DISTRACTED DRIVER DETECTION

Intelligent System
Group 6

Team members

Truong Minh Duy - 1652113

Nguyen Anh Hoang Phuc - 1752041

Tran Trung Quan - 1752044

Table of Contents

1. Introduction
2. Theoretical basis
3. Initial Experiments
4. System design
5. Evaluation
6. Conclusion

Overview

- Drivers are supposed to be focusing on driving by law
- However, many drivers doing something else when driving
- This distracted behaviors easily cause crash incidents

Alarming Statistic

1/5

car accidents in the US is caused by distracted driver

42500

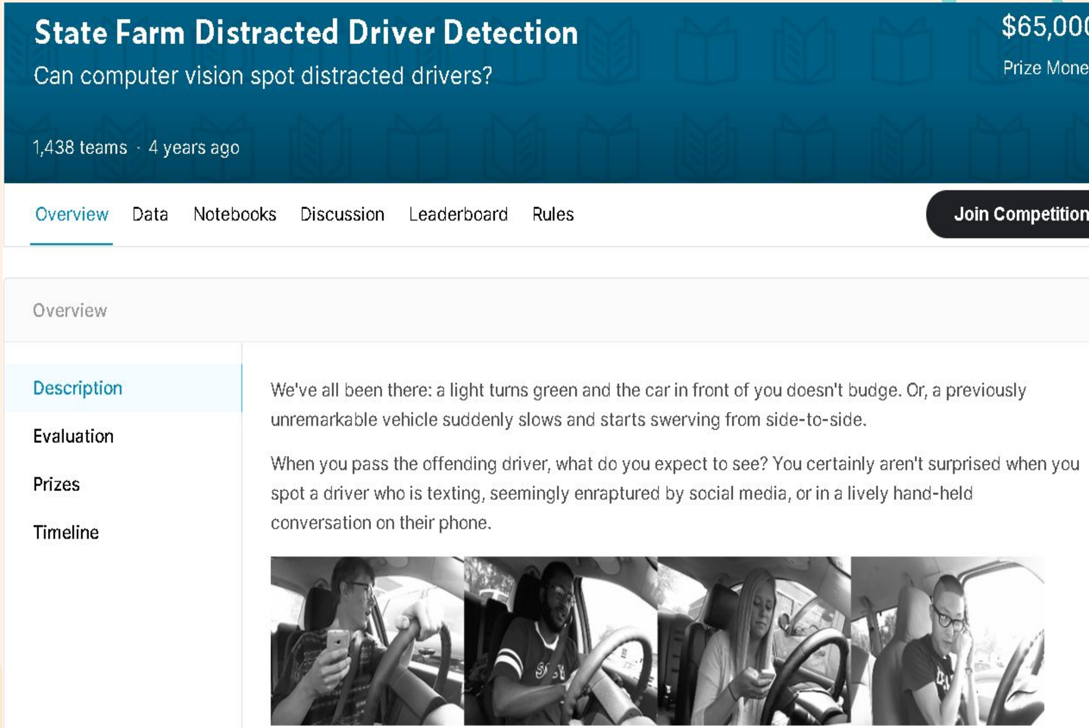
people injured and **3000** people is killed

87%

Young adults in VietNam no concentrate on driving

Overview

- Police cannot be on all roads at any one times to monitor drivers
- Take advantage of surveillance camera system to build a robust computer vision system
- In 2016, State Farm held a online Kaggle competition to solve this task



State Farm Distracted Driver Detection

Can computer vision spot distracted drivers?

1,438 teams · 4 years ago

$65,000

Prize Mone

We've all been there: a light turns green and the car in front of you doesn't budge. Or, a previously unremarkable vehicle suddenly slows and starts swerving from side-to-side.

When you pass the offending driver, what do you expect to see? You certainly aren't surprised when you spot a driver who is texting, seemingly enraptured by social media, or in a lively hand-held conversation on their phone.

Goals

- Survey and discover some machine/learning techniques, especially in the computer vision field.

- Apply machine learning techniques to build an application

- Build an application that can predict the distracting action by a/an input image/video

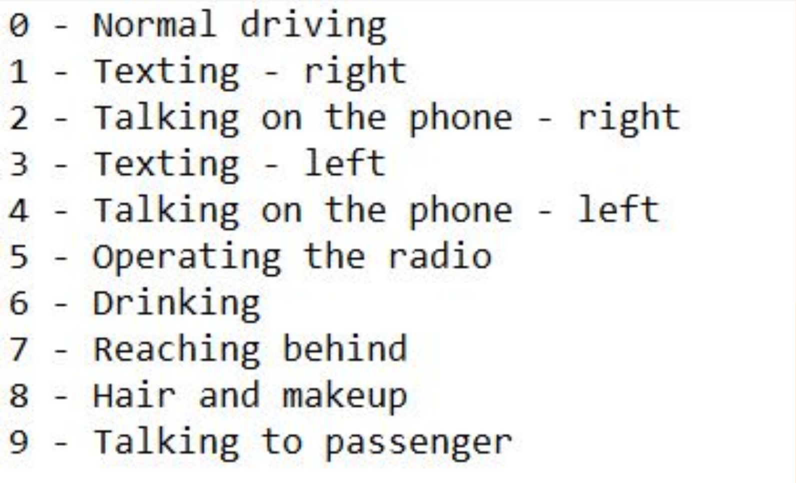
Features

- The application can receive either image or video as its input

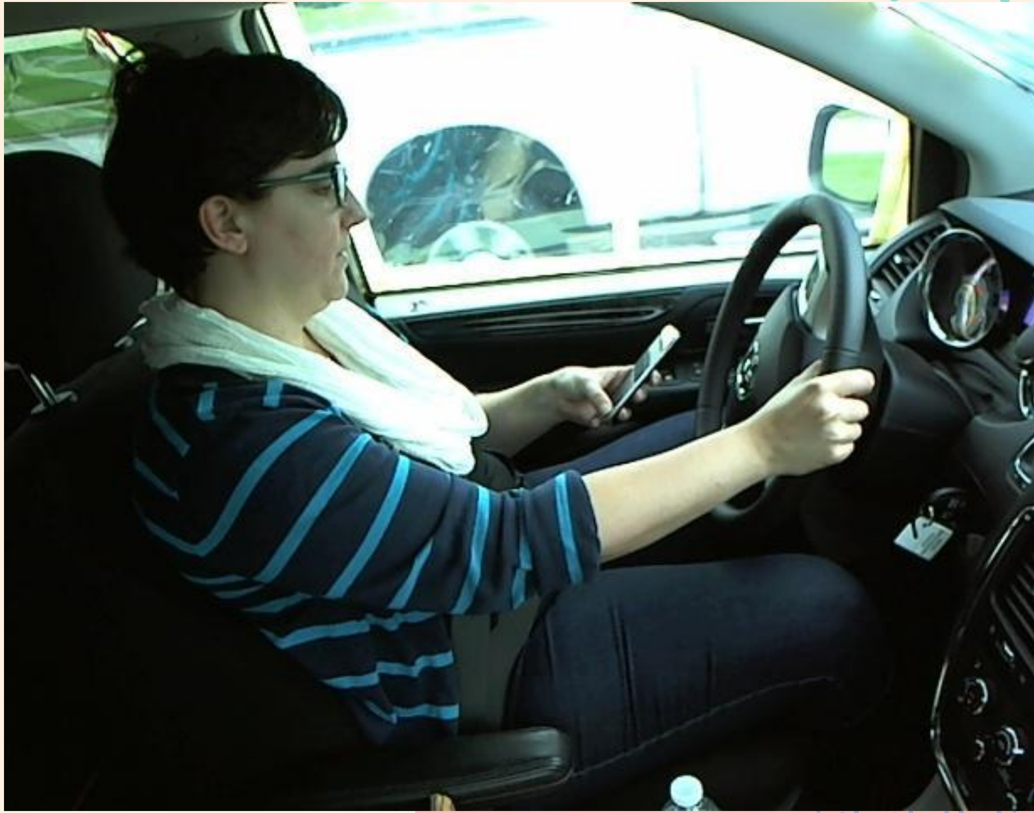
- The application can predict the distracting action (belong to 10 labels) with the accepted accuracy

- The time for prediction should be reasonable to apply to real-life.

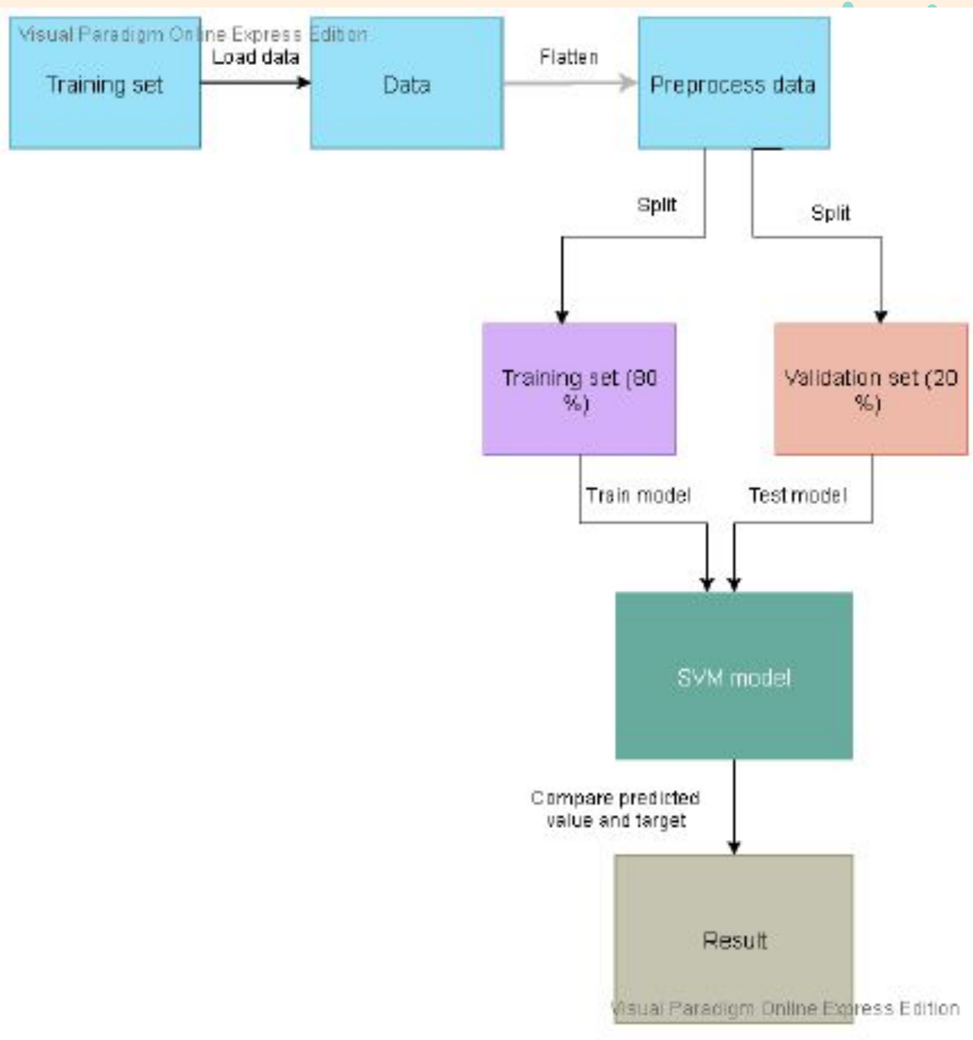
Data



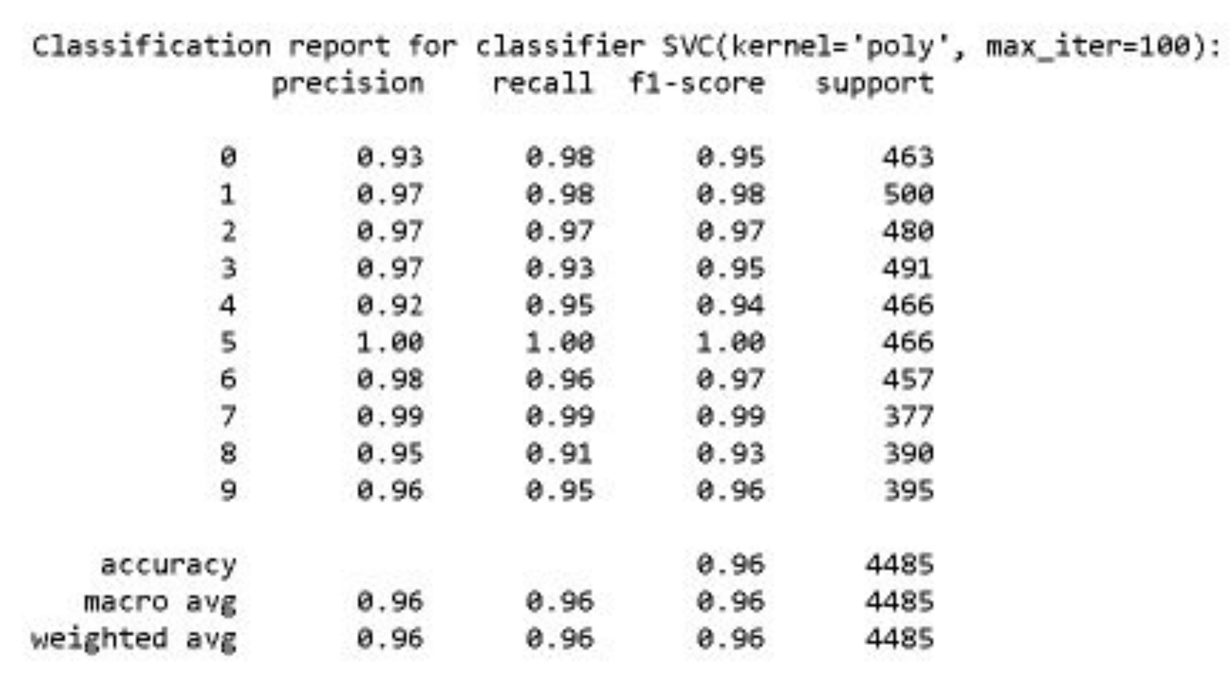
```
0 - Normal driving
1 - Texting - right
2 - Talking on the phone - right
3 - Texting - left
4 - Talking on the phone - left
5 - Operating the radio
6 - Drinking
7 - Reaching behind
8 - Hair and makeup
9 - Talking to passenger
```



Initial experiment

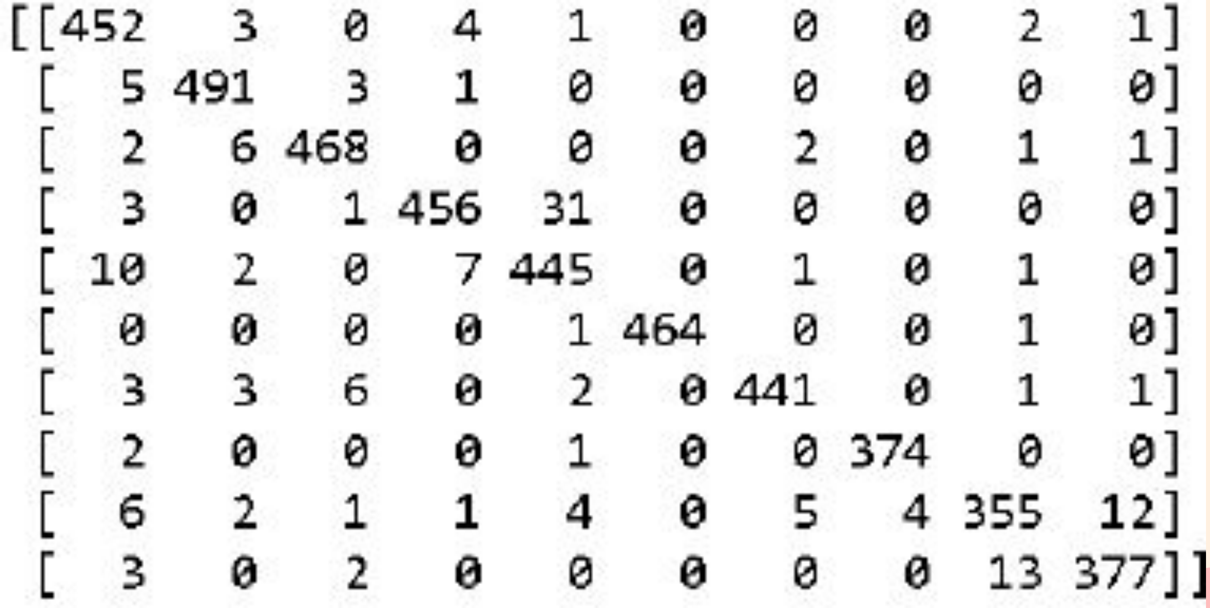


Results on validation set



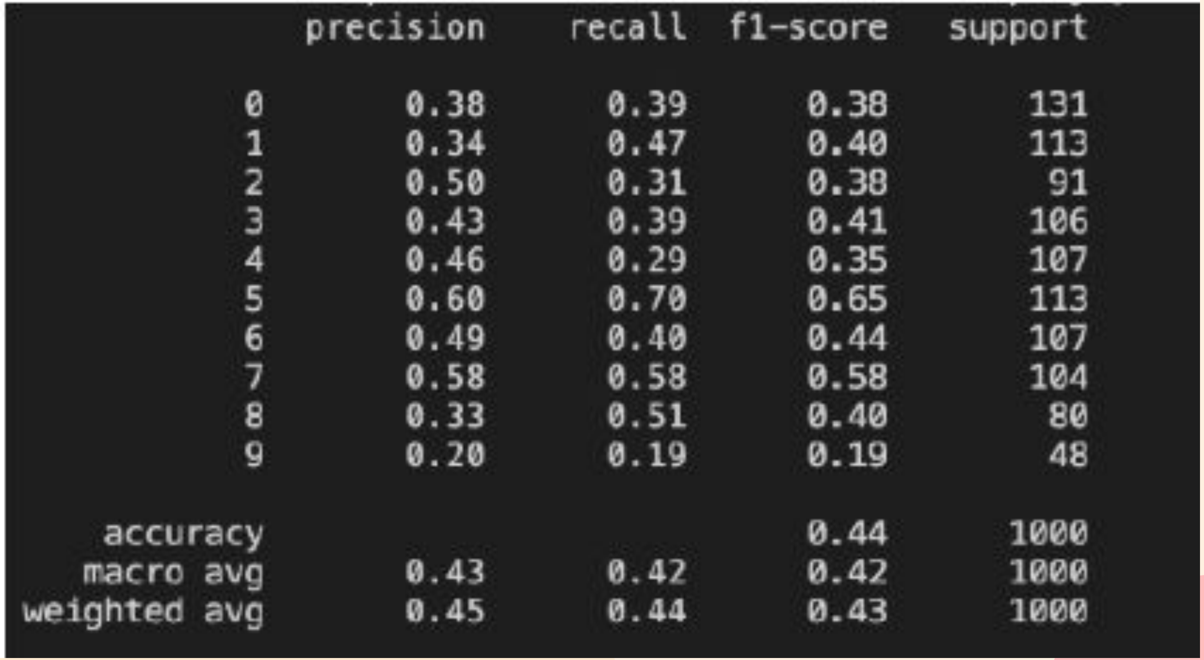
```
Classification report for classifier SVC(kernel='poly', max_iter=100):
              precision    recall  f1-score   support

           0       0.93      0.98      0.95       463
           1       0.97      0.98      0.98       500
           2       0.97      0.97      0.97       480
           3       0.97      0.93      0.95       491
           4       0.92      0.95      0.94       466
           5       1.00      1.00      1.00       466
           6       0.98      0.96      0.97       457
           7       0.99      0.99      0.99       377
           8       0.95      0.91      0.93       390
           9       0.96      0.95      0.96       395

    accuracy                           0.96      4485
   macro avg       0.96      0.96      0.96      4485
weighted avg       0.96      0.96      0.96      4485
```

Results on validation set



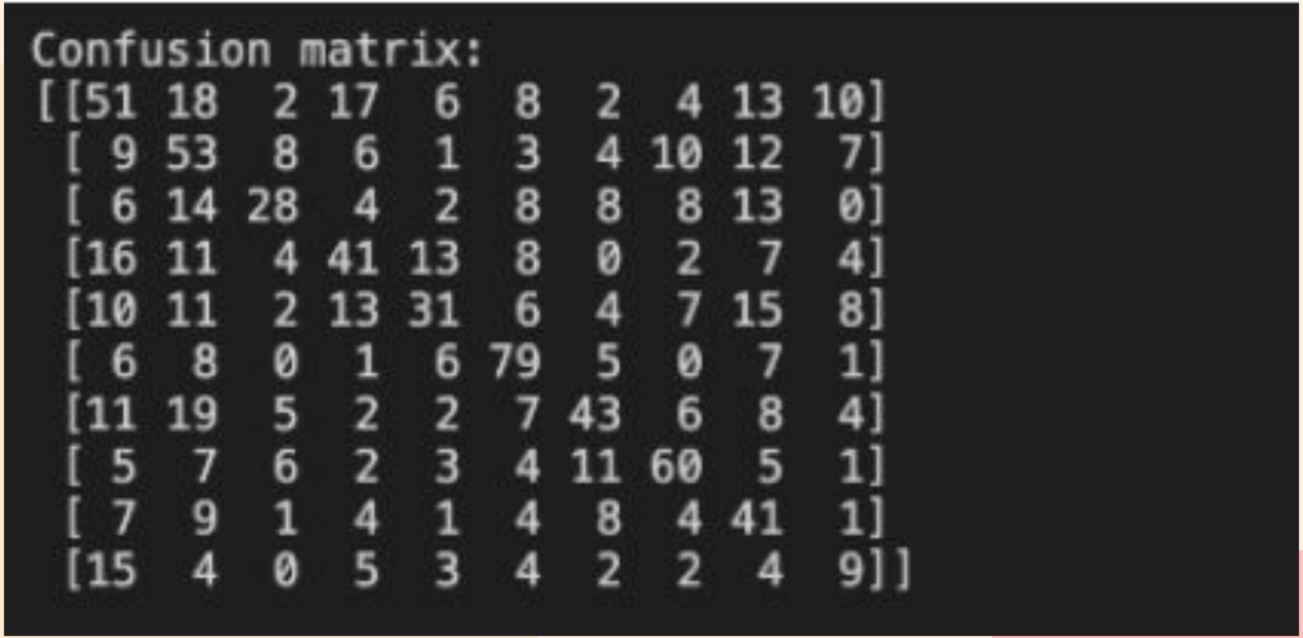
```
[[452   3   0   4   1   0   0   0   2   1]
 [  5 491   3   1   0   0   0   0   0   0]
 [  2   6 468   0   0   0   2   0   1   1]
 [  3   0   1 456  31   0   0   0   0   0]
 [ 10   2   0   7 445   0   1   0   1   0]
 [  0   0   0   0   1 464   0   0   1   0]
 [  3   3   6   0   2   0 441   0   1   1]
 [  2   0   0   0   1   0   0 374   0   0]
 [  6   2   1   1   4   0   5   4 355  12]
 [  3   0   2   0   0   0   0   0  13 377]]
```

Results on test set



```
              precision    recall  f1-score   support

           0       0.38      0.39      0.38       131
           1       0.34      0.47      0.40       113
           2       0.50      0.31      0.38        91
           3       0.43      0.39      0.41       106
           4       0.46      0.29      0.35       107
           5       0.60      0.70      0.65       113
           6       0.49      0.40      0.44       107
           7       0.58      0.58      0.58       104
           8       0.33      0.51      0.40        80
           9       0.20      0.19      0.19        48

    accuracy                           0.44      1000
   macro avg       0.43      0.42      0.42      1000
weighted avg       0.45      0.44      0.43      1000
```

Results on test set



```
Confusion matrix:
[[51 18  2 17  6  8  2  4 13 10]
 [ 9 53  8  6  1  3  4 10 12  7]
 [ 6 14 28  4  2  8  8  8 13  0]
 [16 11  4 41 13  8  0  2  7  4]
 [10 11  2 13 31  6  4  7 15  8]
 [ 6  8  0  1  6 79  5  0  7  1]
 [11 19  5  2  2  7 43  6  8  4]
 [ 5  7  6  2  3  4 11 60  5  1]
 [ 7  9  1  4  1  4  8  4 41  1]
 [15  4  0  5  3  4  2  2  4  9]]
```

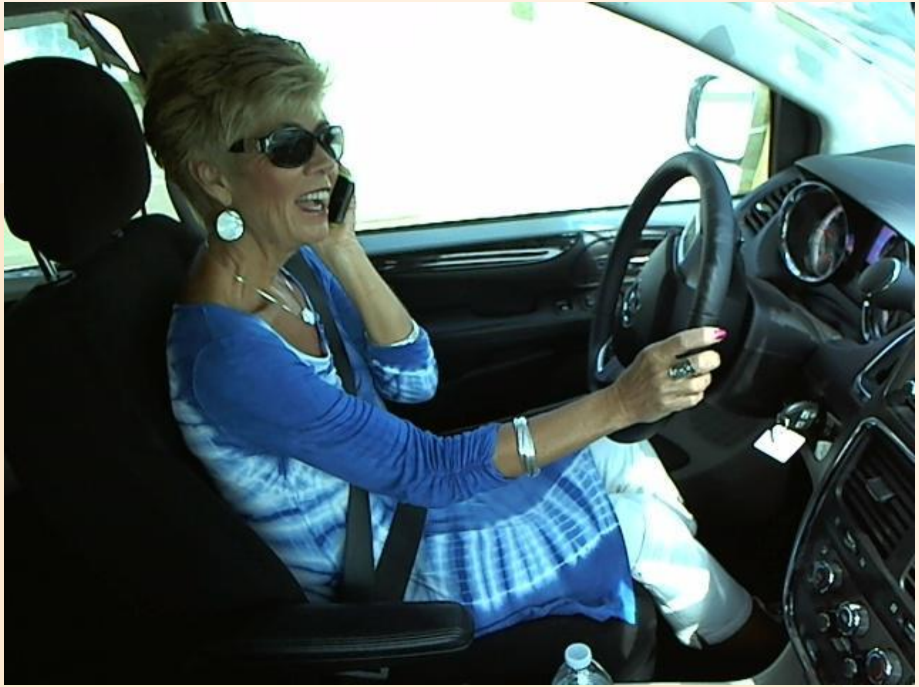
Our problems

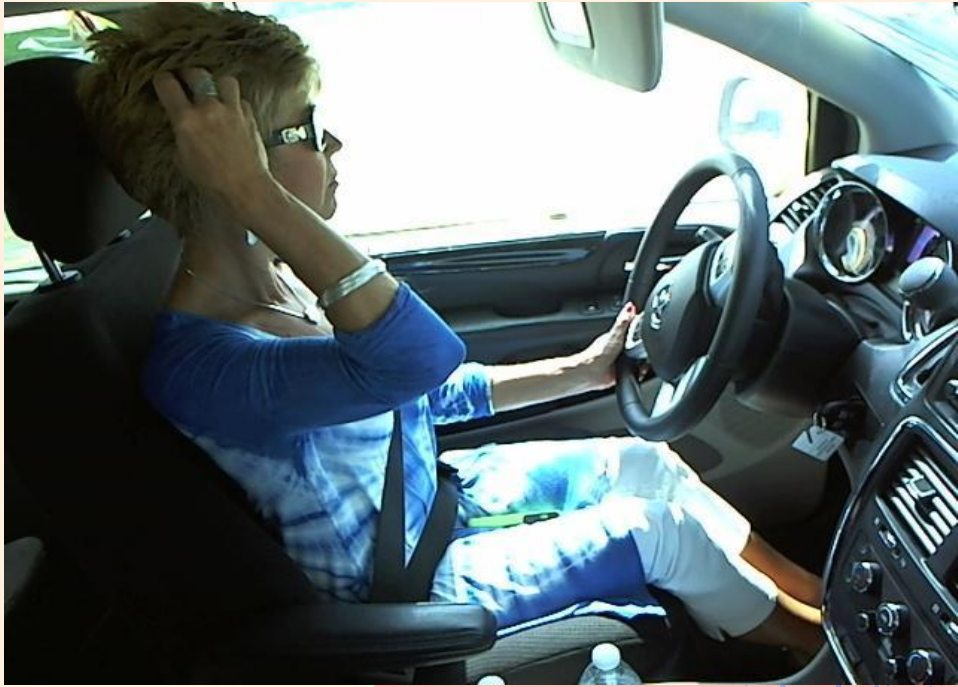
- Flatten data erase some information

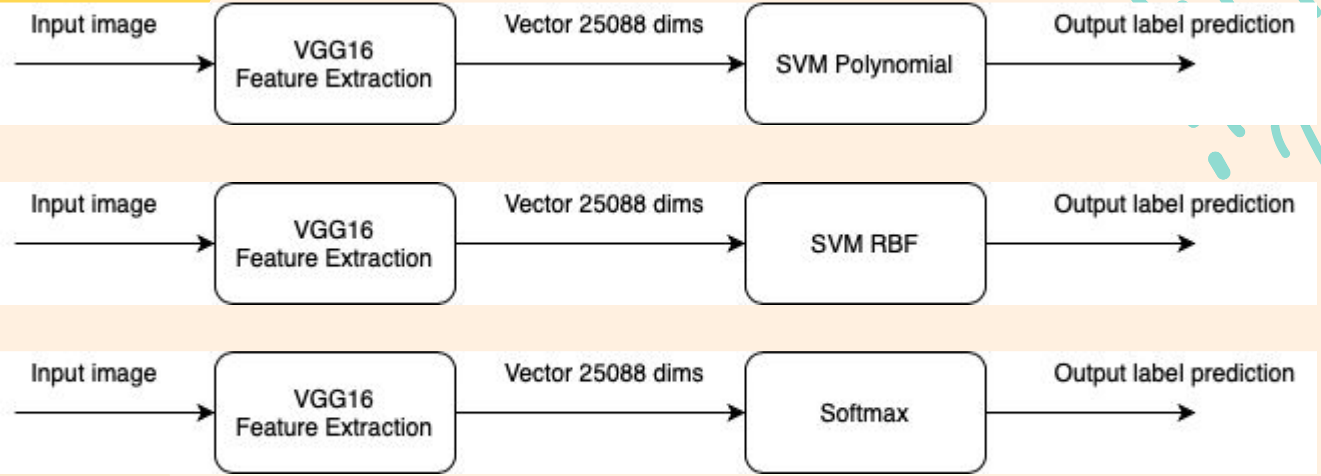
- Each driver can appear either on the test set or training set multiple times.

- Test set is larger than training set (> 4 time) -> easy to overfit

Sample data







Input image → VGG16 Feature Extraction → Vector 25088 dims → SVM Polynomial → Output label prediction

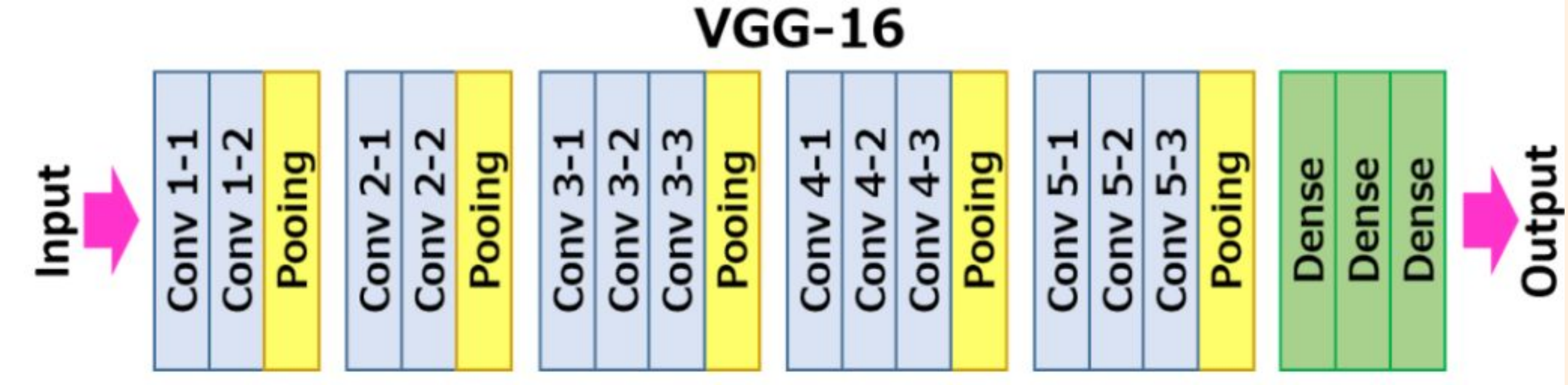
Input image → VGG16 Feature Extraction → Vector 25088 dims → SVM RBF → Output label prediction

Input image → VGG16 Feature Extraction → Vector 25088 dims → Softmax → Output label prediction

Our Proposal

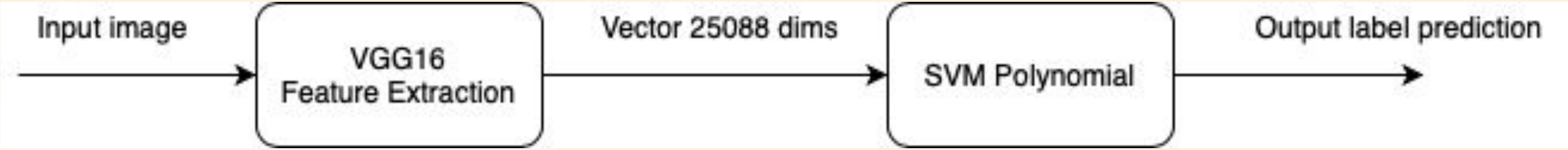
Feature Extraction by VGG16

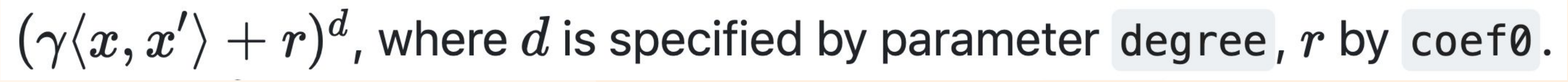
VGG16 Architecture



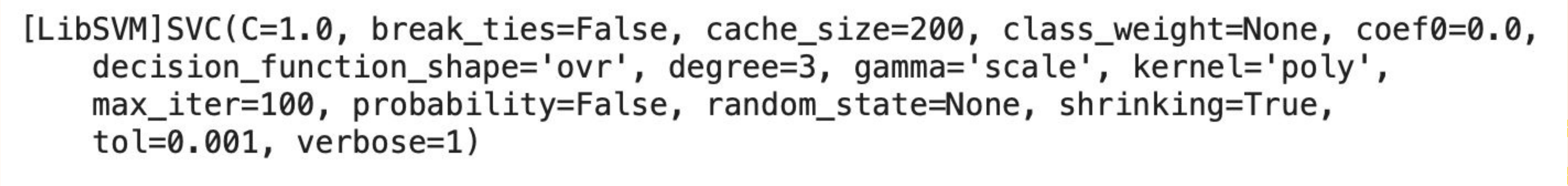
Input image: (224, 224, 3)
Output: Vector 25088-d
Params: 14,714,688

Classify by SVM Polynomial kernel



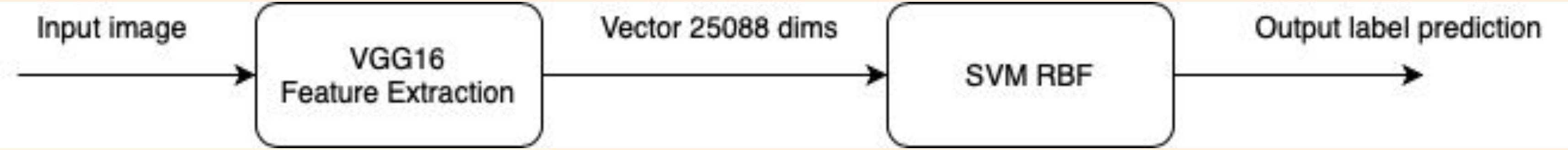


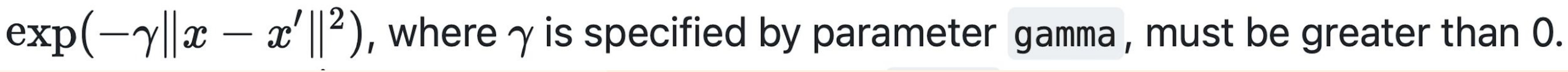
$(\gamma\langle x, x'\rangle + r)^d$, where d is specified by parameter `degree`, r by `coef0`.



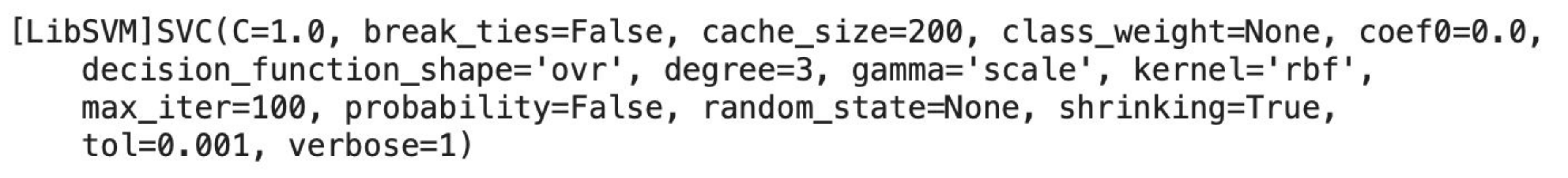
```
[LibSVM]SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='poly',
    max_iter=100, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=1)
```

Classify by SVM RBF kernel



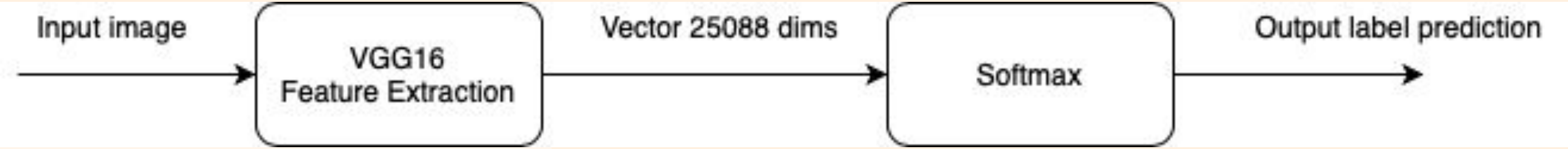


$\exp(-\gamma\|x - x'\|^2)$, where γ is specified by parameter `gamma`, must be greater than 0.



```
[LibSVM]SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=100, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=1)
```

Classify by Softmax



Input image

VGG16
Feature Extraction

Vector 25088 dims

Softmax

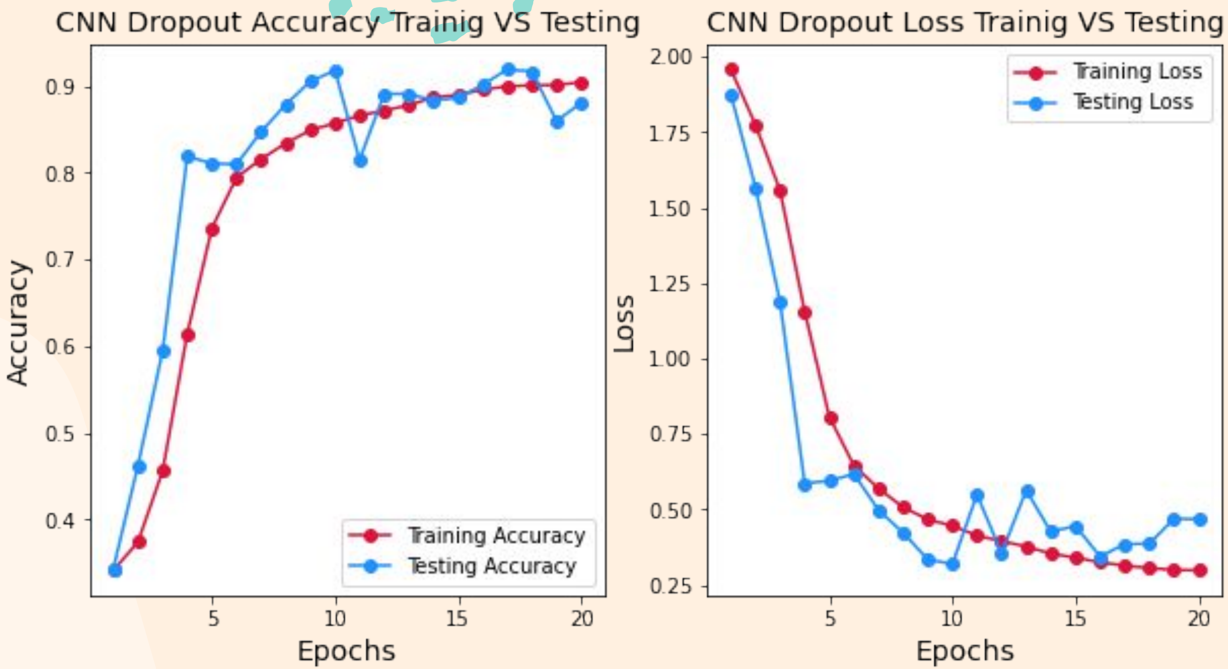
Output label prediction

$$\begin{bmatrix}1.2\\0.9\\0.4\end{bmatrix} \rightarrow \boxed{\text{Softmax}} \rightarrow \begin{bmatrix}0.46\\0.34\\0.20\end{bmatrix}$$

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ for } i = 1, \dots, K \text{ and } \mathbf{z} = (z_1, \dots, z_K) \in \mathbb{R}^K$$

Evaluation

Train VGG16+Softmax



CNN Dropout Accuracy Trainig VS Testing

CNN Dropout Loss Trainig VS Testing

Model	Training accuracy	Training loss	Validate accuracy	Validate loss
VGG16+softmax	0.9046	0.2982	0.8806	0.4684

Evaluation

Test set

Model	Metric			
	Accuracy	Precision	Recall	F1-score
VGG16+SVM Polynomial	0.54	0.77	0.54	0.57
VGG16+SVM RBF	0.91	0.78	0.76	0.75
VGG16+Soft max	0.89	0.73	0.73	0.72

Demonstration Application

Programming Language

QML

Application User Interface

Python

Application Logic

Framework and Library

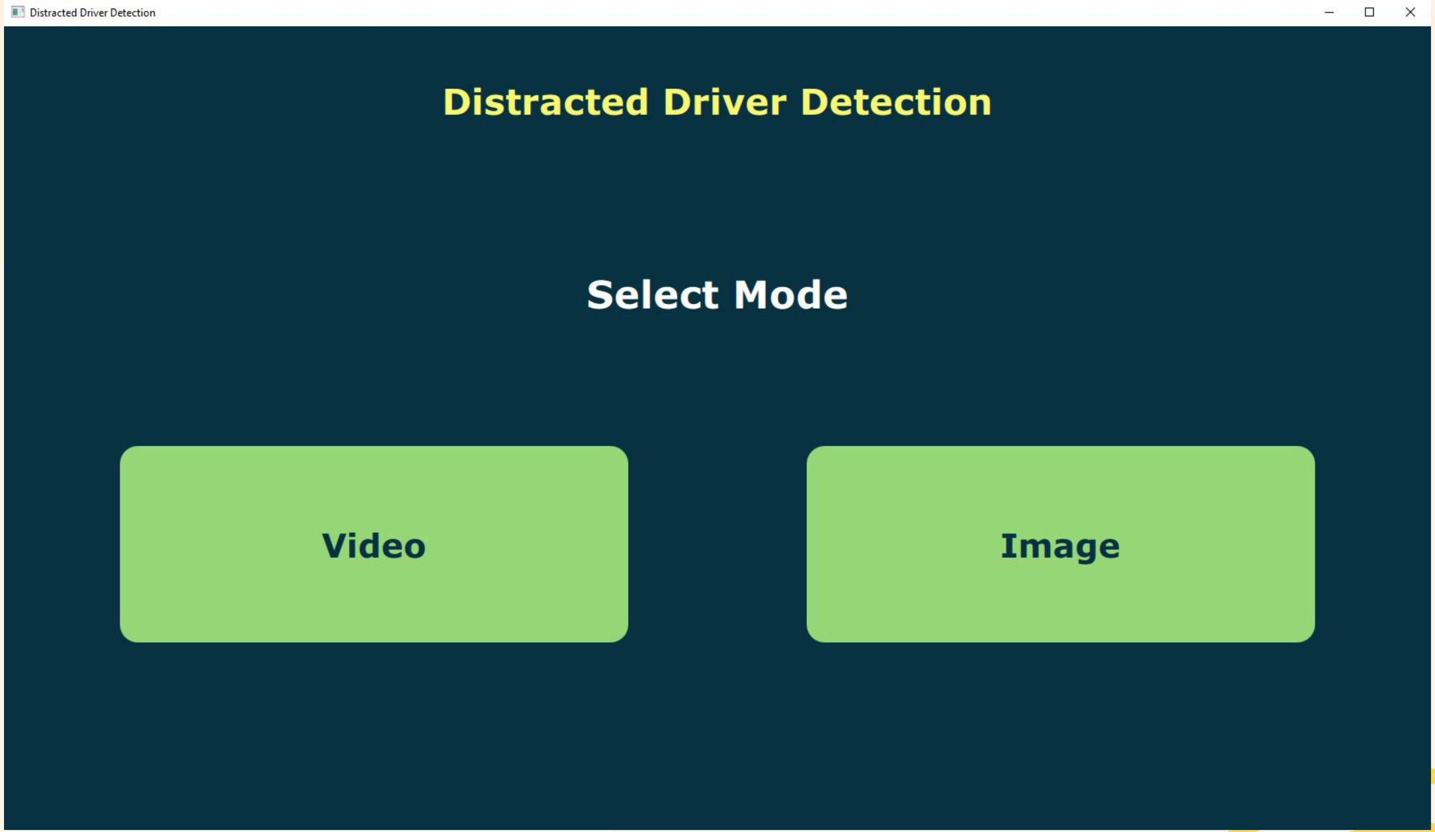
Qt Quick

PyQt5

OpenCV

TensorFlow

Demonstration

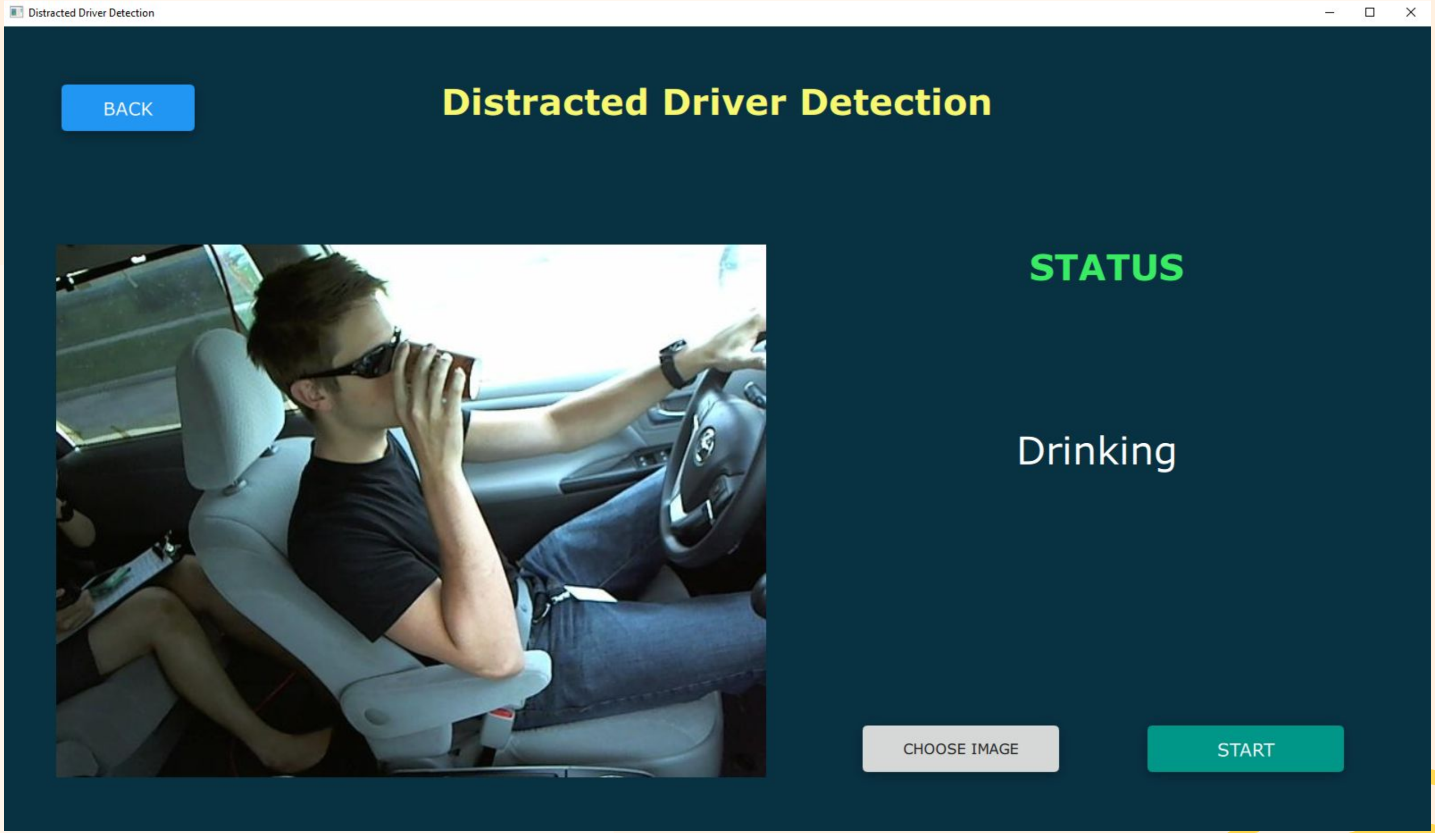


Distracted Driver Detection

Distracted Driver Detection

Select Mode

Demonstration



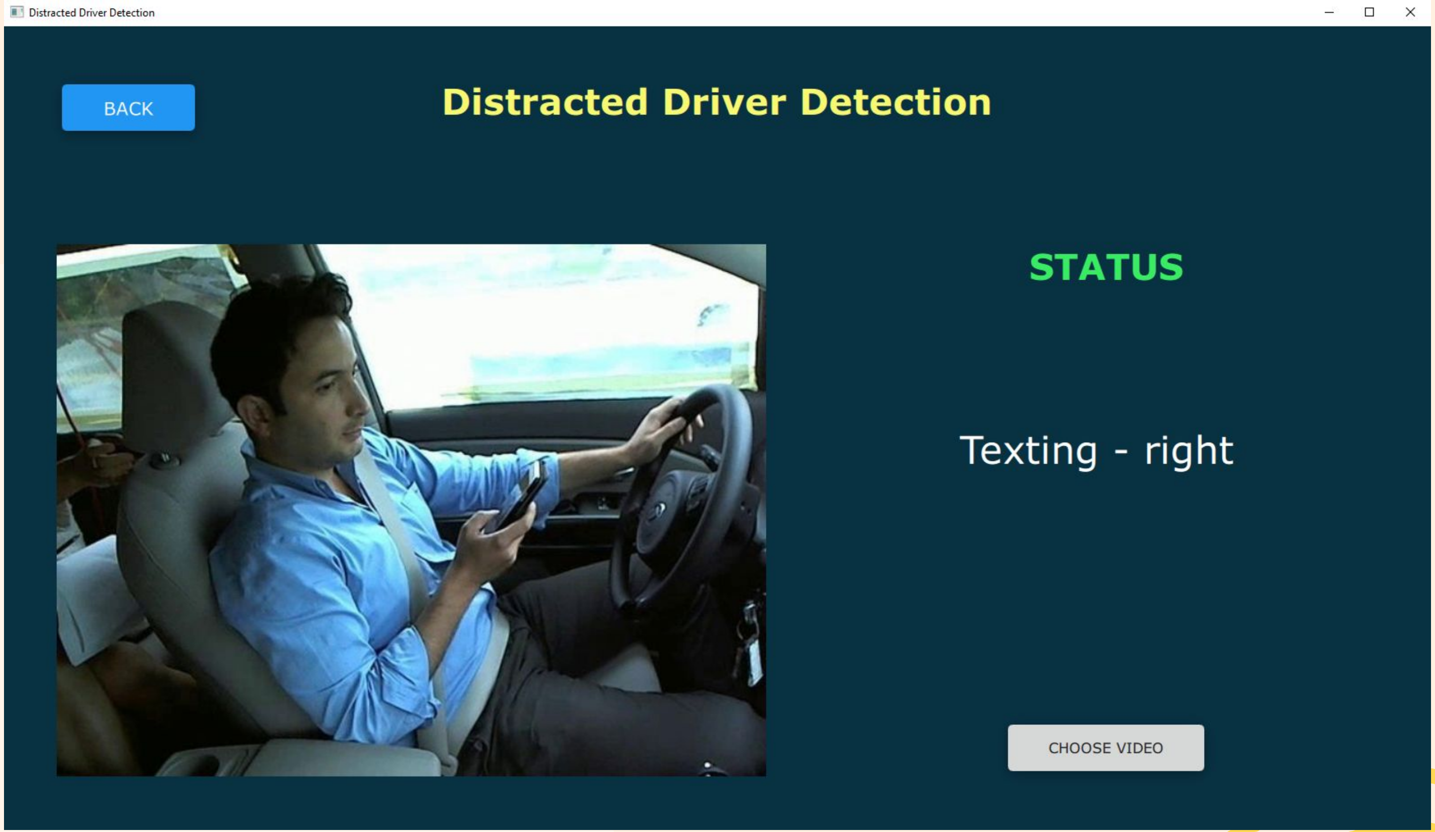
Distracted Driver Detection

Distracted Driver Detection

STATUS

Drinking

Demonstration



Distracted Driver Detection

STATUS

Texting - right

Conclusion

- Not only the model but also the data is very important. The good analysis of data helps to choose the better model.
- All features should not be considered equally. Some features are very important, some others can be removed.
- When applying AI into real applications, we should consider both time and accuracy.

Future Work

- State-of-the-art classification method such as: **ResNet** or **MobileNet**
- Improve accuracy using ensemble learning, so the final result is the major vote of many classification models.
- Using kNN for output weight. Work well in the case of video prediction, at some consecutive frames have similar images. So we can use pixel-loss to update their weight based on kNN.

Future Work

Application:

- Detection from real-time video.
- Notify when driver is distracted.
- Export to embedded device.

THANK YOU