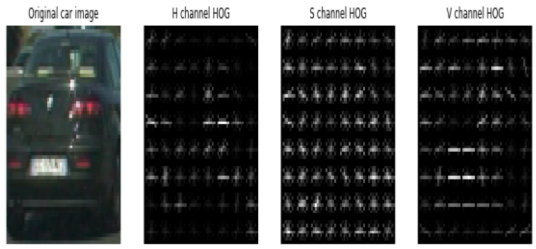


Fig 6: HSV channel Histogram of Gradient

a) Choosing HOG Parameters:

Explored different HOG hyper tuning parameters like number of orientations, pixel per cell, cells per block and color space to select the optimum parameters which help to extract the best HOG features from the image. This process involved fixing control features three of the control features to reasonable values and varying the fourth experimental feature and running the SVM classifier to see which one of the experiment options produced the best SVM classifier test accuracy.

b) Choosing pixel per cell with Test accuracy of SVC

First explored with three options for pixels per cell (pixels per cell=[(8, 8), (16, 16), (32, 32)]) and fixing the other HPG parameters such as color space to HSV ,orientations=9 and cells per block(2,2).

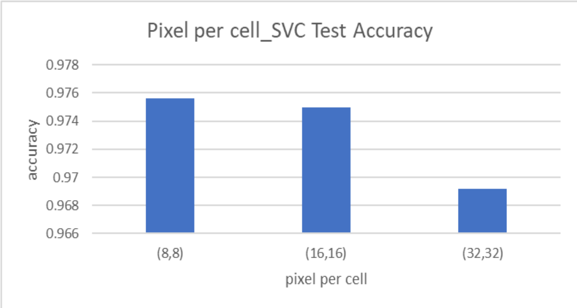


Fig 7:Pixel per cell SVC test accuracy

Fig7 shows Pixel (8,8) gives highest test accuracy of 0.9756.

c)Choosing color space

Second explored with six color spaces= [RGB, HSV, HLS, YCrCb, YUV, LUV]) and fixing the other HOG parametersto orientations=9, pixels_per_cell=(8,8) and cells _per_block=(2, 2).

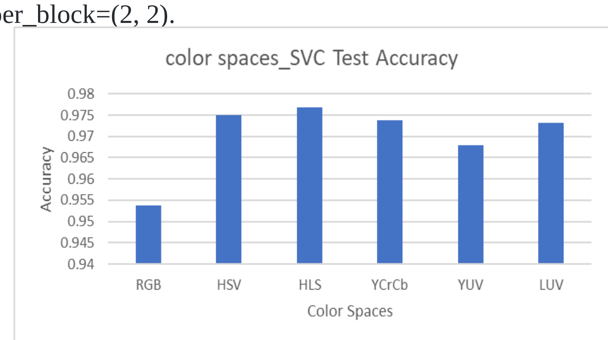


Fig 8: SVC test accuracy for different color space

Fig shows HLS color space gives highest test accuracy of 0.9769

1.2.3 Choosing Orientation with test accuracy of svc classifier

Third explored six orientations [7,8,9,10,11,12] and fixing the other HPG parameters such as color space to HSV, pixel per cell (8,8) and cells per block (2,2).

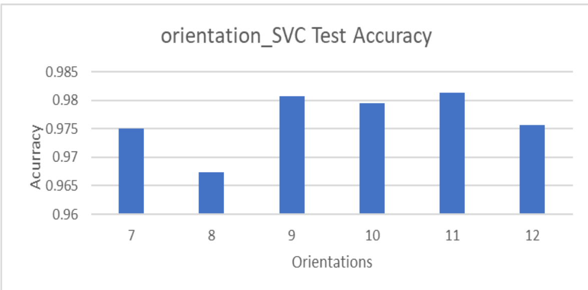


Fig 9 :SVC test accuracy for different orientations

B. *SVC classifier with spatial,color and HOG*

Added spatial binning and color histogram features to the HOG features to see if the test prediction accuracy improves further. With spatial binning the feature vector length obtained is 3072.The test accuracy improved to 98.78% with spatial binning and histogram of color transforms. My final parameter selection for the classifier is as follows: color_space = HSV,orientations=1,pixels_per_cell=(8, 8) cells_per_block=(2, 2),spatial bins = (32,32), histogram bins = 32

The classifier with the above parameters will be used to predict vehicle in the image.

C. *Classifier Training*

a). Normalizing Data

Normalizing the combined feature set is important to make sure it has a Gaussian distribution with zero mean and unit variance. Many machine learning algorithms do not perform well if the data is not centered at zero mean with unit variance. This is because the features with large variance dominate and have undue influence on the classifier. The function StandardScaler from sklearn package is used to scale the data. Used the same scaler obtained from the training data on the prediction data as well.

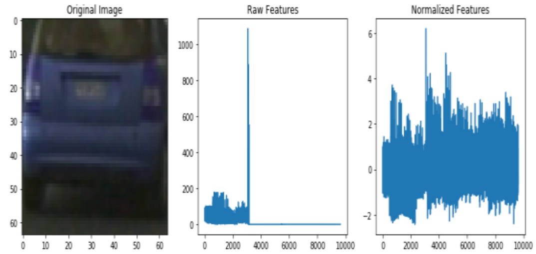


Fig 10 :Normalizing raw features from vehicle image

Fig 10 shows the visualization of raw features and normalized features with zero mean and unit variance for the random data.

a). Split Data

Data set is split in to 80% of the data set to train the classifier and the remaining 20% to determine the accuracy of the classifier. To randomize the splitting of the data used built-in function train_test_split from sklearn and fed it a random number between one and 100 (0,100).

c) *Support Vector Machine Classifier*:

Experimented with random 100 samples to determine training, test accuracy and time taken for each classifier to train data set.

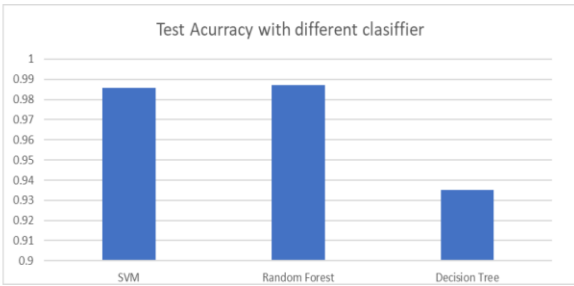


Fig 11: Testing accuracy with different classifier

Fig 11 shows the Random forest with best accuracy of 0.9872.

Fig 12. Time taken for each classifier to train the model for random samples

Fig 12 shows SVM classifier gives the minimum time taken to train the model.

c) *Performance Metrics*:

Evaluated Model performance with different techniques like confusion matrix, Precision, Recall and F1 score. The confusion matrix is the graphical A graphical matrix that show the number or percentage of the true-positive (TP) false-positive (FP),A false-negative (FN),A true-negative (TN).The obtained confusion matrix shows the vehicle and non-vehicle classification with true and predicted values.

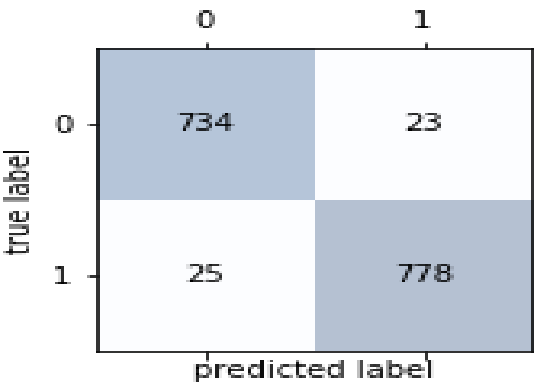


Fig 13: Confusion matrix for vehicle and nonvehicle classification using SVM classifier

Precision (positive predictive value is the fraction of relevant instances among the retrieved instances, while **recall** (also known as sensitivity) is the fraction of the total amount of relevant instances that were actually retrieved. Both precision and recall are therefore based on an understanding and measure of relevance[10].The obtained values are Precision: 0.985,Recall: 0.972,F1: 0.979.The accuracy on the test set is 0.9782.

D. *Sliding Window Techniique*

Implemented sliding window technique for searching vehicles in an image. To get a subregion of an image and run classifier in that region to see if the patch contains a vehicle. First extract HOG features of the whole frame of an image and then subsample to get all its overlapping windows. Each window is defined by a scaling factor of 1 with window (8,8). Increased the step size of 2 cells per step which results in 75% (6/8) overlap between windows. A 75% overlap is granular enough to capture vehicle in the entire image. Four scaling factors and the corresponding regions are shown as Scale 1.0 with ystart = 400 and ystop = 528,Scale 1.5 with ystart = 400 and ystop = 592,Scale 2.0 with ystart = 400 and ystop = 656,Scale 2.5 with ystart = 336 and ystop =656.This will start searching from about the middle (y=336) of the image in the y direction to the bottom of the image (y=656). The smallest scale (1.0) in the horizon of the image as the cars are small at that distance and increased progressively larger scales to find larger size cars which appear closer to the driver.

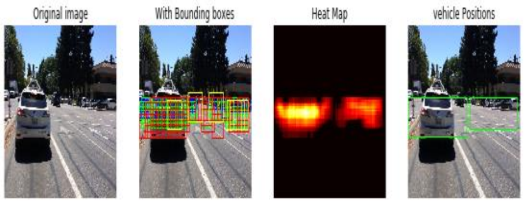
Fig 14: Tested image with Bounding boxes

Fig 14 shows the tested image with the Linear Support vector Machine classifier for the prediction of vehicle detection model.

E. *Heatmaps*

There are overlapping detections and false positive detections in fig. Heatmap is built to combine overlapping detections and remove false positives. Heat map is start with a blank grid and add heat (+1) for all pixels within windows where positive detections are reported by the classifier. The hotter the parts more likely to be a true positive and impose a threshold to reject areas affected by the false positives. Used scipy. ndimage.measurements.label() function to identify individual blobs in the heatmap. Each blob corresponds to a vehicle. Created bounding boxes to cover the area of each blob detected.

The total number of windows is 470..These windows can appear anywhere if you have a lower threshold in the final heatmap image. Eliminated false positives by just increasing the threshold. Here set the threshold to value 4 but again thresholding depends on several factors, the color space used, the SVM accuracy.



IV. Conclusion and Future Work

This paper an efficient implementation of different machine learning classifier on training model to determine best accuracy for prediction. Experimental results show the support vector machine gives the highest accuracy when compared to decision tree and random forest with minimum time taken to train a model. The model performances are evaluated by different metrices like confusion matrix, precision, Recall and F-1 score. The SVM classifier is implemented to predict an vehicle in an image on testing an image. Sliding window technique is used to search an vehicle and eliminated false positive by setting threshold value to 4. Finally, Vehicle is detected with bounding boxes.

In future vehicle detection model can be implemented using Convolutional Neural Networks and various deep learning architecture like VggNet, ResNet. and Inception with large image datasets.

References

[1] https://ftp.dot.state.tx.us/pub/txdotinfo/trf/crash_statistics/2017/01.pdf

[2] https://en.wikipedia.org/wiki/Object_detection

[3] https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e (computer vision)

[4] https://machinelearningmastery.com/what-is-computer-vision/

[5] https://en.wikipedia.org/wiki/Feature_extraction

[6] https://luozm.github.io/cv-tasks

[7] https://www.gti.ssr.upm.es/data/Vehicle_database.html

[8] https://scikit-learn.org/stable/modules/svm.html

[9] Simhambhatla, Ramesh; Okiah, Kevin; Kuchkula, Shravan; and Slater, Robert (2019) "Self-Driving Cars: Evaluation of Deep Learning

[10] https://en.wikipedia.org/wiki/Color_histogram

[11] https://www.learnopencv.com/histogram-of-oriented-gradients/

[12] https://en.wikipedia.org/wiki/Color_histogram.

[13] https://www.learnopencv.com/histogram-of-oriented-gradients/