Distracted Driving Detection

Load the Data

In [1]:
```
#dictionary for distraction category to numerical value
catLabels = {
    'c0': 'safe driving',
    'c1': 'texting - right',
    'c2': 'talking on the phone - right',
    'c3': 'texting - left',
    'c4': 'talking on the phone - left',
    'c5': 'operating the radio',
    'c6': 'drinking',
    'c7': 'reaching behind',
    'c8': 'hair and makeup',
    'c9': 'talking to passenger'
}

def getClass(value):
    index = 'c' + str(value)
    return catLabels[index]
```

In [2]:
```
from sklearn.datasets import load_files
from keras.utils import np_utils
import numpy as np
from glob import glob
import os
from sklearn.model_selection import train_test_split

import tensorflow as tf
hello = tf.constant('Hello, TensorFlow!')
sess = tf.Session()
print(sess.run(hello))

# import tensorflow as tf
# from keras import backend as K

# num_cores = 4
# GPU = 1
# CPU = 0

# if GPU:
#     num_GPU = 1
#     num_CPU = 1
# if CPU:
#     num_CPU = 1
#     num_GPU = 0

# config = tf.ConfigProto(intra_op_parallelism_threads=num_cores,\
```

```
#         inter_op_parallelism_threads=num_cores, allow_soft_placement=True,\
#         device_count = {'CPU' : num_CPU, 'GPU' : num_GPU})
# session = tf.Session(config=config)
# K.set_session(session)

def loadImages(path):
        data = load_files(path)
        files = data['filenames']
        targets = data['target']
        target_names = data['target_names']
        return files, targets, target_names

path = "images/train"
files,targets,target_names = loadImages(path)
predict_files = np.array(glob("images/test/*"))[1:10]
print('Number of Categories: ', len(target_names))
print('Categories: ', target_names)
print('Number of images by category: ')
for c in target_names:
    print(c + ':' + str(len( os.listdir(path+'/'+c))))
    # train_data = np.vstack((files, targets)).T
    # print(train_data.shape)

#Split the original training sets into training & validation sets
train_files, test_files, train_targets, test_targets = train_test_split(files,
 targets, test_size=0.20, random_state=40)

print(train_files.shape, test_files.shape, train_targets.shape, test_targets.s
hape)
print(len(test_files))
```

```
Using TensorFlow backend.
```

```
b'Hello, TensorFlow!'
Number of Categories:  10
Categories:  ['c0', 'c1', 'c2', 'c3', 'c4', 'c5', 'c6', 'c7', 'c8', 'c9']
Number of images by category:
c0:1900
c1:1900
c2:1900
c3:1900
c4:1900
c5:1900
c6:1900
c7:1900
c8:1900
c9:1900
(15200,) (3800,) (15200,) (3800,)
3800
```

Data Analysis

In [3]:
```
import cv2
import matplotlib.pyplot as plt
%matplotlib inline

def displayImage(sample_image):
    gray = cv2.cvtColor(sample_image, cv2.COLOR_BGR2GRAY)

    # convert BGR image to RGB for plotting
    cv_rgb = cv2.cvtColor(sample_image, cv2.COLOR_BGR2RGB)
    plt.imshow(cv_rgb)
    plt.show()

for i in range(1,5):
    sample_image = cv2.imread(train_files[i])
    print(train_targets[i])
    print(getClass(train_targets[i]))
    displayImage(sample_image)
```

8
hair and makeup

9
talking to passenger

0
safe driving

0
safe driving

In [4]:
```
#(nb_samples,rows,columns,channels)
#nb_samples - total number of images
# Resize image to 224x224
# Convert image to an array -> resized to a 4D tensor used by Keras CNN
# Tensor will be (1,224,224,3)

#Adopted from the Deep Learning Project
from keras.preprocessing import image
from tqdm import tqdm

def path_to_tensor(img_path):
    # loads RGB image as PIL.Image.Image type
    img = image.load_img(img_path, target_size=(224, 224))
    # convert PIL.Image.Image type to 3D tensor with shape (224, 224, 3)
    x = image.img_to_array(img)
    # convert 3D tensor to 4D tensor with shape (1, 224, 224, 3) and return 4D
 tensor
    return np.expand_dims(x, axis=0)

def paths_to_tensor(img_paths):
    print (img_paths)
    list_of_tensors = [path_to_tensor(img_path) for img_path in tqdm(img_paths
)]
    return np.vstack(list_of_tensors)
```

Pre-Process the Data

In [5]:
```
#Rescale the images

from PIL import ImageFile
ImageFile.LOAD_TRUNCATED_IMAGES = True

train_tensors = paths_to_tensor(train_files).astype('float32')/255
test_tensors = paths_to_tensor(test_files).astype('float32')/255
#predict_tensors = paths_to_tensor(predict_files).astype('float32')/255
```

```
['images/train\\c3\\img_24663.jpg' 'images/train\\c8\\img_98810.jpg'
 'images/train\\c9\\img_67390.jpg' ..., 'images/train\\c7\\img_31727.jpg'
 'images/train\\c7\\img_82756.jpg' 'images/train\\c5\\img_21995.jpg']

100%|████████████████████████████████████████████████████████████████
███| 15200/15200 [01:32<00:00, 164.72it/s]

['images/train\\c5\\img_68264.jpg' 'images/train\\c6\\img_69335.jpg'
 'images/train\\c2\\img_12280.jpg' ..., 'images/train\\c8\\img_6916.jpg'
 'images/train\\c6\\img_21610.jpg' 'images/train\\c5\\img_46343.jpg']

100%|████████████████████████████████████████████████████████████████
████| 3800/3800 [00:23<00:00, 160.98it/s]
```

Baseline Model Architecture

In [6]:
```
from keras.layers import Conv2D, MaxPooling2D, GlobalAveragePooling2D, B
atchNormalization, ActivityRegularization
from keras.layers import Dropout, Flatten, Dense
from keras.models import Sequential
from keras.utils import plot_model

model = Sequential()

model.add(Conv2D(filters=10, kernel_size=(4,4), input_shape=(224,224,3
)))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(4, 4), strides=None, padding='valid',
data_format=None))
model.add(Conv2D(filters=10, kernel_size=(4,4), input_shape=(224,224,3
)))
model.add(MaxPooling2D(pool_size=(4, 4), strides=None, padding='valid',
data_format=None))
model.add(Conv2D(filters=10, kernel_size=(4,4), input_shape=(224,224,3
)))
model.add(MaxPooling2D(pool_size=(4, 4), strides=None, padding='valid',
data_format=None))
model.add(GlobalAveragePooling2D())
model.add(Dense(units=10, activation='softmax'))
model.add(Dense(units=10, activation='softmax'))
model.add(Dense(units=10, activation='softmax'))
model.summary()


model.compile(optimizer='rmsprop', loss='categorical_crossentropy', metr
ics=['accuracy'])

# from IPython.display import SVG
# from keras.utils.vis_utils import model_to_dot
# plot_model(model, to_file='model.png')
# SVG(model_to_dot(model).create(prog='dot', format='svg'))
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_1 (Conv2D)            (None, 221, 221, 10)      490       
_________________________________________________________________
batch_normalization_1 (Batch (None, 221, 221, 10)      40        
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 55, 55, 10)        0         
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 52, 52, 10)        1610      
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 13, 13, 10)        0         
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 10, 10, 10)        1610      
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 2, 2, 10)          0         
_________________________________________________________________
global_average_pooling2d_1 ( (None, 10)                0         
_________________________________________________________________
dense_1 (Dense)              (None, 10)                110       
_________________________________________________________________
dense_2 (Dense)              (None, 10)                110       
_________________________________________________________________
dense_3 (Dense)              (None, 10)                110       
=================================================================
Total params: 4,080
Trainable params: 4,060
Non-trainable params: 20
_________________________________________________________________
```

Train the Model

In [7]:
```
from keras.callbacks import ModelCheckpoint
from keras.utils import np_utils

print("Train Targets", train_targets)
print ("Test Targets", test_targets)
train_targets_onehot = np_utils.to_categorical(np.array(train_targets),1
0)
test_targets_onehot = np_utils.to_categorical(np.array(test_targets),10)
print ("Train Targets One-hot encoded", train_targets_onehot)
print ("Test Targets One-hot encoded", test_targets_onehot)

print(train_targets_onehot.shape)
print(test_targets_onehot.shape)

checkpointer = ModelCheckpoint(filepath='C:/Users/pushkar/ML/machine-lea
rning/projects/capstone/saved_models/weights.best.from_scratch.hdf5',
                               verbose=1, save_best_only=True)

def train_model(_epochs):
    epochs = _epochs

    history = model.fit(train_tensors, train_targets_onehot, validation_
split=.20,
          epochs=epochs, batch_size=32, callbacks=[checkpointer], verbos
e=2)
    return history

history = train_model(200)
```

```
Train Targets [3 8 9 ..., 7 7 5]
Test Targets [5 6 2 ..., 8 6 5]
Train Targets One-hot encoded [[ 0.  0.  0. ...,  0.  0.  0.]
 [ 0.  0.  0. ...,  0.  1.  0.]
 [ 0.  0.  0. ...,  0.  0.  1.]
 ...,
 [ 0.  0.  0. ...,  1.  0.  0.]
 [ 0.  0.  0. ...,  1.  0.  0.]
 [ 0.  0.  0. ...,  0.  0.  0.]]
Test Targets One-hot encoded [[ 0.  0.  0. ...,  0.  0.  0.]
 [ 0.  0.  0. ...,  0.  0.  0.]
 [ 0.  0.  1. ...,  0.  0.  0.]
 ...,
 [ 0.  0.  0. ...,  0.  1.  0.]
 [ 0.  0.  0. ...,  0.  0.  0.]
 [ 0.  0.  0. ...,  0.  0.  0.]]
(15200, 10)
(3800, 10)
Train on 12160 samples, validate on 3040 samples
Epoch 1/100
 - 146s - loss: 2.2923 - acc: 0.1204 - val_loss: 2.2724 - val_acc: 0.195
7

Epoch 00001: val_loss improved from inf to 2.27243, saving model to C:/U
sers/pushkar/ML/machine-learning/projects/capstone/saved_models/weights.
best.from_scratch.hdf5
Epoch 2/100
 - 129s - loss: 2.2383 - acc: 0.1970 - val_loss: 2.1875 - val_acc: 0.191
8

Epoch 00002: val_loss improved from 2.27243 to 2.18755, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 3/100
 - 128s - loss: 2.1364 - acc: 0.1961 - val_loss: 2.0666 - val_acc: 0.199
0

Epoch 00003: val_loss improved from 2.18755 to 2.06658, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 4/100
 - 129s - loss: 2.0208 - acc: 0.2049 - val_loss: 1.9691 - val_acc: 0.212
5

Epoch 00004: val_loss improved from 2.06658 to 1.96906, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 5/100
 - 129s - loss: 1.9328 - acc: 0.2108 - val_loss: 1.9258 - val_acc: 0.211
2

Epoch 00005: val_loss improved from 1.96906 to 1.92575, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 6/100
 - 129s - loss: 1.8719 - acc: 0.2124 - val_loss: 1.8345 - val_acc: 0.209
5
```

```
Epoch 00006: val_loss improved from 1.92575 to 1.83453, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 7/100
 - 129s - loss: 1.8283 - acc: 0.2148 - val_loss: 1.8820 - val_acc: 0.208
2

Epoch 00007: val_loss did not improve
Epoch 8/100
 - 129s - loss: 1.8003 - acc: 0.2191 - val_loss: 1.7643 - val_acc: 0.212
8

Epoch 00008: val_loss improved from 1.83453 to 1.76427, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 9/100
 - 129s - loss: 1.7746 - acc: 0.2206 - val_loss: 1.8680 - val_acc: 0.220
4

Epoch 00009: val_loss did not improve
Epoch 10/100
 - 129s - loss: 1.7585 - acc: 0.2251 - val_loss: 1.7685 - val_acc: 0.220
7

Epoch 00010: val_loss did not improve
Epoch 11/100
 - 129s - loss: 1.7407 - acc: 0.2255 - val_loss: 1.7249 - val_acc: 0.232
6

Epoch 00011: val_loss improved from 1.76427 to 1.72493, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 12/100
 - 129s - loss: 1.7279 - acc: 0.2254 - val_loss: 1.8392 - val_acc: 0.212
5

Epoch 00012: val_loss did not improve
Epoch 13/100
 - 129s - loss: 1.7163 - acc: 0.2391 - val_loss: 1.7359 - val_acc: 0.229
3

Epoch 00013: val_loss did not improve
Epoch 14/100
 - 129s - loss: 1.7029 - acc: 0.2368 - val_loss: 1.7504 - val_acc: 0.237
2

Epoch 00014: val_loss did not improve
Epoch 15/100
 - 129s - loss: 1.6925 - acc: 0.2529 - val_loss: 1.7666 - val_acc: 0.224
3

Epoch 00015: val_loss did not improve
Epoch 16/100
 - 129s - loss: 1.6782 - acc: 0.2706 - val_loss: 1.6674 - val_acc: 0.267
4
```

```
Epoch 00016: val_loss improved from 1.72493 to 1.66736, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 17/100
 - 129s - loss: 1.6558 - acc: 0.2852 - val_loss: 1.8506 - val_acc: 0.213
5

Epoch 00017: val_loss did not improve
Epoch 18/100
 - 129s - loss: 1.6288 - acc: 0.3137 - val_loss: 1.6224 - val_acc: 0.325
7

Epoch 00018: val_loss improved from 1.66736 to 1.62237, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 19/100
 - 129s - loss: 1.5919 - acc: 0.3248 - val_loss: 1.6282 - val_acc: 0.318
7

Epoch 00019: val_loss did not improve
Epoch 20/100
 - 129s - loss: 1.5468 - acc: 0.3461 - val_loss: 1.5968 - val_acc: 0.334
5

Epoch 00020: val_loss improved from 1.62237 to 1.59677, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 21/100
 - 129s - loss: 1.5015 - acc: 0.3534 - val_loss: 1.4970 - val_acc: 0.348
7

Epoch 00021: val_loss improved from 1.59677 to 1.49700, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 22/100
 - 129s - loss: 1.4594 - acc: 0.3678 - val_loss: 1.4492 - val_acc: 0.364
8

Epoch 00022: val_loss improved from 1.49700 to 1.44925, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 23/100
 - 129s - loss: 1.4163 - acc: 0.3806 - val_loss: 1.4706 - val_acc: 0.324
7

Epoch 00023: val_loss did not improve
Epoch 24/100
 - 129s - loss: 1.3798 - acc: 0.3940 - val_loss: 1.3956 - val_acc: 0.394
7

Epoch 00024: val_loss improved from 1.44925 to 1.39560, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 25/100
 - 129s - loss: 1.3493 - acc: 0.4081 - val_loss: 1.3882 - val_acc: 0.384
9
```

```
Epoch 00025: val_loss improved from 1.39560 to 1.38820, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 26/100
 - 129s - loss: 1.3142 - acc: 0.4212 - val_loss: 1.3404 - val_acc: 0.417
8

Epoch 00026: val_loss improved from 1.38820 to 1.34042, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 27/100
 - 129s - loss: 1.2815 - acc: 0.4406 - val_loss: 1.2864 - val_acc: 0.428
6

Epoch 00027: val_loss improved from 1.34042 to 1.28637, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 28/100
 - 129s - loss: 1.2441 - acc: 0.4576 - val_loss: 1.3100 - val_acc: 0.412
8

Epoch 00028: val_loss did not improve
Epoch 29/100
 - 129s - loss: 1.2123 - acc: 0.4638 - val_loss: 1.3472 - val_acc: 0.408
6

Epoch 00029: val_loss did not improve
Epoch 30/100
 - 129s - loss: 1.1772 - acc: 0.4810 - val_loss: 1.2339 - val_acc: 0.454
9

Epoch 00030: val_loss improved from 1.28637 to 1.23388, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 31/100
 - 129s - loss: 1.1436 - acc: 0.4870 - val_loss: 1.2487 - val_acc: 0.466
1

Epoch 00031: val_loss did not improve
Epoch 32/100
 - 129s - loss: 1.1154 - acc: 0.5028 - val_loss: 1.1625 - val_acc: 0.492
1

Epoch 00032: val_loss improved from 1.23388 to 1.16250, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 33/100
 - 129s - loss: 1.0878 - acc: 0.5092 - val_loss: 1.0902 - val_acc: 0.512
8

Epoch 00033: val_loss improved from 1.16250 to 1.09023, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 34/100
 - 129s - loss: 1.0567 - acc: 0.5149 - val_loss: 1.1319 - val_acc: 0.496
7
```

```
Epoch 00034: val_loss did not improve
Epoch 35/100
 - 129s - loss: 1.0391 - acc: 0.5266 - val_loss: 1.1076 - val_acc: 0.508
9

Epoch 00035: val_loss did not improve
Epoch 36/100
 - 131s - loss: 1.0159 - acc: 0.5373 - val_loss: 1.1327 - val_acc: 0.501
0

Epoch 00036: val_loss did not improve
Epoch 37/100
 - 129s - loss: 0.9898 - acc: 0.5403 - val_loss: 1.0709 - val_acc: 0.519
7

Epoch 00037: val_loss improved from 1.09023 to 1.07093, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 38/100
 - 129s - loss: 0.9732 - acc: 0.5548 - val_loss: 1.1010 - val_acc: 0.507
6

Epoch 00038: val_loss did not improve
Epoch 39/100
 - 129s - loss: 0.9492 - acc: 0.5596 - val_loss: 1.0239 - val_acc: 0.540
8

Epoch 00039: val_loss improved from 1.07093 to 1.02391, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 40/100
 - 129s - loss: 0.9334 - acc: 0.5718 - val_loss: 1.0802 - val_acc: 0.529
3

Epoch 00040: val_loss did not improve
Epoch 41/100
 - 129s - loss: 0.9168 - acc: 0.5716 - val_loss: 1.0100 - val_acc: 0.543
4

Epoch 00041: val_loss improved from 1.02391 to 1.00996, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 42/100
 - 129s - loss: 0.8987 - acc: 0.5830 - val_loss: 1.1494 - val_acc: 0.513
8

Epoch 00042: val_loss did not improve
Epoch 43/100
 - 129s - loss: 0.8898 - acc: 0.5917 - val_loss: 1.0082 - val_acc: 0.554
6

Epoch 00043: val_loss improved from 1.00996 to 1.00821, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 44/100
 - 129s - loss: 0.8759 - acc: 0.5952 - val_loss: 0.9287 - val_acc: 0.577
0
```

```
Epoch 00044: val_loss improved from 1.00821 to 0.92870, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 45/100
 - 129s - loss: 0.8592 - acc: 0.6001 - val_loss: 1.0244 - val_acc: 0.553
9

Epoch 00045: val_loss did not improve
Epoch 46/100
 - 129s - loss: 0.8479 - acc: 0.6055 - val_loss: 0.9599 - val_acc: 0.573
7

Epoch 00046: val_loss did not improve
Epoch 47/100
 - 129s - loss: 0.8349 - acc: 0.6182 - val_loss: 0.9510 - val_acc: 0.577
0

Epoch 00047: val_loss did not improve
Epoch 48/100
 - 129s - loss: 0.8241 - acc: 0.6208 - val_loss: 0.9527 - val_acc: 0.589
5

Epoch 00048: val_loss did not improve
Epoch 49/100
 - 129s - loss: 0.8097 - acc: 0.6239 - val_loss: 0.9086 - val_acc: 0.591
4

Epoch 00049: val_loss improved from 0.92870 to 0.90859, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 50/100
 - 129s - loss: 0.8012 - acc: 0.6299 - val_loss: 1.1966 - val_acc: 0.519
4

Epoch 00050: val_loss did not improve
Epoch 51/100
 - 129s - loss: 0.7924 - acc: 0.6359 - val_loss: 1.0236 - val_acc: 0.551
6

Epoch 00051: val_loss did not improve
Epoch 52/100
 - 129s - loss: 0.7848 - acc: 0.6335 - val_loss: 0.9291 - val_acc: 0.591
4

Epoch 00052: val_loss did not improve
Epoch 53/100
 - 129s - loss: 0.7699 - acc: 0.6421 - val_loss: 0.8689 - val_acc: 0.620
1

Epoch 00053: val_loss improved from 0.90859 to 0.86889, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 54/100
 - 129s - loss: 0.7590 - acc: 0.6525 - val_loss: 0.9352 - val_acc: 0.604
6
```

```
Epoch 00054: val_loss did not improve
Epoch 55/100
 - 129s - loss: 0.7497 - acc: 0.6561 - val_loss: 0.8579 - val_acc: 0.613
2

Epoch 00055: val_loss improved from 0.86889 to 0.85788, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 56/100
 - 129s - loss: 0.7402 - acc: 0.6630 - val_loss: 0.8950 - val_acc: 0.605
3

Epoch 00056: val_loss did not improve
Epoch 57/100
 - 129s - loss: 0.7317 - acc: 0.6642 - val_loss: 0.8293 - val_acc: 0.634
5

Epoch 00057: val_loss improved from 0.85788 to 0.82928, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 58/100
 - 129s - loss: 0.7234 - acc: 0.6647 - val_loss: 0.8997 - val_acc: 0.622
7

Epoch 00058: val_loss did not improve
Epoch 59/100
 - 129s - loss: 0.7140 - acc: 0.6747 - val_loss: 0.8288 - val_acc: 0.626
3

Epoch 00059: val_loss improved from 0.82928 to 0.82885, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 60/100
 - 129s - loss: 0.7102 - acc: 0.6762 - val_loss: 0.8188 - val_acc: 0.636
5

Epoch 00060: val_loss improved from 0.82885 to 0.81883, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 61/100
 - 129s - loss: 0.6936 - acc: 0.6839 - val_loss: 0.8040 - val_acc: 0.638
5

Epoch 00061: val_loss improved from 0.81883 to 0.80401, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 62/100
 - 129s - loss: 0.6900 - acc: 0.6859 - val_loss: 0.8176 - val_acc: 0.643
1

Epoch 00062: val_loss did not improve
Epoch 63/100
 - 129s - loss: 0.6796 - acc: 0.6892 - val_loss: 0.8122 - val_acc: 0.641
8

Epoch 00063: val_loss did not improve
Epoch 64/100
```

```
 - 129s - loss: 0.6710 - acc: 0.6960 - val_loss: 0.8924 - val_acc: 0.637
5

Epoch 00064: val_loss did not improve
Epoch 65/100
 - 129s - loss: 0.6685 - acc: 0.6942 - val_loss: 0.7919 - val_acc: 0.652
0

Epoch 00065: val_loss improved from 0.80401 to 0.79187, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 66/100
 - 129s - loss: 0.6604 - acc: 0.7051 - val_loss: 0.8058 - val_acc: 0.658
6

Epoch 00066: val_loss did not improve
Epoch 67/100
 - 129s - loss: 0.6499 - acc: 0.7044 - val_loss: 0.7991 - val_acc: 0.653
9

Epoch 00067: val_loss did not improve
Epoch 68/100
 - 129s - loss: 0.6473 - acc: 0.7103 - val_loss: 0.8754 - val_acc: 0.649
0

Epoch 00068: val_loss did not improve
Epoch 69/100
 - 129s - loss: 0.6368 - acc: 0.7164 - val_loss: 0.7896 - val_acc: 0.666
4

Epoch 00069: val_loss improved from 0.79187 to 0.78962, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 70/100
 - 129s - loss: 0.6309 - acc: 0.7137 - val_loss: 0.9314 - val_acc: 0.601
3

Epoch 00070: val_loss did not improve
Epoch 71/100
 - 129s - loss: 0.6235 - acc: 0.7200 - val_loss: 0.7481 - val_acc: 0.678
3

Epoch 00071: val_loss improved from 0.78962 to 0.74812, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 72/100
 - 129s - loss: 0.6120 - acc: 0.7235 - val_loss: 0.7824 - val_acc: 0.656
9

Epoch 00072: val_loss did not improve
Epoch 73/100
 - 128s - loss: 0.6078 - acc: 0.7307 - val_loss: 0.7863 - val_acc: 0.671
1

Epoch 00073: val_loss did not improve
Epoch 74/100
 - 129s - loss: 0.6011 - acc: 0.7348 - val_loss: 0.7961 - val_acc: 0.667
```

```
8

Epoch 00074: val_loss did not improve
Epoch 75/100
 - 129s - loss: 0.5929 - acc: 0.7377 - val_loss: 0.7204 - val_acc: 0.694
4

Epoch 00075: val_loss improved from 0.74812 to 0.72041, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 76/100
 - 129s - loss: 0.5892 - acc: 0.7371 - val_loss: 0.7326 - val_acc: 0.703
9

Epoch 00076: val_loss did not improve
Epoch 77/100
 - 129s - loss: 0.5839 - acc: 0.7460 - val_loss: 0.7496 - val_acc: 0.684
9

Epoch 00077: val_loss did not improve
Epoch 78/100
 - 129s - loss: 0.5743 - acc: 0.7520 - val_loss: 0.7692 - val_acc: 0.682
6

Epoch 00078: val_loss did not improve
Epoch 79/100
 - 129s - loss: 0.5713 - acc: 0.7544 - val_loss: 0.7611 - val_acc: 0.677
0

Epoch 00079: val_loss did not improve
Epoch 80/100
 - 129s - loss: 0.5649 - acc: 0.7560 - val_loss: 0.6998 - val_acc: 0.706
6

Epoch 00080: val_loss improved from 0.72041 to 0.69985, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 81/100
 - 129s - loss: 0.5562 - acc: 0.7609 - val_loss: 0.7962 - val_acc: 0.687
2

Epoch 00081: val_loss did not improve
Epoch 82/100
 - 129s - loss: 0.5480 - acc: 0.7692 - val_loss: 0.7921 - val_acc: 0.694
1

Epoch 00082: val_loss did not improve
Epoch 83/100
 - 129s - loss: 0.5397 - acc: 0.7704 - val_loss: 0.7108 - val_acc: 0.719
7

Epoch 00083: val_loss did not improve
Epoch 84/100
 - 129s - loss: 0.5307 - acc: 0.7779 - val_loss: 0.7342 - val_acc: 0.700
3

Epoch 00084: val_loss did not improve
```

```
Epoch 85/100
 - 129s - loss: 0.5272 - acc: 0.7808 - val_loss: 0.6958 - val_acc: 0.729
9

Epoch 00085: val_loss improved from 0.69985 to 0.69578, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 86/100
 - 130s - loss: 0.5157 - acc: 0.7824 - val_loss: 0.7089 - val_acc: 0.704
3

Epoch 00086: val_loss did not improve
Epoch 87/100
 - 129s - loss: 0.5121 - acc: 0.7866 - val_loss: 0.7709 - val_acc: 0.702
6

Epoch 00087: val_loss did not improve
Epoch 88/100
 - 128s - loss: 0.5079 - acc: 0.7912 - val_loss: 0.7016 - val_acc: 0.733
9

Epoch 00088: val_loss did not improve
Epoch 89/100
 - 128s - loss: 0.4997 - acc: 0.7947 - val_loss: 0.6918 - val_acc: 0.741
4

Epoch 00089: val_loss improved from 0.69578 to 0.69183, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 90/100
 - 128s - loss: 0.4897 - acc: 0.7999 - val_loss: 0.7498 - val_acc: 0.718
4

Epoch 00090: val_loss did not improve
Epoch 91/100
 - 128s - loss: 0.4874 - acc: 0.8029 - val_loss: 0.6490 - val_acc: 0.748
4

Epoch 00091: val_loss improved from 0.69183 to 0.64897, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 92/100
 - 129s - loss: 0.4787 - acc: 0.8058 - val_loss: 0.6934 - val_acc: 0.745
4

Epoch 00092: val_loss did not improve
Epoch 93/100
 - 128s - loss: 0.4680 - acc: 0.8145 - val_loss: 0.7272 - val_acc: 0.737
2

Epoch 00093: val_loss did not improve
Epoch 94/100
 - 128s - loss: 0.4653 - acc: 0.8170 - val_loss: 0.6712 - val_acc: 0.729
3

Epoch 00094: val_loss did not improve
Epoch 95/100
```

```
 - 128s - loss: 0.4547 - acc: 0.8193 - val_loss: 0.7357 - val_acc: 0.721
4

Epoch 00095: val_loss did not improve
Epoch 96/100
 - 128s - loss: 0.4527 - acc: 0.8242 - val_loss: 0.6567 - val_acc: 0.757
9

Epoch 00096: val_loss did not improve
Epoch 97/100
 - 129s - loss: 0.4406 - acc: 0.8304 - val_loss: 0.6924 - val_acc: 0.751
6

Epoch 00097: val_loss did not improve
Epoch 98/100
 - 128s - loss: 0.4376 - acc: 0.8335 - val_loss: 0.6922 - val_acc: 0.752
0

Epoch 00098: val_loss did not improve
Epoch 99/100
 - 129s - loss: 0.4295 - acc: 0.8387 - val_loss: 0.6357 - val_acc: 0.779
6

Epoch 00099: val_loss improved from 0.64897 to 0.63572, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 100/100
 - 128s - loss: 0.4236 - acc: 0.8391 - val_loss: 0.6354 - val_acc: 0.772
0

Epoch 00100: val_loss improved from 0.63572 to 0.63542, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
```

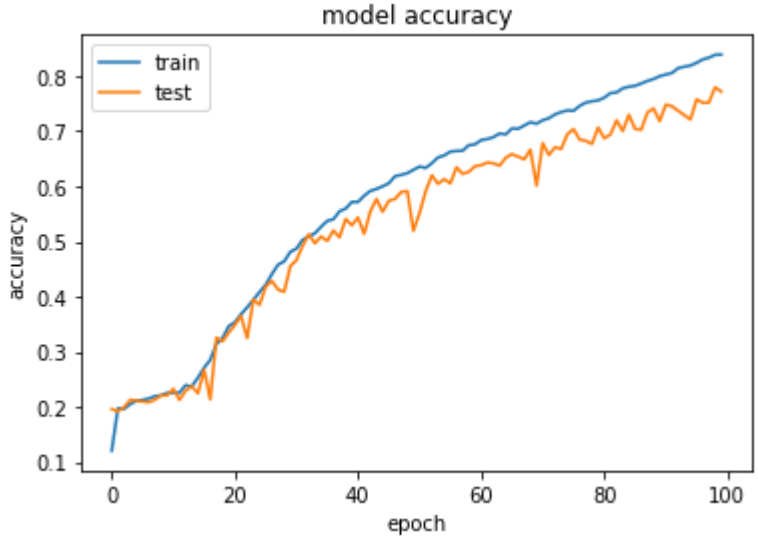
In [10]:
```
from keras.callbacks import ModelCheckpoint
from keras.utils import np_utils

model.load_weights('C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weights.best.from_scratch.hdf5')

history = train_model(250)
```

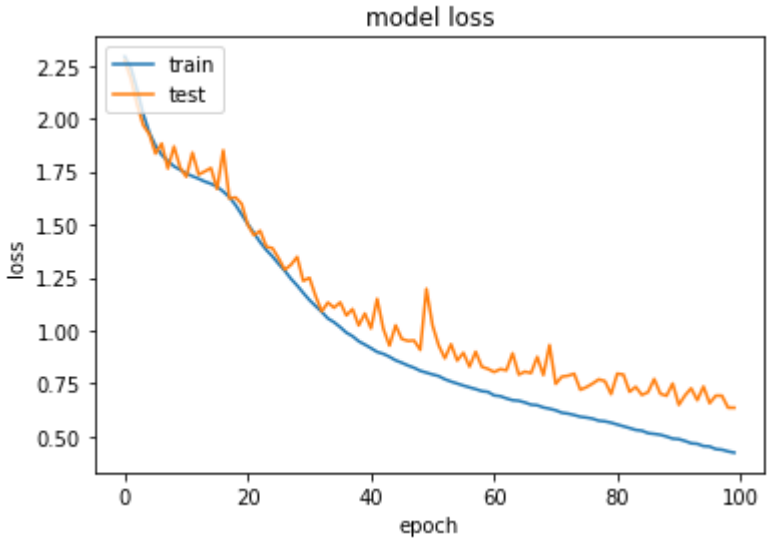
```
Train on 12160 samples, validate on 3040 samples
Epoch 1/250
 - 130s - loss: 0.4146 - acc: 0.8470 - val_loss: 0.6215 - val_acc: 0.7773

Epoch 00001: val_loss improved from 0.63542 to 0.62152, saving model to C:/Us
ers/pushkar/ML/machine-learning/projects/capstone/saved_models/weights.best.f
rom_scratch.hdf5
Epoch 2/250
 - 128s - loss: 0.4108 - acc: 0.8516 - val_loss: 0.6032 - val_acc: 0.7822

Epoch 00002: val_loss improved from 0.62152 to 0.60324, saving model to C:/Us
ers/pushkar/ML/machine-learning/projects/capstone/saved_models/weights.best.f
rom_scratch.hdf5
Epoch 3/250
 - 128s - loss: 0.4038 - acc: 0.8515 - val_loss: 0.6542 - val_acc: 0.7691

Epoch 00003: val_loss did not improve
Epoch 4/250
 - 129s - loss: 0.3956 - acc: 0.8541 - val_loss: 0.6531 - val_acc: 0.7556

Epoch 00004: val_loss did not improve
Epoch 5/250
 - 128s - loss: 0.3958 - acc: 0.8558 - val_loss: 0.6119 - val_acc: 0.7901

Epoch 00005: val_loss did not improve
Epoch 6/250
 - 128s - loss: 0.3943 - acc: 0.8609 - val_loss: 0.6858 - val_acc: 0.7615

Epoch 00006: val_loss did not improve
Epoch 7/250
 - 128s - loss: 0.3861 - acc: 0.8613 - val_loss: 0.6615 - val_acc: 0.7714

Epoch 00007: val_loss did not improve
Epoch 8/250
 - 128s - loss: 0.3795 - acc: 0.8640 - val_loss: 0.5795 - val_acc: 0.7944

Epoch 00008: val_loss improved from 0.60324 to 0.57949, saving model to C:/Us
ers/pushkar/ML/machine-learning/projects/capstone/saved_models/weights.best.f
rom_scratch.hdf5
Epoch 9/250
 - 128s - loss: 0.3712 - acc: 0.8700 - val_loss: 0.5852 - val_acc: 0.7987

Epoch 00009: val_loss did not improve
Epoch 10/250
 - 129s - loss: 0.3686 - acc: 0.8716 - val_loss: 0.5842 - val_acc: 0.8000

Epoch 00010: val_loss did not improve
Epoch 11/250
 - 130s - loss: 0.3627 - acc: 0.8739 - val_loss: 0.6878 - val_acc: 0.7691

Epoch 00011: val_loss did not improve
Epoch 12/250
```

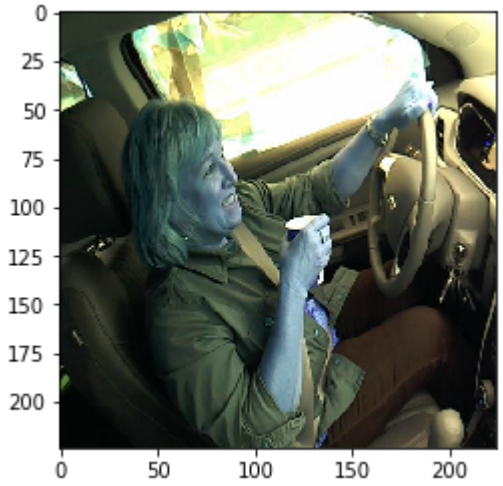
```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-10-19355c9c7b8d> in <module>()
      4 model.load_weights('C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weights.best.from_scratch.hdf5')
      5
----> 6 history = train_model(250)

<ipython-input-7-bfab7781caa9> in train_model(_epochs)
     19
     20     history = model.fit(train_tensors, train_targets_onehot, validation_split=.20,
---> 21           epochs=epochs, batch_size=32, callbacks=[checkpointer], verbose=2)
     22     return history
     23

~\Anaconda3\lib\site-packages\keras\models.py in fit(self, x, y, batch_size, epochs, verbose, callbacks, validation_split, validation_data, shuffle, class_weight, sample_weight, initial_epoch, steps_per_epoch, validation_steps, **kwargs)
    961                               initial_epoch=initial_epoch,
    962                               steps_per_epoch=steps_per_epoch,
--> 963                               validation_steps=validation_steps)
    964
    965     def evaluate(self, x=None, y=None,

~\Anaconda3\lib\site-packages\keras\engine\training.py in fit(self, x, y, batch_size, epochs, verbose, callbacks, validation_split, validation_data, shuffle, class_weight, sample_weight, initial_epoch, steps_per_epoch, validation_steps, **kwargs)
   1710                               initial_epoch=initial_epoch,
   1711                               steps_per_epoch=steps_per_epoch,
-> 1712                               validation_steps=validation_steps)
   1713
   1714     def evaluate(self, x=None, y=None,

~\Anaconda3\lib\site-packages\keras\engine\training.py in _fit_loop(self, f, ins, out_labels, batch_size, epochs, verbose, callbacks, val_f, val_ins, shuffle, callback_metrics, initial_epoch, steps_per_epoch, validation_steps)
   1233                         ins_batch[i] = ins_batch[i].toarray()
   1234
-> 1235                     outs = f(ins_batch)
   1236                     if not isinstance(outs, list):
   1237                         outs = [outs]

~\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py in __call__(self, inputs)
   2473         session = get_session()
   2474         updated = session.run(fetches=fetches, feed_dict=feed_dict,
-> 2475                               **self.session_kwargs)
   2476         return updated[:len(self.outputs)]
   2477
```

```
~\Anaconda3\lib\site-packages\tensorflow\python\client\session.py in run
(self, fetches, feed_dict, options, run_metadata)
    776     try:
    777       result = self._run(None, fetches, feed_dict, options_ptr,
--> 778                          run_metadata_ptr)
    779       if run_metadata:
    780         proto_data = tf_session.TF_GetBuffer(run_metadata_ptr)

~\Anaconda3\lib\site-packages\tensorflow\python\client\session.py in _ru
n(self, handle, fetches, feed_dict, options, run_metadata)
    980     if final_fetches or final_targets:
    981       results = self._do_run(handle, final_targets, final_fetche
s,
--> 982                              feed_dict_string, options, run_meta
data)
    983     else:
    984       results = []

~\Anaconda3\lib\site-packages\tensorflow\python\client\session.py in _do
_run(self, handle, target_list, fetch_list, feed_dict, options, run_meta
data)
   1030     if handle is None:
   1031       return self._do_call(_run_fn, self._session, feed_dict, fe
tch_list,
-> 1032                            target_list, options, run_metadata)
   1033     else:
   1034       return self._do_call(_prun_fn, self._session, handle, feed
_dict,

~\Anaconda3\lib\site-packages\tensorflow\python\client\session.py in _do
_call(self, fn, *args)
   1037   def _do_call(self, fn, *args):
   1038     try:
-> 1039       return fn(*args)
   1040     except errors.OpError as e:
   1041       message = compat.as_text(e.message)

~\Anaconda3\lib\site-packages\tensorflow\python\client\session.py in _ru
n_fn(session, feed_dict, fetch_list, target_list, options, run_metadata)
   1019         return tf_session.TF_Run(session, options,
   1020                                  feed_dict, fetch_list, target_l
ist,
-> 1021                                  status, run_metadata)
   1022
   1023     def _prun_fn(session, handle, feed_dict, fetch_list):

KeyboardInterrupt:
```

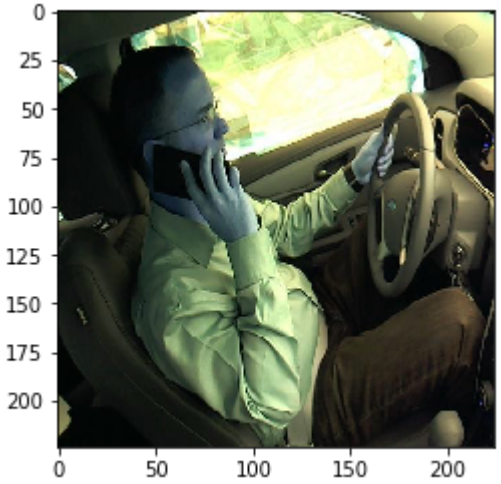
In [11]:
```
import matplotlib.pyplot as plt
import numpy as py


# history for accuracy
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

#  history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()

p = model.predict(test_tensors)
#print (p)
z=np.argmax(p,axis=1)
#print("z = ", z)
for i in range(1,15):
    img = np.squeeze(np.array(test_tensors[i]))
    displayImage(img)
    print("Predicted class", getClass(z[i]))
    print ("Actual Class", getClass(test_targets[i]))

def predict_distraction():
    # get index of predicted distraction for each image in test set
    distraction_predictions = [np.argmax(model.predict(np.expand_dims(tensor,
axis=0))) for tensor in test_tensors]

    # report test accuracy
    test_accuracy = 100*np.sum(np.array(distraction_predictions)==np.argmax(te
st_targets, axis=0))/len(distraction_predictions)
    print('Test accuracy: %.4f%%' % test_accuracy)
    return test_accuracy

predict_distraction()
```



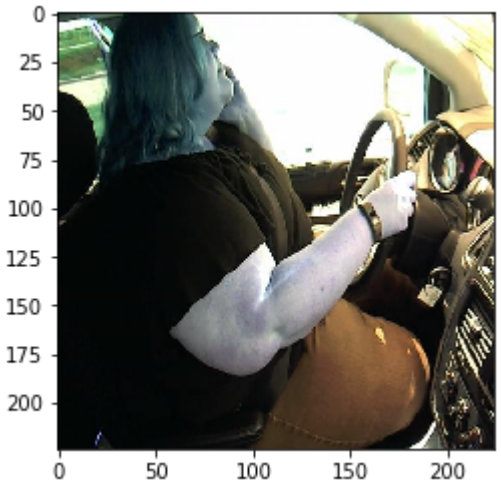




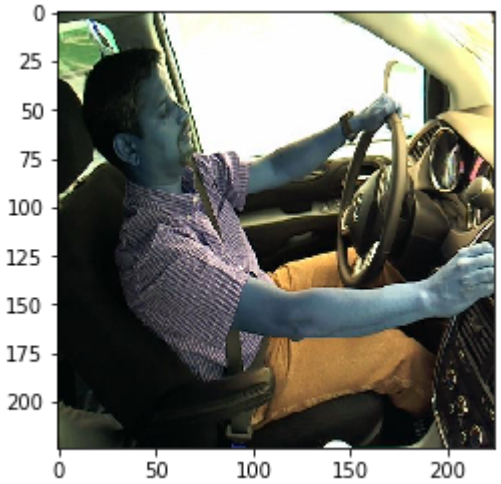

```
Predicted class drinking
Actual Class drinking
```



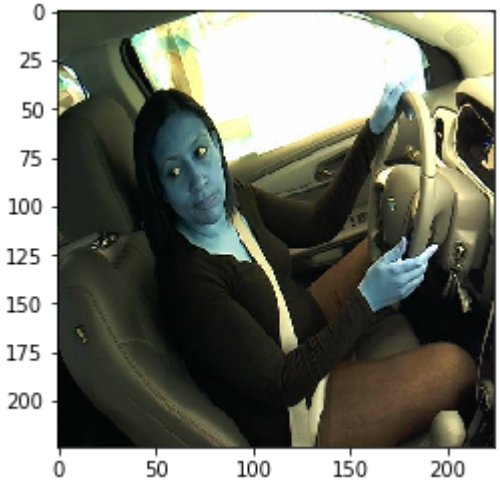

```
Predicted class talking on the phone - right
Actual Class talking on the phone - right
```



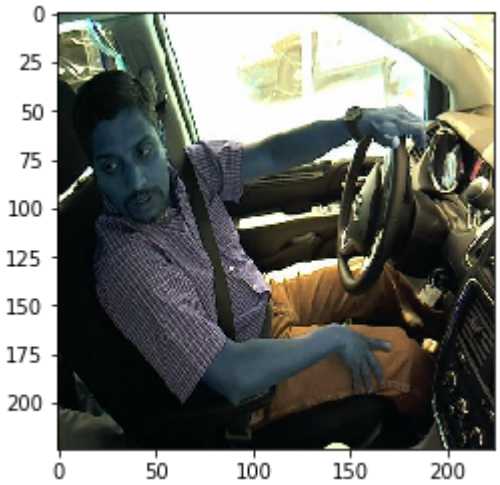

```
Predicted class talking on the phone - left
Actual Class talking on the phone - left
```



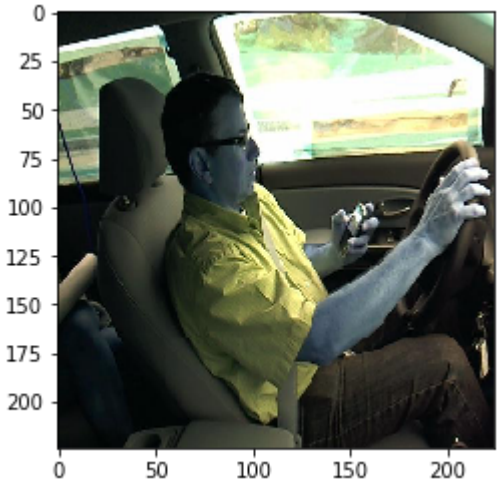

```
Predicted class operating the radio
Actual Class operating the radio
```



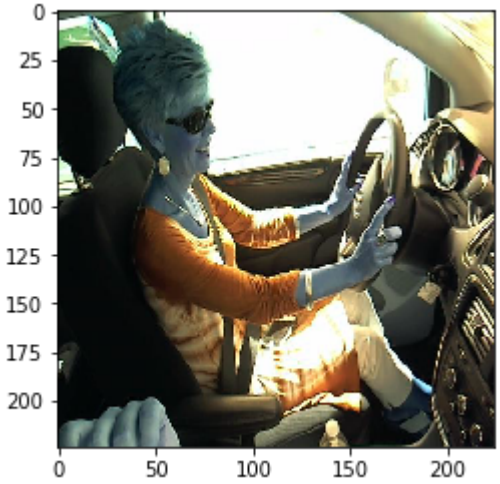

```
Predicted class talking to passenger
Actual Class talking to passenger
```



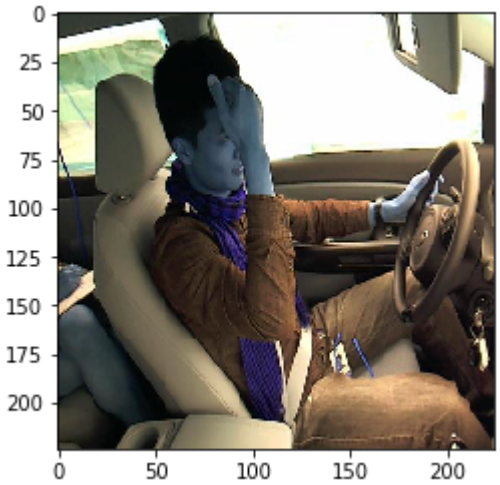

```
Predicted class talking to passenger
Actual Class talking to passenger
```



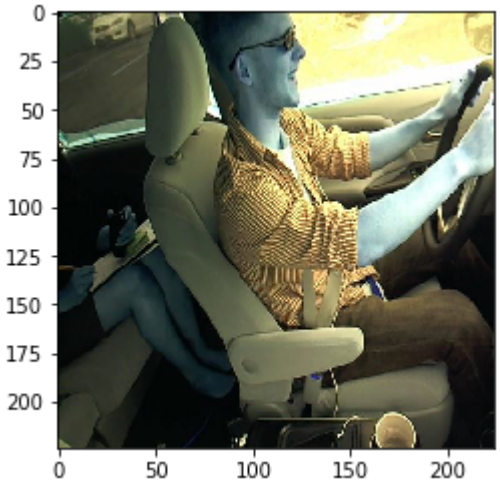

```
Predicted class texting - left
Actual Class texting - left
```



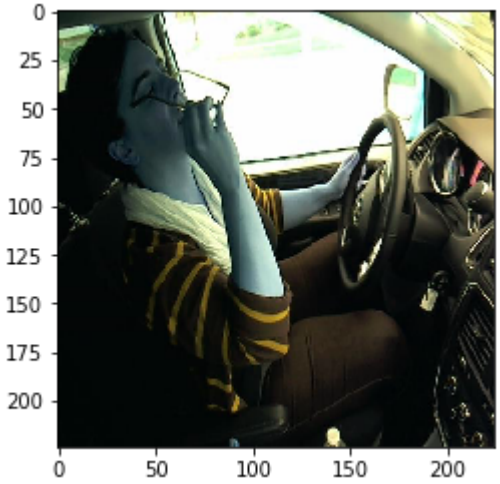

```
Predicted class safe driving
Actual Class safe driving
```



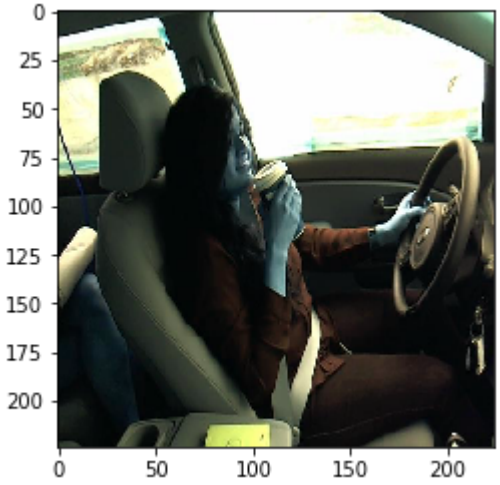

```
Predicted class hair and makeup
Actual Class hair and makeup
```



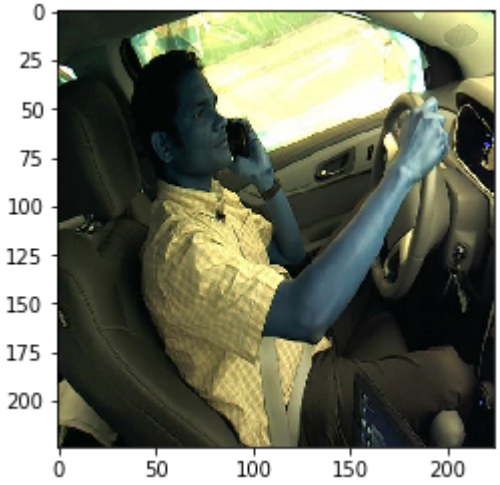

```
Predicted class operating the radio
Actual Class safe driving
```



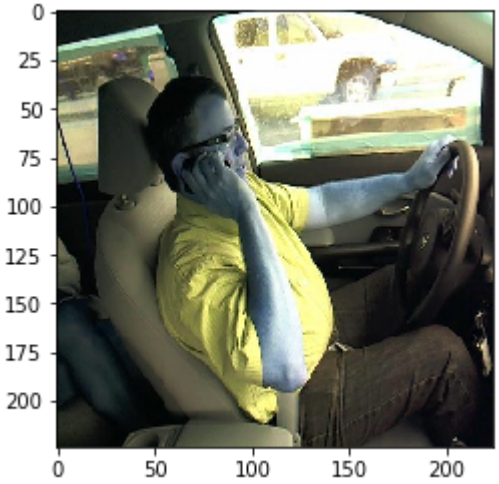
Predicted class talking on the phone - right
Actual Class hair and makeup



Predicted class drinking
Actual Class drinking



Predicted class texting - left
Actual Class talking on the phone - left




```
Predicted class talking on the phone - right
Actual Class talking on the phone - right
Test accuracy: 11.4737%
```

Out[11]: 11.473684210526315