Drivers, Focusing on Your Driving!
Yundong Zhang
CS229 Machine Learning Project Poster
Department of Computer Science
yundong@stanford.edu

Abstract
This project aims to build a computer vision system to detect and alarm the distracted driver. Using the dataset provided by Kaggle, we are interested in using machine learning algorithm to solve the problem. Several methods have been tested and the Convolution Neural Net proves its state-of-the-art performance. Specifically, utilizing the pre-trained VGG network we are able to achieve an accuracy of around 90%, with cross-entropy loss 0.24052.

Motivation
Drivers are supposed to be focusing on driving by law. However, it is very common to see drivers doing something else while driving: texting, drinking, operating the radio, talking on the phone and etc. This distracted behaviors easily cause crash incidents. In US, each day there are over 8 people killed and 1,161 injured in crashes due to a distracted driver, translating to 423,765 people injured and 2920 people killed each year. In China, 30 percent of the car incidents are caused by using cell phones. We hope to design an alarm system that can detect the distracted behavior of car drivers by using a dashboard camera.

Data Preparation
We use the Kaggle Dataset: State Farm Distracted Driver, which consists of 10 classes of image data taken in a car for training. There are in total 22424 training images (labeled) and 79727 test images (unlabeled), each is of size 640x480 pixels.

Approach
SVC
step 1. Bounding Box on human body To make the SVC classifier focus on the indicative features, we train a bounding box according to the HOG features and crop the image. The images are then re-sized to 224x224 for further processing.
step 2. Use PCA for noise removal, dimension reduction & decorrelation To make the small objects more influential in the classification–95% variance is preserved.

Deep Learning using CNN
1. Training CNN from scratch
A simple network structure similar to MNIST digit recognition, consists of three convolution layers connected by Maxpooling.
2. Transfer learning using VGG-16
   - mean normalization and random rotate 45- 10 degrees;
   - 25 Epoch with early-stopping, 5-fold Cross-validation;
   - Learning rate start with 1e-4, decay with 1e-6, batch size 12;
3. VGG-CAM Network
   • Remove the layers after VGG-conv5, resulting in a resolution of 14x14;
   • Add a convolution layers with 1024 filters of 3x3, followed by an average pooling layer of 14x14;
   • Adam optimizer, batch size 16, learning rate start with 1e-5, decay with 1e-6;
   • Visualisation: Map the prediction score back to the added convolution layers, and up-sampling to the input shape.

Semi-supervised Learning
Use Pseudo-label method (equivalent to Entropy-regularization): label test-set image with the class that is predicted with the highest probability. We infuse the train set with additional 10,000 images from test-set.

Result

<table>
<thead>
<tr>
<th>Model</th>
<th>Local accuracy</th>
<th>Local log-loss</th>
<th>Kaggle Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pure SVC</td>
<td>18.3%</td>
<td>1.94</td>
<td>1.98</td>
</tr>
<tr>
<td>SVC + PCA</td>
<td>34.9%</td>
<td>1.72</td>
<td>1.69</td>
</tr>
<tr>
<td>SVC + PCA + HOG</td>
<td>40.7%</td>
<td>1.41</td>
<td>1.55</td>
</tr>
<tr>
<td>Scratched CNN</td>
<td>63.3%</td>
<td>0.99</td>
<td>0.92</td>
</tr>
<tr>
<td>VGG16</td>
<td>90.2%</td>
<td>0.23</td>
<td>0.24</td>
</tr>
<tr>
<td>VGG-CAM</td>
<td>91.3%</td>
<td>0.27</td>
<td>0.32</td>
</tr>
<tr>
<td>Ensemble VGG16 &amp; VGG-CAM</td>
<td>92.6%</td>
<td>0.23</td>
<td>0.24</td>
</tr>
</tbody>
</table>

Table 1: Model Score

Discussion
• The ensemble model reduce the variance of the model and achieve the highest Kaggle score, which ranks top 8.
• The HOG feature bounding box does not seem to improve the performance of VGG16 in my experiment, mainly due to the quality of cropping is not so good. FRC would be a better candidate.
• Might be better to use even more heavier data argumentation, e.g. zero-out a random box in the image, to prevent over-fitting.

Figure 1: From left to right: (1) talking on the phone (right); (2) texting (left); (3) drinking; (4) hair and makeup; (5) operating the radio

Figure 2: Top: Original Images; Bottom: Images cropped by body using HOG bounding box

Figure 3: Top left: scratched CNN; Top right: Network in Network VGG-16; Bottom: VGG-CAM

Figure 4: Left: confusion matrix of VGG-16; Right: Learning curves

Figure 5: Visualize what the Neural Network is looking for

Figure 6: left: sliced CNN; right: Network in Network VGG-16; Bottom: VGG-CAM

Figure 7: Visualize what the Neural Network is looking for