Advanced Driver Assistant System

Zihui Liu zihui6@stanford.edu
Chen Zhu chen0908@stanford.edu
Department of Electrical Engineering

Abstract

In this project, we designed an advanced driver assistant system with three functions: traffic sign recognition, lane deviation detection, and car make identification. The traffic sign recognition implements Viola and Jones detector to accurately detect the traffic sign ahead. Hough transforms and edge detection are used in the lane deviation detection. For vehicle make identification, two models using 2-layer neural network and convolutional neural network have been constructed. These two models will provide high accuracy testing result after training processes.

Keywords: Object Detection, Deep Learning, Edge Detection, Hough Transform

1. Introduction

Advanced driver assistant systems (ADAS) have been implemented in many vehicles to help increase both the safety of drivers and pedestrian. The related technology is also used to develop self-driving cars. Three features including traffic sign recognition, lane deviation detection and car make identification in ADAS that our team built. Traffic sign recognition is crucial to remind the drivers the traffic signs ahead in order to prevent accidents caused by the traffic sign ignorance in the bad weather condition. It can be seen as another eyes to guarantee driving safety on the road. Lane deviation detection system provides lane detection and stability determination, giving drivers warning in the condition that car drifting into directions out of lane. Car make identification is a useful feature when drivers are interested in the make and model of car in their front. We will review these three features in details in following sections.

2. Related Work

2.1 Traffic Sign Recognition

D.M. Gavrila [1] implemented template-based correlation method to identify potential traffic signs in images called distance transforms as detection method. Then radial basis function network is used for classification. Miguel A Garcia-Garrido et al. [2] detected the shape of traffic signs using Hough Transform and used several SVM classifier with Gaussian kernel and probability estimate output to determine about the information on the traffics signs.

2.2 Lane Deviation Detection

Jia He et al. [3] used the Canny detector to find the image gradient and then set threshold for edge detection. Then they implemented the Hough transform as linear model fitting and then set the region of interest to achieve lane detection. Anik Saha et al. [4] implemented steps including selecting RGB image, converting RGB to grayscale, extracting feature, subtracting unwanted recognizing road and road lane for real-time automated road lane detection. Prof. Sachin Sharma et al. [5] used the algorithms following steps of Top-Hat Transform, Dynamic Threshold, ROI Segmentation, Hough Transform, Lane Departure decision.

2.3 Car Make Identification

Sparta Cheng et al. [6] implemented steps including interest point detection using Scale Invariant Feature Transform (SIFT) and Harris corner detection, interest point matching using Fast Normal Cross Correlation, and inlier extraction using RANSAC.

3. Algorithm and Implementation

3.1 Lane Detection

Lane deviation detection includes the functions of lane detection and stability determination. The algorithms contain two main parts, the edge detection and Hough Transform & Hough peak detection. For each frame in the video, first set the area in the image for further image processing. Our team typically selected the lower area of each picture frame in order to get less noise from the background part of image. The edge detection algorithm then

applies to the region selected. Using the Hough transform, the angles of lanes can be found in each frame and lanes can be successfully detected. For stability determination feature, we match the lane markers found in the current video frame with the lane markers detected in the previous video frames. The algorithm will warn the driver if the vehicle moves across the lane marker. The algorithm is pretty robust against multiple road conditions. For the parameter setting of the algorithm, the number of row in the image being process starts from row 800 since the size of the image is 1080 x 1920. The maximum allowable change of lane distance metric between two frames is 50 pixels. The minimum number of frame a lane must be detected to define a valid lane is 10 while the maximum number of frame can be missed without marking it invalid is 10. By setting those parameters, the algorithms can have good performance against different driving circumstances. Our work is based on MATLAB Computer Vision System Toolbox. Below is the result when testing on the real-time driving video.

Figure 1.a: Result 1 in real time video

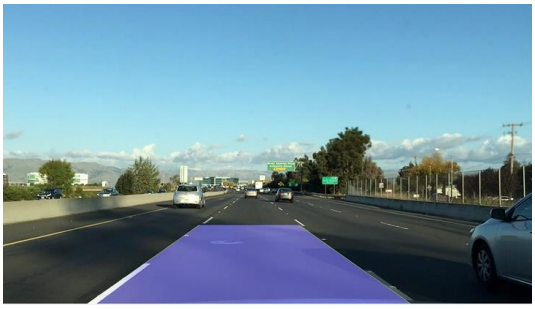

Figure 1.b: Result 2 in real time video

3.2 Traffic Sign Detection

We use viola and Jones Detector [7] to detect traffic signs.

3.2.1 Algorithm

1) Features

Viola and Jones propose to use Harr-like feature extractor to extract features:

$$feature = \sum pixels\ in\ white - \sum pixels\ in\ black$$

Then we normalize the features to make it have a norm of 1:

$$feature = \frac{feature}{areas\ of\ the\ feature\ window}$$

According to Viola et al., for a 24 by 24 feature window, we can extract approximately 160,000 dimensions of features.

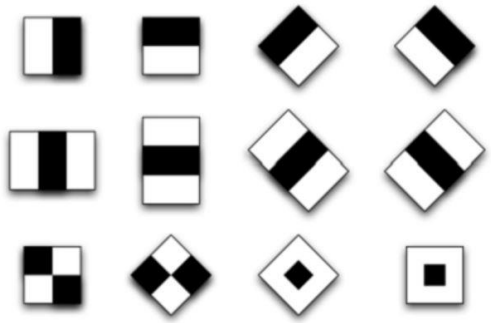

Figure 2: Harr-like Feature Templates

2) Classification

160,000 dimensional feature is a large number. Viola et al. assume in the paper that only a small fraction of these features are useful and that the main problem is how to find useful features for this small fraction. Adaboost classifier is a suitable suitable choice, because this classifier has feature selection capabilities. The principle of Adaboost is to construct a strong classifier connected in parallel by multiple weak learners. Each weak learner multiplies its classification result by weights according to its own accuracy. The final output is the sum of all classifiers output. The classification accuracy of each weak classifier can be very low, but the accuracy of the whole strong classifier is very high.

Traditional adaboost classifier is still too time-consuming. Viola et al. creatively modified the adaboost classifier to a cascade of several adaboost classifiers. Each classifier has high true positive rate (about 99%), while false positive rate is also high (about 50%). However, if we cascade 20 such small adaboost classifier, the false positive rate will become $(50\%)^{20} = 9.5 \times 10^{-7}$, while the true positive rate remains high.

3) Optimizing efficiency by computing integral image

Viola et al. propose to use integral image to improve computing efficiency.

$$I_{img}(x,y) = \sum_{i \le x, j \le y} img(i,j)$$

This can be done within O(mn). Once we have the integral image, we can compute one feature within constant time.

$$sum(a,b,c,d) = I_{img}(c,d) + sum(a,b) - sum(a,d) - sum(c,b)$$

3.2.2 Implementation Detail and Result

In our project, we choose three traffic signs to detect: stop sign, keep right sign, pedestrian crossing sign. LISA Traffic Sign Dataset is selected to train our feature detector. For each traffic sign, we use 60~80 training samples. We cascade 10 adaboost classifier for classification. Our code is built upon MATLAB vision toolbox.

Figure 3: Selected Traffic Signs

Below are some of our results. Once the feature detectors are trained, they can detect our interested areas very efficiently. Using this method, we can do real-time traffic sign detection.

Figure 4: Traffic Sign Detection Results

3.3 Car Make Identification

Car make identification based on the behind view has been a challenging problem. Since pictures of cars are 3 dimensional images, traditional image matching method such as SIFT or SURF descriptor cannot give accurate results. Besides, the difference among different sedan cars are trivial, it's also difficult to achieve accurate results using PCA or Fisher LDA. To achieve more accurate results, we propose to use deep learning methods. In our project, we tried two method. One is a regular 2-layer neural networks, and the other one is convolutional neural networks. Both give better results compared to traditional image matching methods.

Figure 5.a: Original images of Hyundai Sonata

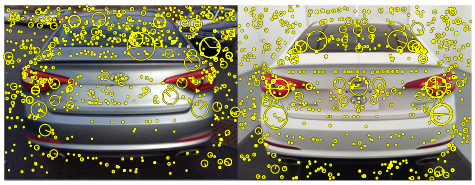

Figure 5.b: SIFT key points overlaid on images

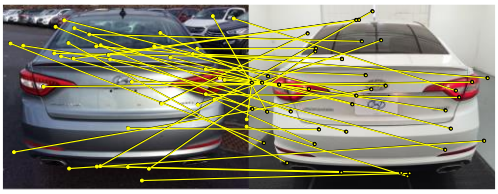

Figure 5.c: Feature correspondences after distance ratio test

Figure 5.d: Feature correspondence after RANSAC

3.3.1 SIFT Descriptor

For car make identification, we first came up with using the SIFT descriptor and SIFT matching with RANSAC. The matching results of two images from Hyundai Sonata are shown in Figure 5.

From Figure 5, we observe that only a very few points get matched while those points don't indicate the same feature on the vehicle. SIFT matching is robust with matching among planes. However, it cannot handle 3D image matching.

3.3.2 Data Collection

In order to train the models on 2-layer neutral network and convolutional neutral network. For each vehicle model (Acura ILX, Honda Civic, Hyundai Sonata) investigated in this project, 120 photos of each car make rear view have been collected. So there are totally 360 photos for the entire data set. A portion of photos are obtained from google images while another portion we collected using photos taken on I-101 and I-280. There are some constraints on gathering data. One constraint is that there is not a comprehensive database for rear view of vehicles on the web and the number of vehicle rear view picture from google images is limited. Also there are time and road condition limitations for our team to take rear view pictures from the vehicles on the road. Our team chooses these three sedans as the target of investigation not only because they are cars normally seen on the road but also because they have similar appearances and not easy to identify even for the human eyes. In the training process, 100 out of 120 (83.3%) photos from each car make randomly chosen from the data set. The test data is the remaining 20 out of 120 (16.7%) photos from each vehicle make.

3.3.3 2-layer Neural Networks

1) Architecture

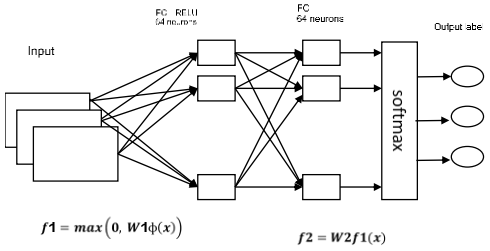

Figure 6: Architecture of our neural networks

Our input images are 64 by 64 by 3 RGB images. The neural networks we use have two fully connected layer. In each FC layer, there are 64 neurons. Each neuron is has the same dimension as the input image. In the output layer, we choose to use softmax loss function to output class scores and select the class label which has the largest class score as our output class label.

2) Regularization

To prevent overfitting, we apply L2 regularization to our networks. We penalize the squared magnitude of all parameters by adding the term $\frac{1}{2}\lambda\omega^2$ to every weight of our networks, where λ is the regularization strength.

3) Results

Below is the weight of the second FC layer of our neural networks. Loss history can be seen from Figure 7 and Figure 8. As can be seen from, our loss

converges after around 5000 iterations. The accuracy on the testing set can be up to 75%.

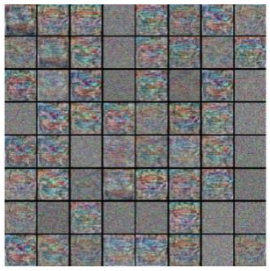

Figure 7: Weights of the 2nd layer

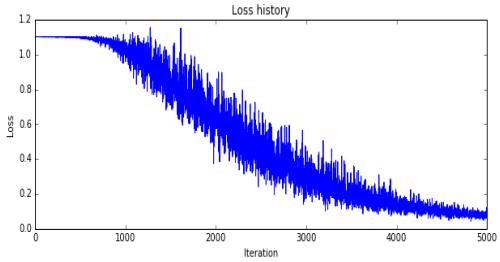

Figure 8: Loss History of Training

3.3.4 Convolutional Neural Networks

1) Architecture

We implement a simple convolutional neural networks based on Tensorflow. Our networks have 3 convolutional layers. Each convolutional layer comes with a max pooling layer to do down-sampling. The filters we use in each layer are specified in the figure 10. We also use a softmax classifier to output class scores.

2) Regularization

This time we use dropout [9] method to prevent from overfitting. While training, neurons are dropped out (set to zero) at certain probability. No dropout is implemented during testing.

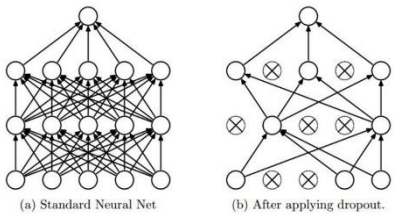

Figure 9: Dropout. Figure taken from "Dropout: A Simple Way to Prevent Neural Networks from Overfitting" that illustrate the idea

3) Result

Our loss function converges after around 6000 iterations and can achieve up to 78% accuracy on our testing set.

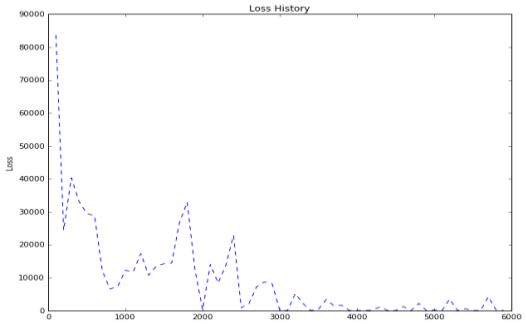

Figure 11: Loss History of CNN Training

4. Discussion and Future Work

4.1 Accuracy of Car Make Identification

For the project, we only implement two simple neural networks. They give relatively good result compared to SIFT matching. Since we only have 100 training images for each car make, the trained networks might be biased so that the accuracy stays around 75%~78%. We believe the accuracy can be improved by using larger datasets. Besides, in our results, a regular neural networks and a convolutional neural networks both achieve satisfying results. However, we still consider convolutional neural networks a better method to do car make identification. Regular neural networks will fail to handle huge number of parameters and lead to overfitting when input images are large [8]. With larger datasets and deeper convolutional neural networks, our proposed method can be robust and accurate in car make recognition.

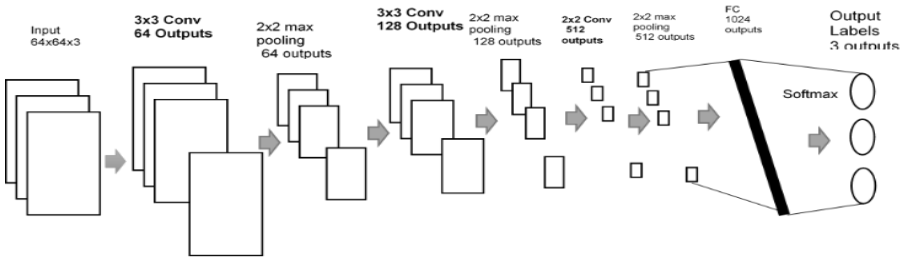

Figure 10: Architecture of Convolutional Neural Networks

4.2 Removing False Positives

Although cascading several adaboost classifiers can effectively reduce false positive rate. Our results still suffer from false positive detections when doing real time tracking as shown in Figure 13. False positives confuse the drivers and can be dangerous. Hence we did several optimization to reduce false detection.

Figure 12: False Positive Detection

First, it is observed that false positive areas are not stable. It may appear in one frame, but disappear in another frame. So we only keep those detected areas that appear in several frames as "true positive area". Second, most traffic signs are on the right side of the video. Our detector only target on those areas that are most likely to have traffic signs. This not only reduces false positive rates, but improves the efficiency as well.

However, there are still some false positives that cannot be removed by our proposed methods. Figure 12 is an example. It can be seen that the detected areas are very similar to the keep right sign. We find that traffic signs that have relatively simple patterns are more likely to be falsely detected, since their features are not distinct enough. Our method works pretty well for stop signs. However when it comes to keep right signs and pedestrian crossing signs, we get a few false positives. Here comes a paradox. When people design a traffic sign, they want to make it simple and easy to remember. However, what are simple to human can confuse computers sometimes, for they do not distinguish themselves from other objects well enough.

4.3 Future Work

Future improvement will mainly focus on improving the robustness of the system in the areas of traffic sign detection and vehicle make identification. To increase the robustness and the accuracy of the traffic signal detection on any signs against different weather condition, more dataset need to be collected and trained in the more complex algorithm model, especially for traffic signals like "keep on right lane". For the vehicle model identification, deeper convolutional neural network will be implemented and larger training data set will also be collected in order to substantially increase the accuracy of vehicle make detection on the real-time road.

5. Acknowledgement

We would like to thank Professor Gordon Wetzstein for his support and instruction throughout the course. We also want to thank Hershed Tilak for his mentorship on our project. Finally, many thanks to Yiran Deng and Haoxuan Chen, for their help on teaching us to use Python and Tensorflow.

6. Reference

[1] Gavrila, D. M. (1999). Traffic sign recognition revisited. In Mustererkennung 1999 (pp. 86-93). Springer Berlin Heidelberg.

[2] García-Garrido, M. A., Ocana, M., Llorca, D. F., Arroyo, E., Pozuelo, J., & Gavilán, M. (2012). Complete vision-based traffic sign recognition supported by an I2V communication system. Sensors, 12(2), 1148-1169.

[3] He, J., Rong, H., Gong, J., & Huang, W. (2010, November). A lane detection method for lane departure warning system. In Optoelectronics and Image Processing (ICOIP), 2010 International Conference on (Vol. 1, pp. 28-31). IEEE.

[4] Saha, A., Roy, D. D., Alam, T., & Deb, K. (2012). Automated road lane detection for intelligent vehicles. Global Journal of Computer Science and Technology, 12(6).

[5] Sharma, S., & Shah, D. J. (2003). A much advanced and efficient lane detection algorithm for intelligent highway safety. Computer Science & Information Technology, 51.

[6] S. Cheung, A. Chu. Make and Model Recognition of Cars. https://cseweb.ucsd.edu/classes/wi08/cse190-a/reports/scheung.pdf

[7] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on (Vol. 1, pp. I-511). IEEE.

[8] CS231n lecture notes

[9] Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(1), 1929-1958.

7. Appendix

Traffic Sign Recognition: Chen Zhu

Vehicle Deviation Detection: Zihui Liu

Vehicle Make Detection: Chen Zhu & Zihui Liu

Proposal: Chen Zhu & Zihui Liu

Poster: Chen Zhu & Zihui Liu

Final Report: Chen Zhu & Zihui Liu