Distracted Driver Detection:
Illustrated through convolutional neural nets and transfer learning

Kavya Mohan Sahai
MS Software Engineering
San Jose State University
kavyamohan.sahai@sjsu.edu

Neil Shah
MS Software Engineering
San Jose State University
neil.m.shah@sjsu.edu

Prabha Veerubhotla
MS Software Engineering
San Jose State University
vvlprabha@gmail.com

Harshita Yerraguntla
MS Computer Engineering
San Jose State University
Harshitha.yerraguntla@sjsu.edu

Abstract* - Road accidents are on the rise. According to the study of the National Roadway Traffic Safety Administrator, almost one of every five vehicle crashes are brought about by a driver that is distracted while driving. We attempt to build a robust machine learning system that can detect distracted drivers accurately and help take precautions against road accidents. Persuaded by the exhibition of Convolutional Neural Networks in PC vision, we present a CNN based framework that not just recognizes the occupied driver yet, in addition, distinguishes the reason for the diversion. We go through several different machine learning models starting with SVMs, simple 2-layer CNN, multi-layer CNN and then finally a modified VGG-16 model that achieves a 99.13% accuracy in predicting the exact type of distraction. Further, we present a transfer learning technique that achieves 97.7% accuracy with far lesser training time than the VGG-16 model.

Keywords *- driver distraction, safety, computer vision, SVM, CNN, VGG-16, transfer learning*

1. INTRODUCTION

The Center for Disease Control and Prevention (CDC) found that nearly one in five motor vehicle accidents were caused by distracted driving. This translates to 425,000 people injured and 3,000 people killed by distracted driving every year. Motivated to reduce these statistics, the purpose of this project is to accurately classify what drivers are doing and detect if they are distracted. The input to our models is images of people driving. Each image belongs to one of the ten classes.

We used different types of models - SVM, convolutional neural networks (CNN) and VGG-16 with transfer learning to predict which class the given images belong. The output is a list of predicted class labels.

II. RELATED WORK AND BACKGROUND MATERIAL

In this section, we condense some of the literature reviews that were explored during the course of the project.

A. Dataset

We found that the primary factor in driver distraction results from cell phone usage [2]. Spurred by this information, a portion of the analysts took a shot at mobile phone utilization discovery while driving. Zhang [4] utilized a camera mounted over a dashboard and recorded data in order to check mobile phone usage. In 2015, Nikhil [5] made a dataset for hand recognition in the car condition and accomplished an accuracy of 70.09% by utilizing Aggregate Channel Highlights (ACF) object identifier. People further utilized Supervised Descent Method, Histogram of Angles (HoG) and an AdaBoost classifier and accomplished 93.9% order accuracy in prediction.

UCSD's Laboratory of Intelligent and Safe Automobiles have also made contributions to this space, however, have only divided the distraction into three classes - altering the radio, changing mirrors and operating the gear. Martin [6] introduced a vision-based investigation structure that perceives in-vehicle exercises utilizing two Kinect cameras that gave frontal and back perspectives on the driver to give "hands on the wheel" data.

A more comprehensive dataset was still required. Zhao [7] planned a comprehensive driving dataset with side perspective on the driver thinking about four classes: safe driving, operating levers, eating and chatting on a cellphone.

B Support Vector Machines

In order to tackle the Distracted Driver Detection problem, the first model utilized was SVM. Support Vector Machines are discriminative classifiers that are based on statistical learning theory and can be used to identify patterns, classify them and infer non-linear relationships between variables [11]. Given labeled data, SVM generates an optimal hyperplane which classifies the new data points. In an n-Dimensional space, this hyperplane is an n-1 plane dividing the space into parts. SVMs can generate both non-linear as well as linear classifiers efficiently with different kernels and can pull useful data from noisy data thus avoiding overfitting by minimizing the upper bound of generalization error [10].

C. Convolutional Neural Networks

Convolutional Neural Network (CNN) is an artificial neural network, mainly used for image recognition which is one of the main reasons it was chosen to tackle this problem. The pattern detectors or filters to detect images are in the hidden layers, which are called convolutional layers in CNN. The filters convolve with the pixels on the image as they move over the whole image, and this output is passed onto the next convolution layer. The initial convolutional layers contain filters that detect simple shapes such as circles, squares and a car windshield. As you go further deep, the layers can recognize more complex objects like hair, fur and also a complete animal as an object. These are the fully-connected layers. This can be mirrored to how different regions in the human brain get triggered on seeing different objects, likewise different convolutional layers filter objects with different shapes and sophistication.

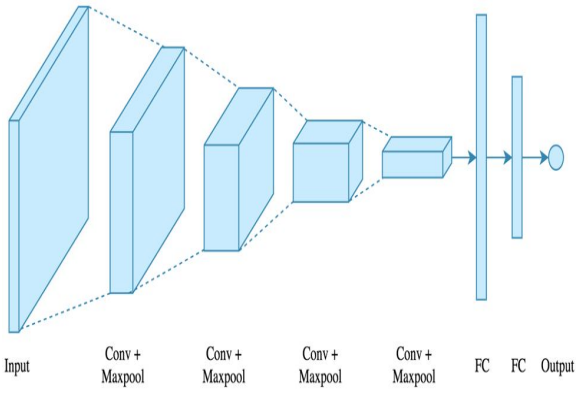

Figure 2. CNN with 4 convolutional layers and 2 fully connected layers

D. VGG16

VGG16 is a CNN presented by Andrew Zisserman and Karen Simonyan from the University of Oxford in their research paper Very Deep Convolutional Networks for Large-Scale Image recognition [9]. It is a 16-layer network with 13 convolutional layers and 3 fully connected layers named after University of Oxford's Visual Geometry Group. VGG used NVIDIA Titan Black GPUs, was trained for weeks and achieved an accuracy of 92.7% [9] on the ImageNet dataset which consists of over 15 million labeled high-resolution images with around 22000 categories. VGG and its weights are available online and were used in our project to reduce time and computation cost. Figure 3 shows the architecture of VGG16 model.

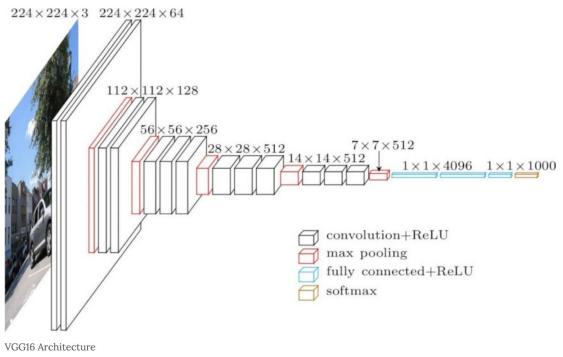

Figure 3. VGG16 Architecture

E. Transfer Learning

Transfer learning is the idea of using a model pre-trained for a particular problem as an initialization for a different but similar problem. It gives one a significant boost in terms of speed and performance. Two strategies employed are - 1) to either use the weights of a pre-trained model and train it using these as base weights or 2) to use the pre-trained model as a feature extractor. These can be achieved by using the model without its fully connected layers because these are the layers which were trained specifically for the original dataset. In VGG16 pre-trained on ImageNet, the initial layers which can detect edges, blobs, etc can be used and further the fully connected layers can be re-trained on a new dataset [12]. This learning technique was used in our project as a way to overcome computational constraints.

III. TECHNICAL APPROACH

In this section, we will understand how a CNN is able to identify patterns in the input image. As we saw, there are three different components in a CNN- convolutional layers, pooling layers, and fully connected layers.

Convolutional layers consist of kernels that learn local features in an input image and then these features are convolved over the image resulting in an activation map. The activation map is a matrix with high and low values - high if the kernel feature is present at that position in the image, otherwise low. The values are decided based on the following formula where m is the kernel width and height, h is the convoluted output, x is the input and w is the kernel.

$$h_{i,j} = \sum_{k=1}^{m} \sum_{l=1}^{m} w_{k,l} x_{i+k-1,j+l-1}$$

Next, the pooling layer is responsible for making these features translation invariant- reduce the number of parameters in the network so as to control overfitting. The formula for the pooling operation which outputs the maximum value of input in the kernel, is given below.

$$h_{i,j} = max\{x_{i+k-1,j+l-1} \forall 1 \leq k \leq m \text{ and } 1 \leq l \leq m\}$$

Finally, the fully-connected layers produce various activation patterns based on features learnt by various kernels so as to present a global representation of the entire image. The neurons in these layers get activated if the objects represented by the features are there in the input therefore producing different activation patterns depending on the features present and providing a holistic view of what is present in the image to the output layer which it then uses to correctly classify the image.

IV. EXPERIMENTAL METHODOLOGY

A. Dataset and Features

The dataset that we have used is composed by State Farm Insurance. In 2016, State Farm presented this

(a) C0: Drive safe (b) C1: Text left (c) C2: Talk left (d) C3: Text right (e) C4: Talk right

(f) C5: Adjust radio (g) C6: Drink (h) C7: Hair and makeup (i) C8: Reaching behind (j) C9: Talk to passanger

Figure 1: Ten Classes of Driver Postures from the Dataset

dataset in a kaggle competition named "Distracted Driver Detection". They have defined 10 different classes to be predicted - Safe driving + 9 different types of distractions. This is the first publicly available highly comprehensive dataset. We found that CNNs proved to be the most effective models [8] in achieving high prediction accuracy.

The data was supplied by State Farm for a public Kaggle challenge. The dataset consists of 22400 training and testing images (640 × 480 full color) of people either driving safely or doing one of nine kinds of distracted behaviors. Training and testing are split in the ratio of 70:30.

The training images come with correct labels and the challenge is to make the best multi-class classifications possible. Figure 1 shows the 10 different classes for prediction.

B. SVM

The trial and error of our experiments started with a Support Vector Machine(SVM) classifier. We decided to try this classifier first as we had recently learned about it in our course. The training dataset images were resized to 60x60. We tried with linear, Gaussian and polynomial kernels. Gaussian gave the highest accuracy of 69%. The setup was a MacBook Pro(8GB 2.7GHz i5). Here, we used global features and labels (from the train data) to train the SVM. With only a single vector, global features can generalize an entire image. Seeing the low accuracy, we moved onto convolutional neural networks, the most widely used classifier for images and computer vision.

C. CNN

We initially started with a simple CNN model - 2 convolutional layers and 2 fully connected layers. Conv2D, Maxpool2D is used for the convolutional layer. These functions are useful for spatial convolution and pooling operation over data. We can customize the functions by tuning the parameters like the number of filters, kernel size, padding, activation unit type. This simple CNN is tested on a Pro(16GB, 3.1GHz, i5). The image is resized to 24x24 to make it compatible with the setup. Rectified Linear Unit(ReLU) activation is used for all the convolution layers. We chose, ReLU as the activation function as it is simple and computationally cheap.. Softmax activation is used in the output layer, to get the probability(the confidence with which the neural network predicts the image) of different kinds of driver distractions. Softmax converts logits or numbers into probabilities. With this simple CNN, we got an accuracy of 12%. We reasoned it to the resized image (24x24) as a 24x24 image could not retain a lot of information.

D. CNN with more layers

Owing to the low accuracy with the simple CNN, we added more convolutional layers (conv2D, MaxPool2D) with different number of filters and kernel size. The new CNN (Figure 3) consists of 4 convolutional layers, 1 fully connected layer. To give our experiment ample resources, we moved to Google Cloud Platform(GCP) to a setup with 8 vCPUS and 52GB. With the same ReLU activation and the softmax for the output layer, the CNN gave an accuracy of 96.88%. The image size is the same as the given image size (480x360) from train dataset. The accuracy did not get affected after 10 epochs. So, we concluded this experiment with the 10th epoch. Now, we understand that not only we need a good CNN, but also sufficient computational resources to train a good model, for better confidence in predicting the image.

E. VGG-16 with Transfer Learning

The VGG16 CNN [9] is pre-trained on the ImageNet dataset and available through the *Keras* library. We removed the fully connected layers from VGG16 as their weights were specific to ImageNet dataset and added a fully connected classifier as done in section 4.4 on top of it. The ImageNet weights of the VGG16 model were extracted and fed to the fully connected layers to re-train them on the Kaggle dataset.

The training was done similar to that of the CNN with more layers in 4.4. The model was trained on GCP with 8 vCPUs and 52GB RAM. The image size was taken to be 224x224. The accuracy did not show any more positive changes after the 7th epoch so the experiment was concluded with the 7th epoch

TRAINING MODEL	DATA / PARAMETERS	ACCURACY
SVM	60 X 60 image size, Gaussian Kernel	69%
2-LAYER CNN	24 X 24 image size Activation – ReLU Conv Layer – Conv2D Maxpool – Maxpool2D	12%
5-LAYER CNN	480 x 360 image size Epochs – 10 Batch Size – 16 Conv Layer – Conv2D Maxpool – Maxpool2D	96%
VGG-16	224 x 224 grayscale image Epochs – 10 Conv Layer – Conv2D Maxpool – Maxpool2D	99.5%

Table 1: Models & Accuracy

giving an accuracy of 99.5%. We were surprised by this result as the accuracy was high and the model took only an hour to train in contrast to the previous models that took more than 12 hours.

F. Results and Discussion

The overall results of our various attempts along with the respective accuracy scores are shown in Table 1 (above).

As you can see, we initially started with the SVM and experimented with different hyperparameters. The best accuracy was achieved by the Gaussian Kernel. We were also computationally constrained and hence the image size was reduced to 60 x 60 which in effect reduced our feature set as well. Next we went on to implement a Convolutional Neural Network, starting with 2-layer and then increasing up. The final winning model was the VGG-16 model that achieved an accuracy of 99.5% where the data was transformed into 224 x 224 grayscale images and the training was run for 10 epochs.

As the time and memory requirements for the computation of the model were high, we moved all of our work onto a deep-instance of Google Cloud Platform and ran our model there.

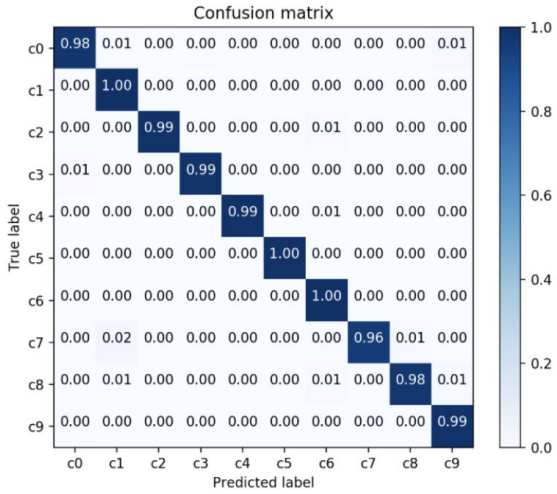

Figure 4: VGG-16 Confusion Matrix

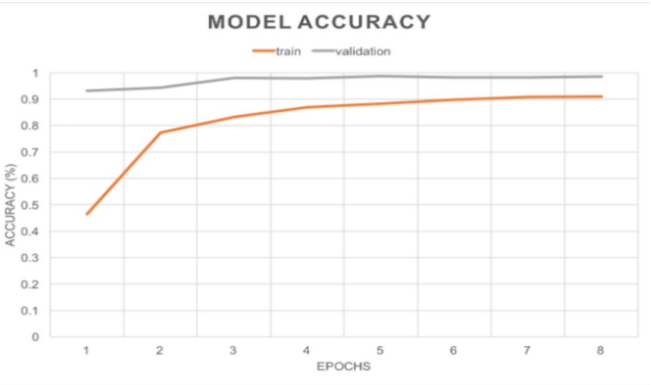

Figure 4 shows the confusion matrix for the VGG-16 model. It can be seen that although most classes have been predicted with a 100% accuracy, the class c7 has a 96% accuracy which is low in comparison to the others. c7 corresponds to the class *hair and makeup* and according to the confusion matrix, it can be concluded that this class is the most misclassified. We can also see that c7 is most commonly misclassified as c1 which correspond to *texting left* which makes sense since the driver would make their hair or put makeup using their left hand.

V. CONCLUSION

Distracted drivers are one of the primary causes of accidents on the road. This project is a stepping stone in fighting the ultimate goal of using computer vision to help fight the distracted driver problem. In this project, we have explored several different machine learning models and finally arrived at the VGG-16 model that gives us an accuracy of 97.7% with only one hour of learning using the transfer learning technique and a pre-trained model. During the course of this project, the importance of data pre-processing, overfitting, computational & time constraints became apparent and we overcame all of those obstacles to finally arrive at a usable learning model which can accurately predict a specific distraction of a driver.

Distractions are habitual and ingrained in our daily routine. Most people end up distracting themselves sub-consciously without realizing. With fine tuning and proper engineering, if this detection system was built into cars, it could help make the drivers consciously aware of their distraction and/or assist the car into an auto-pilot mode in the middle of a distraction to help avoid accidents.

REFERENCES

[1] Center for disease control and prevention. https://www.cdc.gov/motorvehiclesafety/ distracted_driving/

[2] National highway traffic safety administration traffic safety facts. https://www.nhtsa.gov/ risky-driving/distracted-driving/

[3] State farm distracted driver detection. https://www.kaggle.com/c/state-farm-distracted-driver-detection.

[4] X. Zhang, N. Zheng, F. Wang, and Y. He. Visual recognition of driver hand-held cell phone use based on hidden crf. In Proceedings of 2011 IEEE International Conference on Vehicular Electronics and Safety, pages 248–251, July 2011

[5] N. Das, E. Ohn-Bar, and M. M. Trivedi. On performance evaluation of driver hand detection algorithms: Challenges, dataset, and metrics. In 2015 IEEE 18th International Conference on Intelligent Transportation Systems, pages 2953– 2958, Sept 2015.

[6] S. Martin, E. Ohn-Bar, A. Tawari, and M. M. Trivedi. Understanding head and hand activities and coordination in naturalistic driving videos. In 2014 IEEE Intelligent Vehicles Symposium Proceedings, pages 884–889, June 2014.

[7] C. H. Zhao, B. L. Zhang, J. He, and J. Lian. Recognition of driving postures by contourlet transform and random forests. IET Intelligent Transport Systems, 6(2):161–168, June 2012.

[8] R. P. A. S. Murtadha D Hssayeni, Sagar Saxena. Distracted driver detection: Deep learning vs handcrafted features. IS&T International Symposium on Electronic Imaging, pages 20–26, 2017.

[9] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[10] Vapnik, V. N.. Support Vector Networks. 1995.

[11] Vapnik, V. N.. The nature of statistical learning theory. New York: Springer, 1995.

[12] Francois Chollet. Building powerful image classification models using very little data, 2016.