Detection of Driver Distracted Behavior and Drowsy Driving

Group 15

Haoyan Yang 1930026145

Qirun Yang 1930026146

Yuao Qin 1930026111

Content

1. Background

Driving safety has always been a concern of people. During driving, distracted driving and drowsy driving are the two main causes of traffic accidents. The behaviors that lead to distracted driving include using phones, drinking. Research shows that a glance at a phone is equivalent to about 50 meters of "blind driving" of a car. Another major factor in driving accidents comes from drowsy driving. According to statistics, traffic accidents caused by drowsy driving account for 21% in China, and the fatality rate is as high as 83%.

2. Motivation

In order to reduce accidents caused by distracted behavior and drowsy driving, our project is to establish models to identify these two aspects of behaviors accurately. When the situations are detected, we can remind the driver to reduce the occurrence of accidents.

3. Existing Techniques

In the field of distraction detection, EEG signals have been proposed to analyze drivers' attention concentration. Someone also proposed a system based on feedforward neural network (FFNN) to monitor the three-dimensional head rotation Angle and the position of the upper body joint of the driver and identify the driving task.

Previous studies used neural networks and deep learning to detect face recognition, extracted region of interest (ROI) with facial markers, and finally detected head and eye movements. The video data is input to the system and the distracted results are output to the system.

There are also recognition models for pupil detection. The viola-Jones algorithm was used to extract face regions from input frames. The template matching algorithm was used to extract the eye position and estimate the gaze position according to the distance between the canthus and the pupil to judge the distraction.

In recent years, driving detection based on physiological signals such as electroencephalogram (EEG), electrocardiogram (EOG), electromyogram (EMG) and electrocardiogram (ECG) has been widely used. EEG has been widely used because of its advantages of high time resolution, good portability, and sensitivity to fatigue.

The above is a brief introduction of the existing detection techniques and related works of distracted driving behaviors, and the following is about drowsy driving.

Drowsy driving will lead to obvious abnormalities in the driver's control of the vehicle, so driving operation parameters are used to monitor drowsy driving, including steering wheel Angle, lane departure and driver's attitude.

Steering wheel Angle is the most commonly used parameter to monitor driving fatigue, which shows a good advantage in characterizing driving fatigue. Based on the instruments of the National Advanced Driving Simulation Laboratory at the University

of Iowa, the method of monitoring drowsy driving was studied by detecting steering wheel Angle parameters and using random prediction algorithm. Based on the theory of nonlinear feature construction, a multi-stage drowsiness determination model was established by calculating the approximate entropy (ApEn) and complexity of the steering wheel Angle short time series.

In addition to steering wheel Angle parameters, track deviation parameters are often used in drowsy driving monitoring. Drivers were tested for drowsy driving on a simulated driving test-bed, and a determination model of trajectory deviation parameters and drowsy degree was established. A gauss-Hidden Markov drowsiness lane departure recognition model was established, and the model parameters were obtained by off-line training. By comparing the effects of different models and parameters on the recognition effect, it is concluded that the drowsiness lane departure recognition model based on GM-HMM has the best recognition effect.

Part of the driver's face information and head movement signals can also be a good representation of driving fatigue, such as eye closing time, fixation time, driving posture, yawning, nodding frequency, etc., are typical signals reflecting driving fatigue. Generally speaking, drowsy monitoring technology based on driver's body features can be divided into eyes, mouth, and head detection according to the detection location, and feature values are extracted mainly through machine vision and image processing technology.

The detection of eye state is obtained by image acquisition equipment, and the feature information is obtained by image processing technology. The degree of drowsy driving was mainly reflected by extracting parameters such as the frequency of eye closure, the distance between upper and lower eyelids, and the percentage of eyes open.

Yawning is a typical manifestation of fatigue, so yawning and mouth opening can also be used to characterize the driver's drowsy state. In the real driving environment, the driver's face image was collected, the features of mouth state were extracted by support vector machine and edge slope algorithm, and the degree of drowsiness was represented by counting The Times of yawn.

Driving fatigue, especially drowsiness, causes drivers to nod off, which is manifested in frequent nodding movements in the driver's physical characteristics. Drowsy driving can be detected by counting the number of nods per unit time through video image processing.

4. Introduction to Dataset

The link of all datasets:

https://www.kaggle.com/datasets/joyyang123/mlgroup15dataset

4.1 Distracted behavior dataset

The dataset collects 8 states of drivers while driving, including safe driving and 7 different distraction behaviors, for a total of 22,424 images. A picture of each class behavior is shown below.

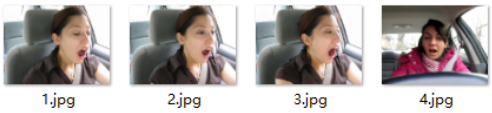
4.2 Drowsy driving dataset

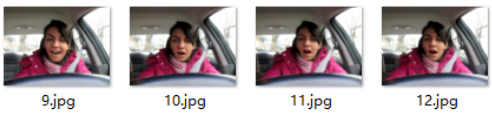
The images were divided into two categories, one is pictures of various people's eyes, which had two classes; closed and open eyes, and the other is images of various drivers, which had two classes; yawning and not yawning. Below we show some images from each class.

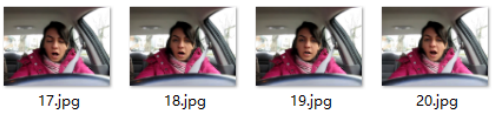
images which close eyes

images which close eyes

not yawning images







yawning images

5. Methodology

Recap we have two tasks; one is detection of distracted behaviors, and another is detection of drowsy driving. We believe that when identifying distracting behaviors, we are primarily concerned with the driver's overall movements. When recognizing drowsy driving, we pay more attention to the driver's facial details, including yawning or not, opening and closing eyes, etc. So, we split it into two parts to introduce the method we used.

Detection of distracted behaviors

5.1 HOG + Classifier

HOG, histogram of oriented gradients, the main idea is to analyze an image, and the appearance and shape of local targets can be well described by the distribution of shaving or edge density direction. We collect the direction histogram of mound or edge for each pixel of the image, and the characteristics of the image can be described according to the information of the histogram. According to the collected hog feature vector, it is used by the classifier. This is the flow chart of using HOG process combined

with classifier to classify distracted behaviors.

flow chart of using HOG process combined with classifier

When extracting HOG features, it is necessary to calculate the horizontal gradient and vertical gradient of each pixel, as shown in formula 5.1.1 and formula 5.1.2.

$$g_x(x,y) = H(x+1,y) - H(x-1,y) \qquad (5.1.1)$$

$$g_y(x,y) = H(x,y+1) - H(x,y-1) \qquad (5.1.2)$$

where g is gradient, H is pixel value of the corresponding point and x, y are horizontal and vertical directions respectively

The amplitude and angle of the gradient at this point can be obtained by formula 5.1.3 and 5.1.4.

$$g = \sqrt{g_x^2 + g_y^2} \qquad (5.1.3)$$

$$\theta = \arctan\frac{g_x}{g_y} \qquad (5.1.4)$$

Because the HOG feature is a local gradient feature, it is not sensitive to light, which is helpful to weaken the interference caused by light difference. This is useful in identifying distracted driving behavior. Because the road conditions and weather are different when the camera takes photos, we need to eliminate the influence of light in the picture. Here is the process of HOG feature extraction of an input image.

process of HOG feature extraction

After HOG feature extraction, we choose logistic regression and support vector machine models as our classifiers.

(1) Logistic Regression

The principle is to map the result of linear regression $(-\infty, \infty)$ to $(0,1)$ with sigmoid function. The linear regression function and sigmoid function are shown in formula 5.1.5 and 5.1.6.

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \boldsymbol{\theta}^T \mathbf{x} \qquad (5.1.5)$$

$$g(z) = \frac{1}{1+e^{-z}} \qquad (5.1.6)$$

Set a threshold between (0, 1). Each input x will produce a probability y after linear

regression and sigmoid function. y exceeding the threshold is classified one category, and the others are another category.

(2) Support Vector Machine

The goal of SVM is to learn to find a hyperplane with correctly classified samples and the largest geometric interval. The problem can be transformed into a constrained optimization problem by formula 5.1.7.

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad (5.1.7)$$

$$s.t\, y_i(\omega x_i + b) - 1 \geq 0, i = 1,2,\dots,n$$

The optimal hyperplane is obtained as formula 5.1.8 by using the algorithm of convex quadratic programming problem.

$$\omega^* x + b^* = 0 \quad (5.1.8)$$

The corresponding classification decision function as formula 5.1.9.

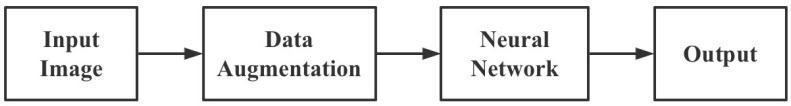
$$f(x) = \text{sign}(\omega^* x + b^*) \quad (5.1.9)$$

In the actual classification, the sample x is substituted into the above function, and when the result is +1, it is recognized as a positive sample, otherwise it is a negative sample.

Because what we need to solve is a multi-classification problem, we can use logistic regression and SVM to generate multiple binary classifiers, and then merge them to solve it.

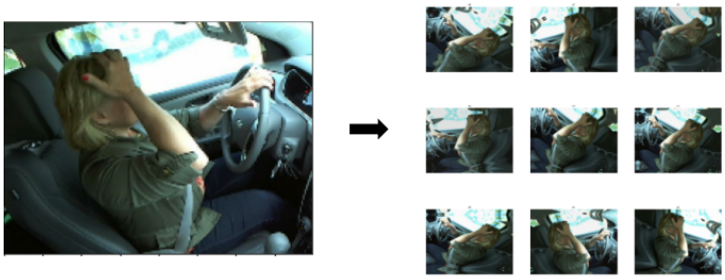
5.2 Neural network

For image recognition, neural network model is the mainstream method. When detecting distractions, we use two neural network models, Convolutional Neural Network (CNN) and multilayer stacking CNN.



flow chart of using neural network model

Before we train natural network model, we make the data augmentation process and transform the image to array. Data augmentation is to generate more training data from existing training samples to improve the generalization ability of the model.



data argumentation

(1) Convolutional Neural Network (CNN)

Convolutional neural network is a kind of feedforward neural network, which has excellent performance for large-scale image processing. The purpose of convolution operation is to extract different input features. The first convolution layer may only extract some low-level features, such as edges, lines and angles. Networks with more layers can iteratively extract more complex features from low-level features. The CNN model in this project includes convolution layer, activation layer, pooling layer, flatten layer and dense layer. Here is the model structure.

Layer (type)	Output Shape	Param #
img_augmentation (Sequential)	(None, 150, 200, 3)	0
rescaling (Rescaling)	(None, 150, 200, 3)	0
conv2d (Conv2D)	(None, 150, 200, 16)	448
activation (Activation)	(None, 150, 200, 16)	0
max_pooling2d (MaxPooling2D)	(None, 75, 100, 16)	0
conv2d_1 (Conv2D)	(None, 75, 100, 32)	4640
activation_1 (Activation)	(None, 75, 100, 32)	0
max_pooling2d_1 (MaxPooling2D)	(None, 37, 50, 32)	0
conv2d_2 (Conv2D)	(None, 37, 50, 64)	18496
activation_2 (Activation)	(None, 37, 50, 64)	0
max_pooling2d_2 (MaxPooling2D)	(None, 18, 25, 64)	0
flatten (Flatten)	(None, 28800)	0
dense (Dense)	(None, 128)	3686528
dense_1 (Dense)	(None, 8)	1032

CNN model structure in the project

(2) Multilayer stacking CNNs

Since the previous CNN model may be simple, we try to stack three CNN models to make the network structure more complex so that it has better performance probably. Considered that the model is complex, to avoid overfitting problem, we add dropout layer. Moreover, we add batch_normalization layer to speed up the convergence. Here is the model structure.

Layer (type)	Output Shape	Param #
conv2d_6 (Conv2D)	(None, 62, 62, 32)	320
batch_normalization_8 (BatchNormalization)	(None, 62, 62, 32)	128
conv2d_7 (Conv2D)	(None, 62, 62, 32)	9248
batch_normalization_9 (BatchNormalization)	(None, 62, 62, 32)	128
max_pooling2d_3 (MaxPooling2D)	(None, 31, 31, 32)	0
dropout_5 (Dropout)	(None, 31, 31, 32)	0
conv2d_8 (Conv2D)	(None, 29, 29, 64)	18496
batch_normalization_10 (BatchNormalization)	(None, 29, 29, 64)	256
conv2d_9 (Conv2D)	(None, 29, 29, 64)	36928
batch_normalization_11 (BatchNormalization)	(None, 29, 29, 64)	256
max_pooling2d_4 (MaxPooling2D)	(None, 15, 15, 64)	0
dropout_6 (Dropout)	(None, 15, 15, 64)	0
conv2d_10 (Conv2D)	(None, 13, 13, 128)	73856
batch_normalization_12 (BatchNormalization)	(None, 13, 13, 128)	512
conv2d_11 (Conv2D)	(None, 13, 13, 128)	147584
batch_normalization_13 (BatchNormalization)	(None, 13, 13, 128)	512
max_pooling2d_5 (MaxPooling2D)	(None, 7, 7, 128)	0
dropout_7 (Dropout)	(None, 7, 7, 128)	0
flatten_1 (Flatten)	(None, 6272)	0
dense_3 (Dense)	(None, 512)	3211776
batch_normalization_14 (BatchNormalization)	(None, 512)	2048
dropout_8 (Dropout)	(None, 512)	0
dense_4 (Dense)	(None, 128)	65664
batch_normalization_15 (BatchNormalization)	(None, 128)	512
dropout_9 (Dropout)	(None, 128)	0
dense_5 (Dense)	(None, 8)	1032

Multilayer stacking CNNs structure in the project

Detection of drowsy driving

We first judge whether the driver yawns or closes his eyes, which is two binary classification problems. Then select the best performing model for each task to merge. Combining the two aspects of yawning and closing eyes, judge whether it is drowsy driving according to the rules established by ourselves.

The flows of two problems are similar, as follows

a) Read in the images of drivers and convert the format to array for subsequent processing.
b) Apply grayscale processing.
c) Apply image enhancement. This image enhancement was carried out without changing the label. We apply imageDataGenerator (rescale=1/255) which comes

from keras.preprocessing.image).

d) Randomly divide all the images into training sets and test sets in a ratio of 7:3.

e) Use different models for training.

We still use two basic models, Logistic Regression and SVM, in each task. In addition, we also use a new and more complex model in each task.

5.3 MobileNet

In the task of judging the driver's eyes to open and close, we use the MobileNet model. It's a network built on depth-level separable convolution, which splits the standard convolution into two operations: Depthwise convolution and Pointwise convolution. Depthwise convolution is different from standard convolution. For standard convolution, its convolution kernel is used in all input channels, while Depthwise convolution uses different convolution kernel for each input channel, that is, one convolution check should have one input channel. So Depthwise convolution is a depth operation. In fact, Pointwise convolution is ordinary convolution, but it adopts 1x1 convolution kernel.

In addition to the standard convolution kernel used in the first layer, the remaining convolution layers use Depthwise convolution. The core of MobileNet is the use of Depthwise convolution, which can reduce the computational complexity of the model and greatly reduce the size of the model.

Type / Stride	Filter Shape	Input Size
Conv / s2	3 × 3 × 3 × 32	224 × 224 × 3
Conv dw / s1	3 × 3 × 32 dw	112 × 112 × 32
Conv / s1	1 × 1 × 32 × 64	112 × 112 × 32
Conv dw / s2	3 × 3 × 64 dw	112 × 112 × 64
Conv / s1	1 × 1 × 64 × 128	56 × 56 × 64
Conv dw / s1	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 128	56 × 56 × 128
Conv dw / s2	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 256	28 × 28 × 128
Conv dw / s1	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 256	28 × 28 × 256
Conv dw / s2	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 512	14 × 14 × 256
5× Conv dw / s1	3 × 3 × 512 dw	14 × 14 × 512
5× Conv / s1	1 × 1 × 512 × 512	14 × 14 × 512
Conv dw / s2	3 × 3 × 512 dw	14 × 14 × 512
Conv / s1	1 × 1 × 512 × 1024	7 × 7 × 512
Conv dw / s2	3 × 3 × 1024 dw	7 × 7 × 1024
Conv / s1	1 × 1 × 1024 × 1024	7 × 7 × 1024
Avg Pool / s1	Pool 7 × 7	7 × 7 × 1024
FC / s1	1024 × 1000	1 × 1 × 1024
Softmax / s1	Classifier	1 × 1 × 1000

MoibleNet structure

5.4 Ensemble learning

Ensemble learning is the process of constructing and combining multiple Base learners to accomplish a learning task. By training several individual learners, we can eventually form a strong learner by certain combination strategies and thus learn from all.

Ensemble learning is often considered as a meta-algorithm.
In the task of judging whether the driver is yawning or not, we will perform ensemble learning on three models, XGBoost and LightBGM in Boosting algorithm and RandomForest in Bagging algorithm.

- XGBoost

XGBoost is based on the gradient boosting decision tree (GBDT) boosting algorithm, and its core idea is that in the first step, keep adding trees, keep splitting features to grow a tree, and each time a tree is added, it is learning a new function to fit the residuals of the last prediction. The second step, when we finish training to get k trees, we have to predict the score of a sample, which is based on the features of this sample, in each tree will fall to the corresponding a leaf node, each leaf node corresponds to a score. In the third step, finally, we only need to add up the scores corresponding to each tree is the predicted value of the sample. The object function as shown by formula 5.4.1.

$$\mathrm{Obj}^{(t)} = \sum_{i=1}^{n} \left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) + \text{constant} \quad (5.4.1)$$

- LightBGM

LightBGM, on the contrary, is another step up based on the XGBoost ontology. Since XGBoost itself is based on decision trees, and the decision tree algorithm is firstly, all features are pre-sorted by the value of the feature. Second, the best segmentation point on a feature is found with O (All Data) cost when traversing the segmentation points. Finally, after finding the best split point on a feature, the data is split into left and right child nodes. This is destined to consume a lot of time and memory. In contrast, LightBGM is based on the Histogram-based decision tree algorithm. Using Gradient-based One-Side Sampling (GOSS): reduce many data instances with small gradients and use only the remaining data with high gradients when calculating information gain, which saves time and space complexity compared with XGBoost. Use Exclusive Feature Bundling to achieve data dimensionality reduction. Leaf splitting gain is enhanced with Leaf-wise leaf growth strategy with depth limitation. Also directly supports category features, efficient parallelism, and the presence of Cache hit ratio optimization.

- RandomForest

Random forest is implemented on the basis of bagging and decision trees and improves on both. Random forest uses a weak learner for decision trees, and there is no dependency between each decision tree, which can be generated in parallel. A normal decision tree selects the best one among all the n sample features on a node for decision tree partitioning, while random forest selects some of the features on a node (the number of features is less than n, and the fewer the number of features selected, the more robust the model is). Then an optimal feature is selected among the randomly selected partial features for tree partitioning (double-layer selection), which further enhances the generalization ability of the model.

5.5 Merge two subtasks result

When the two subtasks are completed, we select the best performing models in the two

tasks and merge them to judge whether the driver is drowsy or not. We consider that drowsy driving is not an instantaneous process, so we make rules that in 5 frame pictures, if the driver has more than 3 frames in a row where his eyes are closed, and he is yawning, we consider this driver to be driving in a drowsy state.

6. Experimental Study and Result Analysis

The following results show the performance of the model on the test set. Because we think that the good performance of the model in the training set is not enough to explain everything. The model should also have good performance in the test set, we think that this model can be used in practical problems.

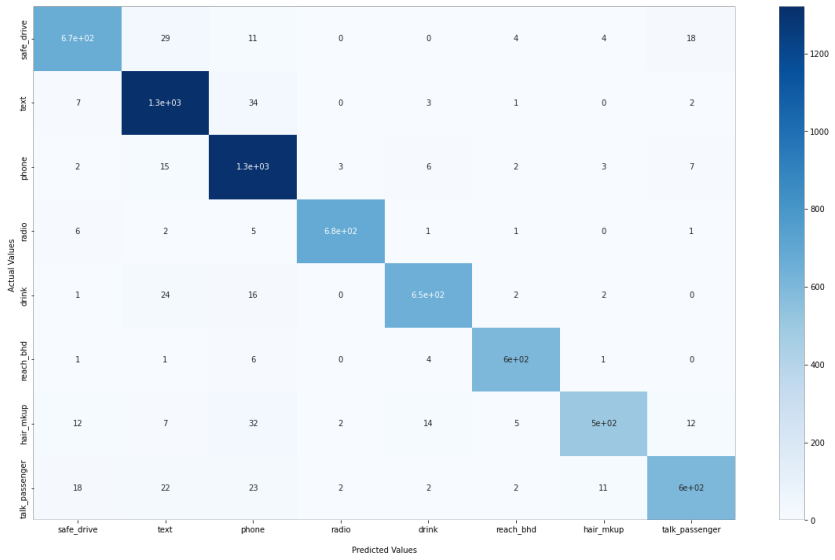
6.1 Detection of distracted behaviors

6.1.1 Logistic Regression

We set "L1" penalty and C=0.5 to regularize the model. The following is model performance and confusion matrix.

Accuracy	Precision
94.20%	94.89%

Logistic Regression performance



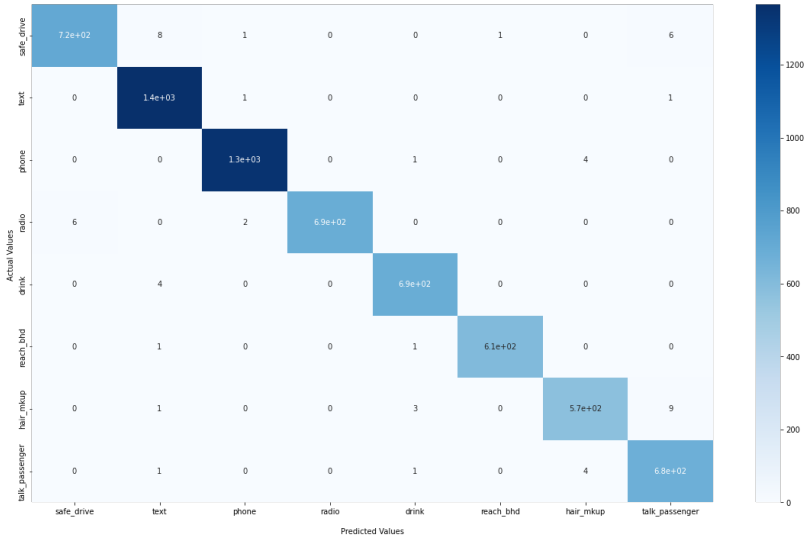
Confusion Matrix of Logistic Regression

6.1.2 SVM

We use "rbf" kernel and let penalty coefficient C=1. The following is model performance and confusion matrix.

Accuracy	Precision
99.17%	99.13%

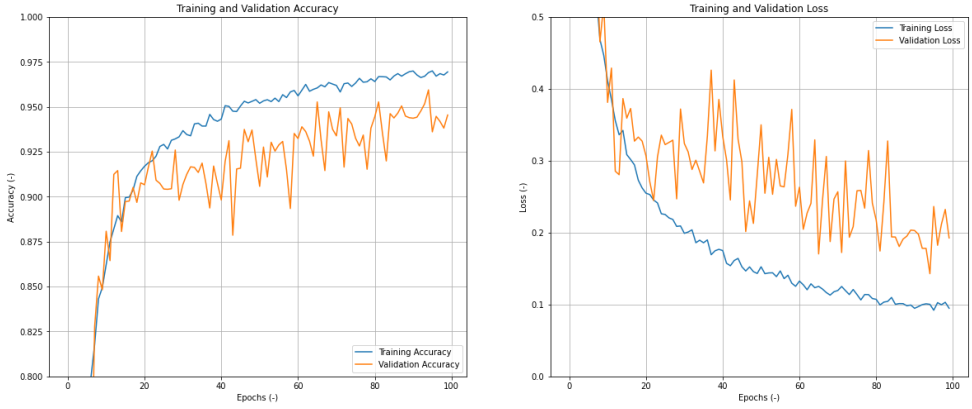
SVM performance



Confusion Matrix of SVM

6.1.3 CNN

We set epochs = 100 and it needs 30 minutes to train the model. The following is the model accuracy and loss curve in the training set and validation set.

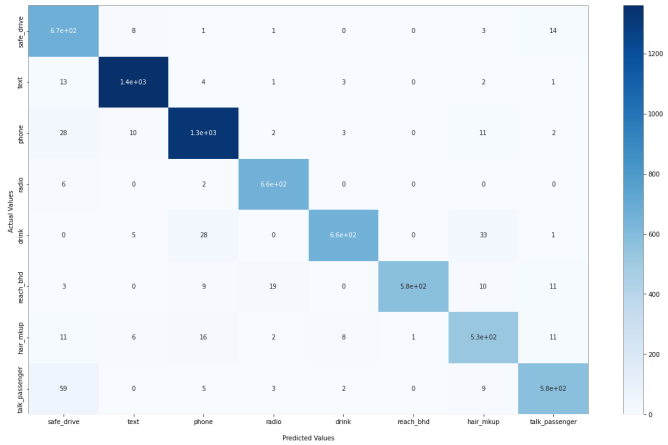


Accuracy and loss curve of CNN

The below shows the model performance in the test set and confusion matrix.

Accuracy	Precision
94.54%	84.82%

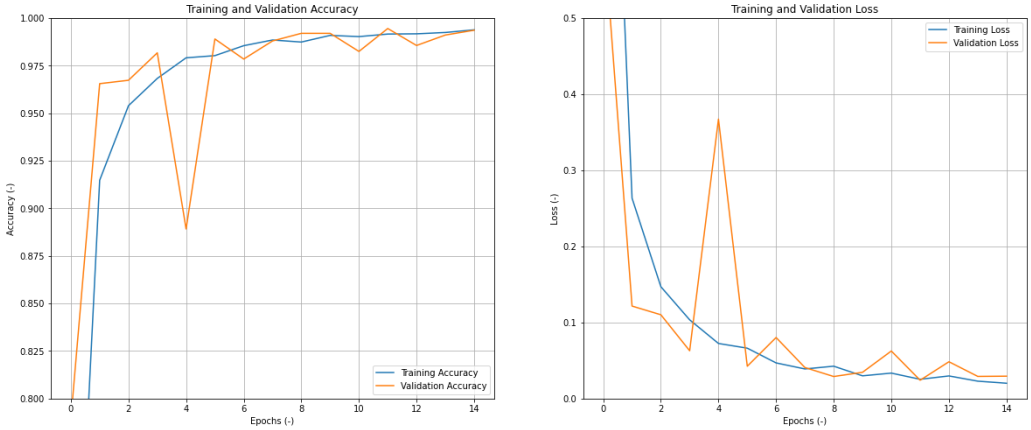
CNN performance



Confusion Matrix of CNN

6.1.4 Multilayer stacking CNNs

We set epochs = 15 and it only needs about 6 minutes to train the model. The following is the model accuracy and loss curve in the training set and validation set.

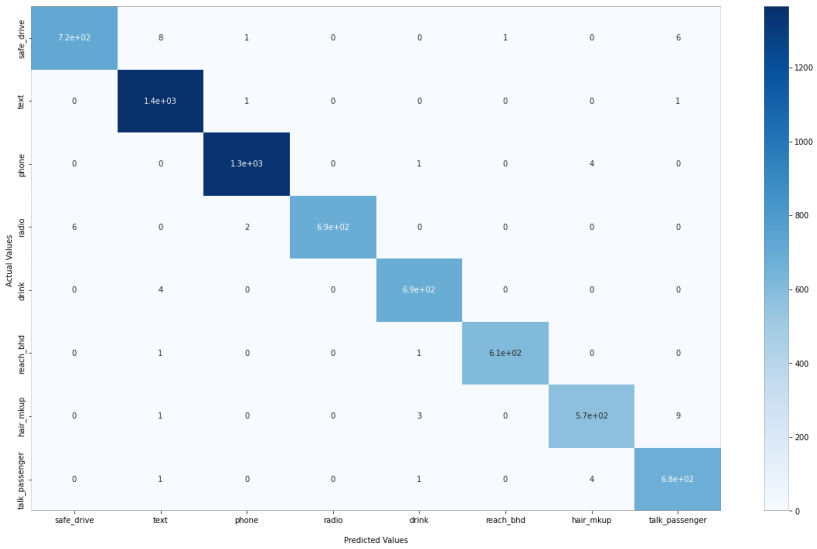


Accuracy and loss curve of Multilayer stacking CNNs

The below shows the model performance in the test set and confusion matrix.

Accuracy	**Precision**
99.36%	98.86%

Multilayer stacking CNNs performance



Confusion Matrix of Multilayer stacking CNNs

Recap that the reason why we use multilayer stacking CNNs is that we are worried that CNN is not complex enough and has no good performance, so we propose multilayer stacking CNNs, a more complex model. The experimental results also prove this point.

6.1.5 Model comparison

Lastly, we compare the performance and training speed of 4 models.

Model	Accuracy	Precision	Training Speed
Logistic Regression	94.20%	93.44%	Normal
SVM	99.20%	99.17%	Normal
CNN	94.54%	84.82%	Slow
Multilayer stacking CNNs	99.36%	98.86%	Normal

Model comparison result

In conclusion, SVM and multilayer stacking CNNs have excellent performance. Only about 1% probability that both will fail to detect distracting behavior. So, we can use these two models solve the task detection of distracted behaviors successfully.

6.2 Detection of drowsy driving

6.2.1 Eye-closing task

(1) Logistic Regression

	Precision	Recall	f1-score	Support
Yawn	0.95	0.93	0.94	223
No yawn	0.93	0.94	0.94	213
macro average	0.94	0.94	0.94	436

Logistic Regression performance in Eye-closing task

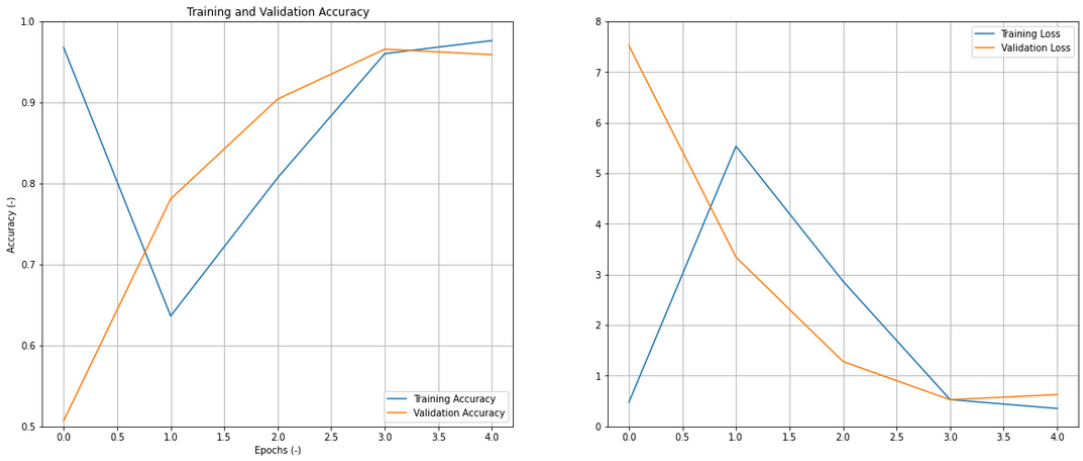
The accuracy of Logistic Regression in the test set is 95.18%.
(2) SVM

	Precision	Recall	f1-score	Support
Yawn	0.95	0.93	0.94	223
No yawn	0.93	0.94	0.94	213
macro average	0.94	0.94	0.94	436

SVM performance in Eye-closing task

The accuracy of SVM in the test set is 93.81%.

(3) MoibleNet
We set epochs = 5 and the convergent speed is fast. The following is the model accuracy and loss curve in the training set and validation set.



Accuracy and loss curve of MoibleNet

The accuracy of MoibleNet in the test set is 95.89%.

(4) Model comparison
Lastly, we can make a model comparison.

Model	Accuracy	Training Speed
Logistic Regression	95.18%	Quick
SVM	93.81%	Quick
MoibleNet	95.89%	Satisfied

Model comparison result in Eye-closing task

Since MoibleNet has best performance and its convergence speed is satisfied, we choose it as our final model to solve eye-closing task. And it will be used when we merge models in the detection of drowsy driving in part 6.3.

6.2.2 Yawning task
(1) Logistic Regression

	Precision	Recall	f1-score	Support
Yawn	0.88	0.82	0.85	74
No yawn	0.82	0.88	0.85	68
macro average	0.85	0.85	0.85	142

Logistic Regression performance in Yawning-task

The accuracy of Logistic Regression in the test set is 85.21%

(2) SVM

	Precision	Recall	f1-score	Support
Yawn	0.91	0.85	0.88	74
No yawn	0.85	0.91	0.88	68
macro average	0.88	0.88	0.88	142

SVM performance in Yawning-task

The accuracy of SVM in the test set is 88.73%.

(3) Ensemble Learning

Before proceeding to integration learning, we would like to discuss three models here, XGBoost, LightBGM and RandomForest.

Among the Boosting algorithms, we do not choose GBDT here, but XGBoost and LightBGM. XGBoost is based on GBDT, which has the advantage of higher accuracy through the second-order Taylor expansion of the loss function. It supports both linear regression and logistic regression. It also adds a regular term to the objective function to control the complexity of the model. The learning rate is enhanced by weakening the effect of leaf nodes after each iteration. The LightBGM algorithm is based on XGBoost with the addition of Histogram algorithm. The time complexity and space complexity of the algorithm are effectively reduced.

Among the Bagging algorithms, here we have chosen RandomForest algorithm. This algorithm can handle high-dimensional data very well. It has strong anti-interference ability and anti-overfitting ability, and at the same time can handle classification problems very well. We think it can fit well to predict our problem.

Finally, to balance the interaction between the three models, we set the learning weights as XGBoost : LightGBM : RandomForest = 2:3:3. The following table shows the accuracy of each model.

Modeling	Accuracy
XGBoost	88.03%
LightBGM	87.32%
RandomForest	85.92%
Ensemble learning	89.03%

Accuracy table in ensemble learning

(4) Model comparison

Model	Accuracy	Training Speed
Logistic Regression	85.21%	Quick
SVM	88.73%	Quick
Ensemble learning	89.03%	Satisfied

Model comparison result in Yawning-task

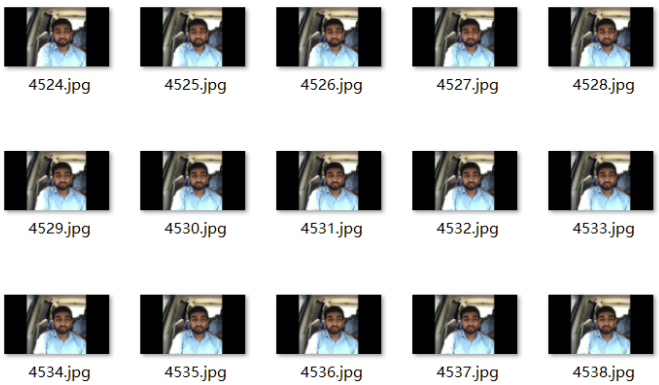
We finally chose to use ensemble learning model to fit the yawning data because this model has better resistance to overfitting and interference. Besides, it will be used when we merge models in the detection of drowsy driving in part 6.3.

6.3 Merge model in the detection of drowsy driving

Recap the rules for judging drowsy driving mentioned in part 5.5 is that in in 5 frame pictures, if the driver has more than 3 frames in a row where his eyes are closed, and he is yawning, we consider this driver to be driving in a drowsy state. After the above experiment, lastly, we use MobileNet model to judge whether the driver close his eye and ensemble learning model to judge whether the driver yawns.

To verify the effect of our merged model, and to be closer to the actual scene. We select a 30s driving video as the test set of drowsy driving detection.

We take 0.2s as a frame to intercept the video into pictures like the following

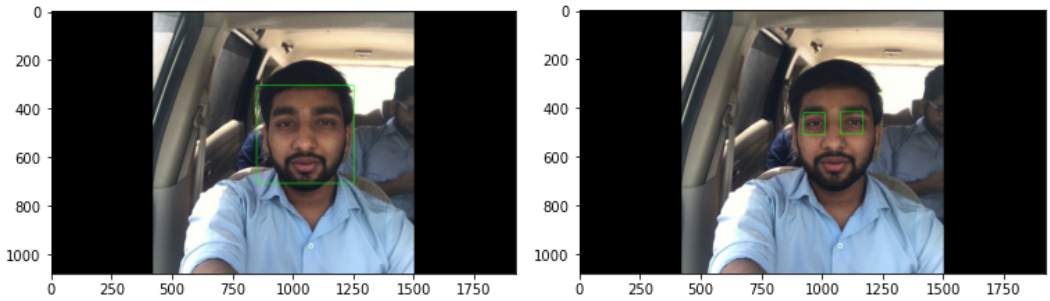


Extracted frames from the video

Before training, we need to implement face recognition and eye recognition, which are similar. In the case of face recognition, for example, the face images are first considered as the original data set. It is processed and downscaled using PCA methods to get the "principal components", the feature faces. Each face can then be represented as a combination of feature faces. The key idea of this approach is to consider that the same class of things must have the same characteristics (principal components). By finding out the characteristics of the same target (face image), it can be used to distinguish different things. The essence is a classification problem to distinguish different faces from each other.

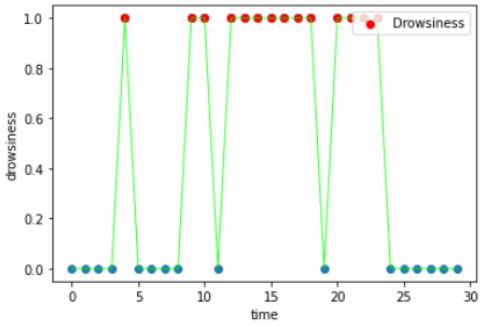
Here we use the CascadeClassifier function in OpenCV and the two "haarcascade_frontalface_default.xml" and " haarcascade_eye.xml" recognition files to form a face recognizer and an eye recognizer. These two files can effectively recognize the face and eyes in a picture. The CascadeClassifier function is implemented based on many positive and negative examples. After training, it can be applied to use the same size region of interest as in the training process. The detection process generates a

search box that moves around the image, passing over each location and detecting it with the classifier. The size of the search box is changed after each retrieval pass.
We use face recognizer and eye recognizer for each image separately and input the data obtained after recognition into yawn detection model and eye-closing detection model to predict respectively.



Use face recognizer and eye recognizer to the images

Finally, we can get a line graph of whether this driver is drowsy within 30s or not. In the figure, the x-axis is time. y-axis is whether the driver is drowsy, 0 is safety driving, 1 is drowsy driving.



Drowsy driving or not graph

And after comparing the video images frame by frame manually, we can get the prediction result of 86.6% for accuracy and 91.7% for recall according to this method.

7. Conclusion and Future Work

7.1 Conclusion

Our project focus on the detection of distracted driving behavior and drowsy driving. For each problem, we use a variety of models and compare them to select the best model. For the detection of distracted behaviors, SVM and multilayer stacking CNNs can get the best fitting results with 99% accuracy and 99% precision. For the detection of drowsy driving, we use the merged model by MobileNet which has best performance in the eye-closing recognition and ensemble learning model composed of XGBoost, LightGBM, RandomForest which has best performance in the yawn recognition. When using our model to test the actual driving video, it has 86.6% accuracy and 91.7% recall.

In general, our model can correctly detect distracted driving behavior and fatigue

driving most of the time. Therefore, the driver can be reminded when the above situations occur, so as to reduce the occurrence of accidents.

7.2 Future work

- More datasets

We found that the number of data sets marked is still small. In the future research, we can collect some distracted driving behaviors and drowsy driving and mark them by ourselves.

- Advanced algorithm for face and eye frame

In this project, when we frame the face and eyes, we only focus on the picture of the face. In the next research, we can improve the algorithm so that the face and eyes in the side face image are also well framed.

- One model, two problems

When solving the two problems of distracted driving behavior and drowsy driving, we use different models. But in essence, they all belong to image recognition. In the next research, we will explore whether we can use one model to solve two problems at the same time.

Reference

[1] C. Huang, X. Wang, J. Cao, S. Wang and Y. Zhang, "HCF: A Hybrid CNN Framework for Behavior Detection of Distracted Drivers," in IEEE Access, vol. 8, pp. 109335-109349, 2020, doi: 10.1109/ACCESS.2020.3001159.

[2] Dalal, N. , & Triggs, B. . (2005). Histograms of Oriented Gradients for Human Detection. IEEE Computer Society Conference on Computer Vision & Pattern Recognition. IEEE

[3] Eoh HJ，Chung MK，Kim SH. Electroence-phalographic Study of Drowsiness in Simulated Driving with Sleep Deprivation[J].International Journal of Industrial Ergonomics，2005，35(4): 307 - 320.

[4] Gulbadan Sikander, Shahzad Anwar. Driver Fatigue Detection Systems： A Review[C]. IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS，2019，20（6）：2339-2352.

[5] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.

[6] Lamia Alam,Mohammed Moshiul Hoque.Real-Time Distraction Detection Based on Driver's Visual Features[A].2019 International Conference on Electrical, Computer and Communication Engineering. 7-9 February, 2019.

[7] S.Wang,Y.Zhang,C..Darvas,“Online Prediction of Driver Distraction Based on Brain Activity Patterns,”IEEE Trans. Intell. Transp. Syst, vol.16,136-150, Feb. 2015.

[8] T. Danisman, I. M. Bilasco, C. Djeraba and N. Ihaddadene, "Drowsy driver detection system using eye blink patterns," 2010 International Conference on Machine and Web Intelligence, 2010, pp. 230-233, doi: 10.1109/ICMWI.2010.5648121.

[9] Wang JJ, Wang YK, Zhang F, Zhang SW, Dai Y, Yu XD. Real-Time Detection for Eye Closure Feature of Fatigue Driving Based on CNN and SVM. Computer Systems and Applications, 2021, 30(6): 118-126(in Chinese).http://www.c-s-a.org.cn/1003-3254/7968.html.

[10] Y. Xing et al.,Identification and analysis of driver postures for invehicle driving activities and secondary tasks recognition[A]. IEEE Trans. Comput. Social Syst., vol. 5,pp. 95-108, March 2018.

[11] Z. Jie, M. Mahmoud, Q. Stafford-Fraser, P. Robinson, E. Dias and L. Skrypchuk, "Analysis of Yawning Behaviour in Spontaneous Expressions of Drowsy Drivers," 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), 2018, pp. 571-576, doi: 10.1109/FG.2018.00091.