ENPM673
Project 6
Perception for Autonomous Robots

Project Group -

Amrish Baskaran (116301046)
Arpit Maclay (116314992)
Bala Murali Manoghar Sai Sudhakar (116150712)

Traffic Recognition

Traffic sign recognition is a part of an advanced driver assistance system which can recognize road signs and pass on the corresponding information for further vehicle control.

The problem is broken into 2 parts Sign Detection and Sign Classification. In detection, we will detect the coordinates and size of a signboard in the image and in classification, we try to categorize what the signal represents or what class this signal belongs. In our case there are eight classes of signboards that need to to be classified are shown below:

Output generated from the below can be found in this drive link-
https://drive.google.com/drive/folders/1y5SPqamBfkV4t6dzHujdMO1OpZHfm1WZ?usp=sharing

(a) 45

(b) 21

(c) 38

(d) 35

(e) 17

(f) 1

(g) 14

(h) 19

Overall Pipeline Used

1.) Traffic Sign Detection-

a. *Denoise*

Continuous contours are interested candidates. But when the raw image is processed for extracting contours they will be broken into small pieces because of the noise in the image. To average the noise and get a uniform intensity space, such that required region is detected as a single contour, denoising is performed. Since MSER takes into account of continuous intensity region, we do ‘fastNlMeansDenoisingColored’.

b. *Normalization on each channel*

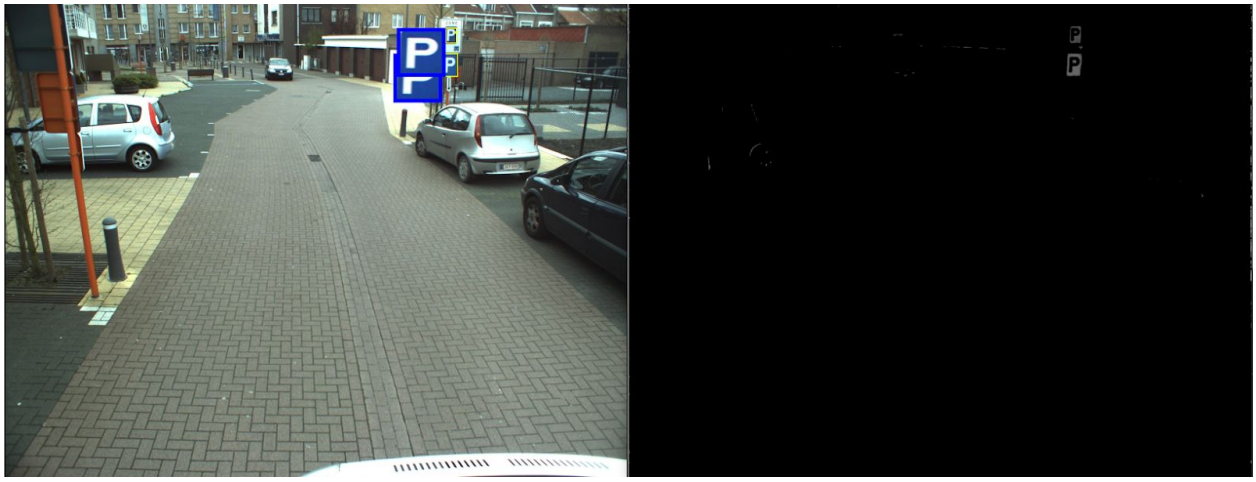
For Red Channel

$$C' = \max(0, \frac{\min(R-B, R-G)}{R+G+B})$$

For Blue Channel

$$C' = \max(0, \frac{B-R}{R+G+B})$$

Normalization is done to enhance each color in the image so that thresholding becomes easy. Above formula is applied to identify the blue and red region specifically. Blue and red colors are particularly taken since the set of traffic signs that need to be detected have these colors that mark the outer boundary. Even though normalization enhances the region of interest, due to lighting variation the colors are disturbed. To overcome the lighting effects to some extent, the image is adjusted similar to equalization is used. We used an adjust function (gamma adjustment) similar to Matlab adjust to is used for this purpose



Blue color enhancement and normalization output

red color enhancement and normalization output

c. *MSER-*

A Maximally Stable Extremal Regions is obtained by using MSER. This gives us contours that cover the area with the same properties.
This is a better choice than FindContour() as provides contours around areas that may be inside another similar region which FindContours may not detect. Some traffic signs like parking have this property. Or the background may be blue or red.
MSER also provides us with various options for rejecting contours. The parameters that were chosen-
delta = 3
maxVar = 0.5
minDiv = 0.2
minArea = 300
maxArea = 4500
Explanation to these parameters can be found here-
https://docs.opencv.org/3.4/d3/d28/classcv_1_1MSER.html

Example output of MSER

2.) Traffic Sign Classification

a. *HOG*

Image has to be broken down to descriptors so that classifier can use these features to classify the traffic sign type. HOG i.e histogram of gradient is a better descriptor which is robust to intensity variation. HOG of the extracted region is computed for further processing.

b. *SVM*

Support Vector Machine is a machine learning model which is a subset of supervised learning technique. It is a classifier and tries to fit a hyperplane as a decision boundary between multiple classes. The model is essentially these boundaries.

c. *Prediction*
   Once HOG descriptor is computed for the region to be checked and a trained classifier model is ready, this descriptor that describes the features of the identified region (potential traffic sign candidate) is sent to model for prediction. Based on the prediction, the region is classified.
d. *Annotation*
   Once the region is classified the region is annotated based on the confidence level.

Pipeline for robust region extraction

1. Approximately the top 1/3rd of the input image is taken/ cropped out. As the traffic signs do not go below this before they pass by the right or left of the viewed image.
2. The contours are extracted using MSER.
3. To remove the false positives using aspect ratio, ellipses are fit and the major and minor axes are obtained. By dividing and checking if they lie in [0.8, 1.2] we are able to reject many contours.
4. This allows a few trapezoids to pass through. This rejected by fitting a rectangle and dividing the width by height and checking if it lies between [0.8, 1.2]. Finally giving us the required contours with minimum false positives.
5. A lot of Contour for the same region is seen. These are combined by
    a. Creating a mask of the size of the image
    b. Drawing filled contours for all the above into the image.
    c. And finally, a contour is fit over this mask to obtain a single contour that covers the required sign.
6. Also to reject inner contours, the inside of the contour is checked intensity in the greyscale normalized image. If it less than 0.2% of the area then it is rejected.

Pipeline for Training and Testing Classifier

1. Training Phase
    a. Collect all the labeled data from the training set.
    b. Compute HOG features for all these data
    c. Send the hog descriptor and corresponding label to SVM model and train the model
    d. Check the fitting accuracy by back predicting on the training set
    e. Save the trained model for future prediction
2. Testing Phase
    a. Collect all the labeled data from the training set.
    b. Compute HOG features for all these data
    c. Predict the for each dataset using the above-trained model

    d. Compare the prediction results with the ground truth data and analyze the accuracy of the model
3. Tune the hyperparameters based on results obtained

Turning Hyperparameters for SVM

The parameters C, Gamma, and the kernel function affect the accuracy of the model.

C	Gamma	Kernal	Training data Accuracy	Test data accuracy
12	auto	linear	72.5%	55.3%
2	auto	RBF	75.3%	62.5%
7	auto	RBF	80.7%	73.8%
12.5	auto	RBF	99.2%	92.3%
15	auto	RBF	99.3%	93.2%

From the above table, it can be seen that the radial basis function kernel gives better results compared to the linear kernel. This is due to the face that when the linear mapping is done using RBF, the separation between the clusters can be higher when compared to the linear counterpart. Also for value C = 15, the test data accuracy is slightly better but the train data accuracy did not show a step in improvement when compared to C = 12.5. So C = 12.5 is chosen over 15 since is might overfit the train data and future predictions can be worse. Also “one vs rest” and “one vs one” models are compared. the one vs one model gave the best results.

OpenCV vs Scikit Learn

Initially, SVM from cv2 was used to train the model, option of auto for gamma parameter was not available with cv2 SVM. Sklearn’s SVM had better flexibility in choosing parameters. One major drawback was that the model only gave a class prediction as output but the probability was not given as output. The probability helps us to build more intuition on the confidence of the prediction.

HOG parameters

one vs one classifier
winSize = (128,128)
cellSize = (8,8)
blockSize = (16,16)
blockStride = (8,8)
nbins = 9
derivAperture = 1
winSigma = -1.
histogramNormType = 0
L2HysThreshold = 0.2
gammaCorrection = 1
nlevels = 64
signedGradient = False

Since the image size is 128 x 128 the hog descriptor for each image is very big. When the entire data set is loaded and HOG was calculated, the kernel crashed due to memory overflow. So batch training approach was taken. A batch of 16 images are taken together, HOG descriptors are computed and the model is trained. For the next batch, the already trained model is used and retrained using the new set of data. This improved both speed and memory usage.

Pipeline for high confidence classification:

For every possible marked traffic sign cropped out and sent for prediction by the SVM, we are provided with a set of probabilities. Using this we can conclude how close is the prediction. If any of the values provided is greater than 0.6 (chosen based on experiments and the best value is chosen) then the prediction is good enough and the corresponding sign of the prediction is shown beside the marked part in color. If not the prediction is displayed in Grayscale beside the marked part.

Colored annotation- High confidence
Grey annotation- Low confidence

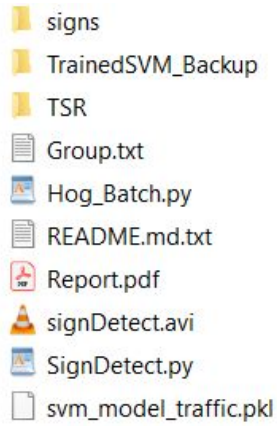
Colored annotation- High confidence
Grey annotation- Low confidence
The white arrow turns into colored annotation as the confidence level increases as shown in the next image

Colored annotation- High confidence
Grey annotation- Low confidenc

Other Addition that can be done

1. Homography- When the aspect ratio is used for rejection of contours, it is assumed that homography is performed and the image of the sign that is obtained is parallel to the camera. But this isn't so, As the object gets closer and to the right/ left of the image it shrinks in the direction the camera is facing. hence its length is perceived to be small, this results in false detection through the aspect ratio criteria.
2. Using shape as a factor and training an SVM and cascading with SVM trained for further classification.
3. Having separate SVMs for red and blue colored signs. Giving more options through which false positives can be neglected

Folder Structure



References

1. Drive Link to videos and files for this project
   https://drive.google.com/drive/folders/1y5SPqamBfkV4t6dzHujdMO1OpZHfm1WZ?usp=sharing
2. Imadjust for python
   https://stackoverflow.com/questions/39767612/what-is-the-equivalent-of-matlabs-imadjust-in-python/44529776#44529776
3. Contour function
   https://docs.opencv.org/3.4/dd/d49/tutorial_py_contour_features.html
4. MSER
   https://docs.opencv.org/3.4/d3/d28/classcv_1_1MSER.html
   https://github.com/opencv/opencv/blob/master/samples/python/mser.py
5. SVM- OpenCV tutorials
   https://www.learnopencv.com/handwritten-digits-classification-an-opencv-c-python-tutorial/