Enhanced Road Sign Detection and Recognition Using Color Segmentation and Support Vector Machine

Chenyang Shi (cshi67)

(Link to video presentation https://www.youtube.com/watch?v=Otdv0wfI9dg)

(Link to data https://drive.google.com/file/d/1IdX-MxneD_nYNWQZvyM7SPpiawkx3lbC/view)

Abstract

In this report, a pipeline has been built to detect and recognize seven road signs in real-world images (i.e. stop, added lane, signal ahead, merge, speed limit, pedestrian crossing and keep right). The detection part utilizes color segmentation and contour-finding algorithms to locate the region of the interest (ROI) while the recognition part trains on 1214 images from LISA database [1] of these seven traffic signs using a support vector machine (SVM) algorithm. An evaluation of current implementation is discussed with suggestions for further improvements.

Introduction

Road signs detection and recognition have attracted tremendous attentions in recent years as they are central to the development and the application of intelligent transportation systems. In a real-world application, the traffic sign recognition system takes in the environment images through the installation of the high-speed camera. These images are processed in real-time and the relevant traffic information are extracted and sent back for further decision-making. However, it is challenging to build such a recognition system as the real-world images are often recorded in unpredicted illuminations, at different camera angles and with different image sizes. Thus, finding a robust way to detect and recognize traffic signs has become the focal point of recent research.

In the literature, various approaches have been taken to attack this problem. For example, several research groups [2-4] have reported progresses based on color segmentation and/or shape features. In their work, they processed the images in HSV color space with predefined threshold values for each color. In conjunction with shape features, they were able to isolate the traffic sign of interest before feeding it to a machine learning classifier (e.g. a support vector machine or a neural network) for recognition. Other group [5] reported an improved method by combining detection and classification in a single step. They trained a convolution neural network on a 100000-image dataset (Tsinghua-Tencent 100 K) and reported a superior result.

In this report, a canonical approach similar to authors in references 2-4 is adopted based on color segmentation and contour finding.

Implementation

A flow chart shown in Figure 1 best summarizes my approach on the project.

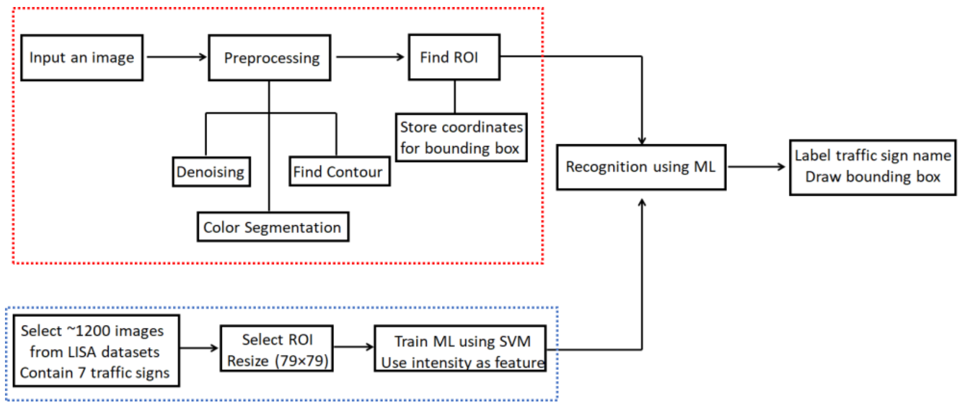
Input an image
Preprocessing
Find ROI
Store coordinates for bounding box
Denoising
Find Contour
Color Segmentation
Recognition using ML
Label traffic sign name Draw bounding box
Select ~1200 images from LISA datasets Contain 7 traffic signs
Select ROI Resize (79×79)
Train ML using SVM Use intensity as feature

Figure 1: A flow diagram demonstrates the algorithms implemented in this project.

The whole process can be divided into two subprocesses, i.e. the detection and recognition of traffic signs. The detection step, as highlighted in the red dotted box, starts from an input image to a preprocessing step that includes denoising (using Gaussian Blur with a window size of (7,7)), color segmentation and contour finding. Color segmentation is a critical step to efficiently filter out unwanted background objects. For stop, added lane, signal ahead, merge and pedestrian crossing signs, the color segmentation is done in HSV (i.e. Hue, Saturation and Value) color space while for speed limit and keep right signs, it is done in HLS (Hue Lightness and Saturation) color space where white color is best separated out by tuning Lightness. The threshold values for each sign is summarized in Table 1.

Table 1: The threshold values for isolating traffic signs using color segmentation.

Traffic Sign	Color Space	Lower Bound	Higher Bound
Stop	HSV	[156, 43, 46], [0, 43, 46]	[180, 255, 255], [10, 255, 255]
Added lane	HSV	[18, 148, 66]	[45, 255, 255]
Signal ahead	HSV	[18, 148, 66]	[45, 255, 255]
Merge	HSV	[18, 148, 66]	[45, 255, 255]
Pedestrian crossing	HSV	[18, 148, 66]	[45, 255, 255]
Speed limit	HLS	[0, 200, 0]	[255, 255, 255]
Keep right	HLS	[0, 190, 0]	[255, 255, 255]

Although using color thresholding can locate the traffic signs, it is not sufficient to isolate them from the background noise/objects. To improve it, a Gaussian Blur is applied to the input image (using `cv2.GaussianBlur(image, (7,7), 0)`). Additionally, an object shape detection algorithm is performed to further isolate desired traffic signs from background objects. This is realized using `cv2.findContours`. After these steps, a ROI region is found whose upper left and bottom right coordinates are stored for drawing a bounding box in the final step.

In the recognition step (as depicted in blue dotted box in Figure 1), a machine learning algorithm, i.e. support vector machine is trained on ~1200 images from LISA database on seven traffic signs of interest. The training samples are the ROI images extracted from the original dataset as shown in Figure 2. The traffic signs are grayscale images taken at various angles and lighting conditions with a large distribution of sizes.

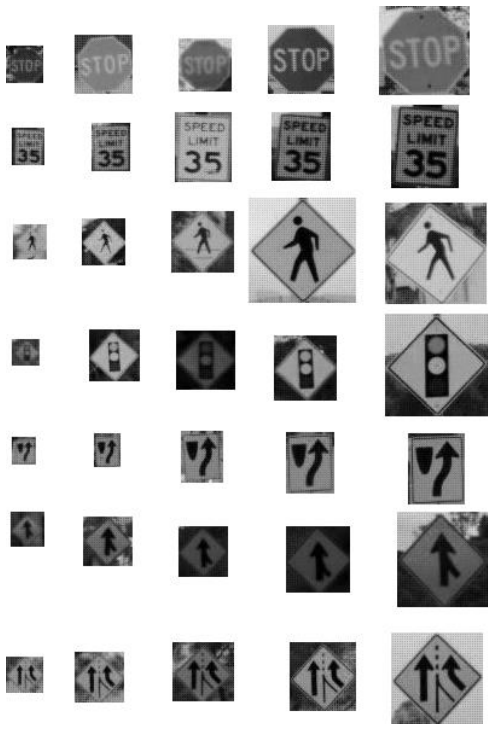

Figure 2: The ROI images extracted from LISA dataset that are used for machine learning.

The number of images used for training are summarized in Table 2. In LISA dataset there are 1821 stop and 1085 pedestrian crossing samples. To balance with other categories with fewer instances, 200 samples are randomly selected for stop and pedestrian crossing (using `random.sample(stop_signs, 200)`). Also, in original LISA dataset, one image could contain two or more traffic signs. These images are excluded for the sake of classification accuracy.

Table 2: A distribution of traffic signs used for machine learning.

Traffic Sign	Number of images	Percentages
Stop	200	16.47%
Added lane	168	13.84%
Signal ahead	203	16.72%
Merge	138	11.37%
Pedestrian crossing	200	16.47%
Speed limit	170	14.00%
Keep right	135	11.12%
Total	1214	100.00%

The 1214 ROI images are resized to be of the same mean size (i.e. 79 × 79 pixels). The images are flattened to a long vector of a size 6241, which is scaled using `StandardScaler()` from Scikit-Learn [6] before feeding into a Support Vector Machine classifier (`clf = svm.SVC(kernel='linear', C=1)`). A ten-fold cross-validation is implemented. The final SVM model together with the scaler are saved using `pickle.dump(clf, open(modelname, 'wb'))` which will be called later.

The overall detection and recognition processes is illustrated using the stop sign as an example in Figure 3. The input stop sign is blurred and then color segmented using the red color. The edges are detected before sending the cropped ROI to a SVM classifier. The final image with a bounding box and correct label is generated.

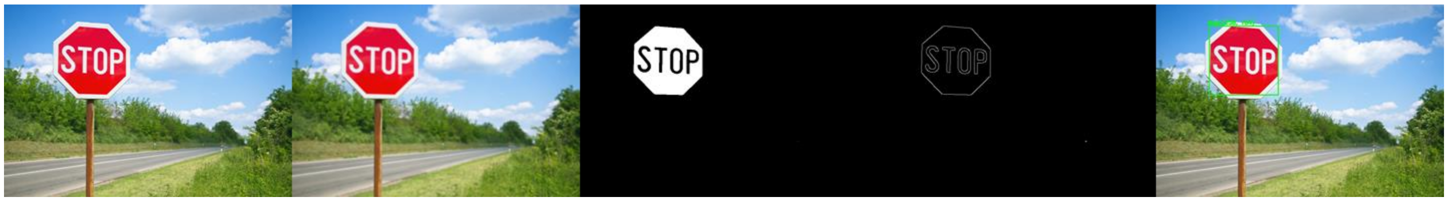

Figure 3: The pipeline to detect and recognize a stop sign.

Results

The classification of ~1200 ROI images extracted from LISA dataset is performed using an SVM classifier. After ten-fold cross-validation, a classification accuracy of 99.67±0.54% is achieved. Other classifier achieves a similar performance, for example, using a Random Forest Classifier, it returns an accuracy of 98.93±0.90%. In the end an SVM classifier is adopted, trained and saved for a later prediction.

The detection and recognition results on real images [7] are shown in Figure 4. The results shown here are those with true positives (while the failed cases are shown in Discussion section). Other than the traffic signs in interest, the real images contain other objects such as cars, roads, houses, snows, trees and skies, all of which may complicate the detection task. However, in these successful cases, the colors from the traffic signs show a strong contrast from those in the background, thus a color threshold method works. Using the same threshold values for yellow signs, one correctly detects added lane, merge, pedestrian crossing, signal ahead signs. The stop sign detection is relatively successful as well if background contains no red component, which is usually the case. The isolation of white color for speed limit and keep right signs is challenging, as the background, for example, the sky, always contains the white color component. The detection tends to break unless the right images are carefully chosen that suppress the white in the background.

Once the ROIs are correctly found. The classification step is straightforward and yields the reliable result.

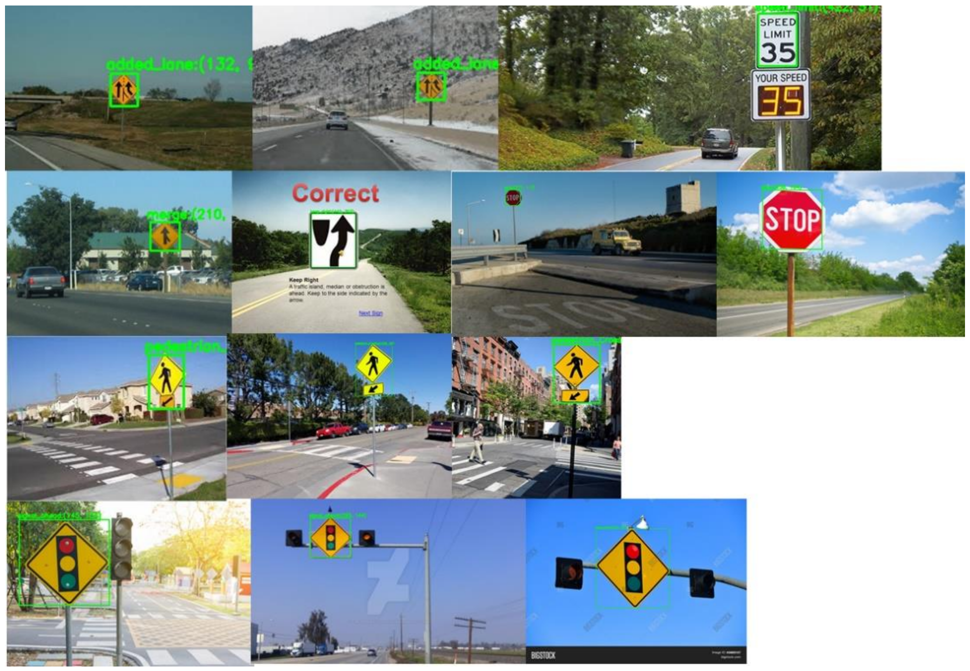

Figure 4: The traffic signs detection and recognition on real images.

Discussion

The failed detection/recognition cases are summarized in Figure 5. In the top row, the detection algorithm fails to identify the ROI. The algorithm breaks because the background has the similar color as the traffic signs of interest. The yellow added lane shares the same color as those from the traffic lights; the signal ahead sign has the same dominant yellow color as the sign behind it; the speed limit 65 sign has the same white color as the white truck passing by. In the bottom two rows, the recognition algorithm fails despite feeding it with a correct ROI. This may seem a bit surprising as the classification accuracy after ten-fold cross-validation is ~99.67%. The reason for the failure in detection may come from a small dataset of ~1200 images, which doesn't encompass all different scenarios of these signs. For example, both the stop and speed limit that are misclassified have a significant rotation, which are both misclassified as signal ahead. The pedestrian crossing sign in the middle of Figure 5 is misclassified as stop which may due to the reddish building in the background or the unwanted 'next 600 FT' included in the cropped ROI. The keep right sign is mislabeled because the ROI is not correctly chosen. The merge sign is not correctly recognized as it has great resemblance with the pedestrian crossing sign—both have the black symbol in the center surrounded by yellow color, forming an overall diamond shape.

Other possible reason for the mislabeling may arise from the training images are grayscale images which discard the fine details of the color. Another potential suggestion for improvement is do a better feature engineering using Histograms of Oriented Gradients (HOG) descriptors as opposed to use raw intensities as features (using `cv2.HOGDescriptor()`).

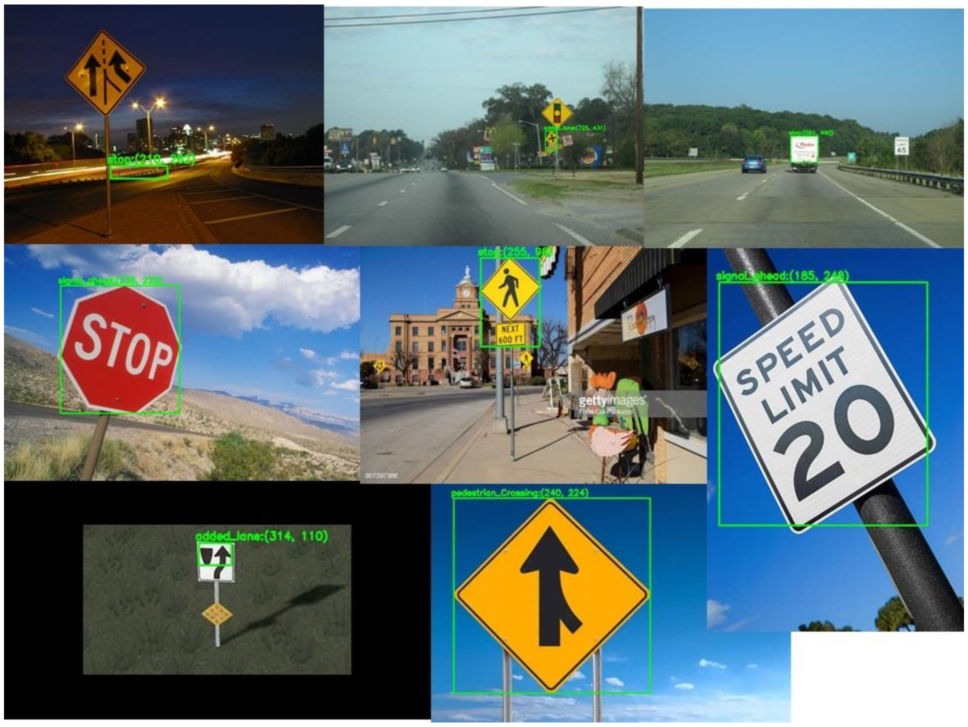

Figure 5: A collage of traffic sign images where the current algorithm breaks.

Here are several thoughts on future improvements.

- One may consider training a Cascade Classifier where remarkable success has been made in face detection using a Haar Cascade. It is a machine learning based approach where a cascade function is trained from many positive and negative images, which is later used to detect objects in other images [8].
- One may do a better job in color segmentation. Instead of using a single- color threshold, different threshold values should be developed for different weather, light or other real-world conditions.
- Large real-life images in the order of hundreds of thousands that covers almost all-aspect of real-world traffic signs should be used to train a reliable classifier. In our case, only ~1200 images are used which is far from sufficient for real-world object detection.
- Instead of training a machine learning model for recognition, one may use feature matching with homography to find objects. OpenCV has built-in functions for implementing this, for example, `cv2.findHomography, cv2.perspectiveTransform() and cv2.SIFT().` By the way, this feature matching using a template has been implemented by me but since it requires installing `opencv-contrib-python` which is not recommended by course TAs, thus not included in the report.

Conclusion

In this report, a traffic sign detection and recognition pipeline has been built and tested on real-world images. While the algorithm works correctly on certain images, it fails at more complicated situations. Further works need to be done to make it robust to unfavorable traffic/weather conditions.

References

[1] Andreas Møgelmose, Mohan M. Trivedi, and Thomas B. Moeslund, "Vision based Traffic Sign Detection and Analysis for Intelligent Driver Assistance Systems: Perspectives and Survey," IEEE Transactions on Intelligent Transportation Systems, 2012. http://cvrr.ucsd.edu/LISA/lisa-traffic-sign-dataset.html.
[2] Yixin Chen, Yi Xie and Yulin Wang, ‘Detection and Recognition of Traffic Signs Based on HSV Vision Model and Shape Features’, *Journal of Computers*, 8, 1366-1370, 2013.
[3] H. Rashimi, M. Shashidhar, G. Prashanth Kumar, ‘Automatic Tracking of Traffic Signs Based on HSV’, *International Journal of Engineering Research and Technology*, 3, 2014.

[4] Hasan Fleyeh, ‘Color Detection and Segmentation for Road and Traffic Signs’, Proceedings of the 2004 IEEE Conference on Cybernetics and Intelligent Systems, Singapore.
[5] Zhe Zhu, Dun Liang, Songhai Zhang, Xiaolei Huang, Baoli Li and Shimin Hu, ‘Traffic Sign Detection and Classification in the Wild’, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
[6] Pedregosa, F. et al. (2011). *J. Mach. Learn. Res.* 12, 2825–2830.
[7] The images used in this report are taken from Google Images. Their respective link is provided as below.

http://www.ideachampions.com/weblogs/archives/2012/11/a_sign_of_the_t.shtml (stop 1)
https://thenewswheel.com/are-stop-signs-with-white-borders-optional/ (stop 2)
http://www.ilankelman.org/stopsigns.html (stop 3)
https://www.radarsign.com/why-radar-speed-signs-work/(speed limit 1)
https://www.pinterest.com/salagraphicsinc/speed-limit-signs/ (speed limit 2)
https://www.istockphoto.com/photos/speed-limit-sign?sort=mostpopular&mediatype=photography&phrase=speed%20limit%20sign (speed limit 3)
https://www.picswe.com/pics/480-exits-b8.html (speed limit 4)
http://www.route56.com/highways/highways.php?hwy=10&seg=1&photos=2 (added lane 1)
https://www.needpix.com/photo/296324/lane-added-highway-road-lane-sign-street-information-blank-direction (added lane 2)
https://quizlet.com/17715112/drivers-ed-and-warning-signs-with-pictures-flash-cards/(added lane 3)
https://www1.nyc.gov/html/dot/html/pedestrians/enhanced-crossings.shtml (ped crossing 1)
http://www.pedbikeinfo.org/cms/downloads/PBIC_WhitePaper_Crosswalks.pdf (ped crossing 2)
https://www.cityofsanrafael.org/uncontrolled-crosswalks-evaluation-and-prioritization/(ped crossing 3)
https://slideplayer.com/slide/6189513/ (keep right 1)
https://steamcommunity.com/sharedfiles/filedetails/?id=880767734 (keep right 2)
https://www.edmunds.com/driving-tips/car-merging-psychology-dont-hate-the-sidezoomer.html (merge 1)
https://driversed.com/driving-information/city-rural-and-freeway-driving/merging.aspx (merge 2)
https://www.freepik.com/premium-photo/signal-ahead-traffic_3160157.htm (signal ahead 1)
https://www.deviantart.com/the-freeway-railfan/art/Signal-Ahead-668635216 (signal ahead 2)
https://www.crystalgraphicsimages.com/view/cg4p0988107c/traffic-signal-ahead-image(signal ahead 3)
http://deko4halloweenguenstig.info/signal-ahead-sign-image-87e62/ (signal ahead 4)

[8] https://docs.opencv.org/3.4.2/d7/d8b/tutorial_py_face_detection.html