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 [01:25<00:00, 177.75it/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, 160.48it/s]
```

Baseline Model Architecture

In [7]:
```
from keras.layers import Conv2D, MaxPooling2D, GlobalAveragePooling2D
from keras.layers import Dropout, Flatten, Dense
from keras.models import Sequential
from keras.utils import plot_model

model = Sequential()

    ### TODO: Define your architecture.
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(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_5 (Conv2D)            (None, 221, 221, 10)      490       
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 55, 55, 10)        0         
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 52, 52, 10)        1610      
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 13, 13, 10)        0         
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 10, 10, 10)        1610      
_________________________________________________________________
max_pooling2d_6 (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: 4,040
Trainable params: 4,040
Non-trainable params: 0
_________________________________________________________________
```

Train the Model

In [8]:
```
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(100)
```

```
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
 - 86s - loss: 2.3025 - acc: 0.1024 - val_loss: 2.2996 - val_acc: 0.1428

Epoch 00001: val_loss improved from inf to 2.29956, saving model to C:/U
sers/pushkar/ML/machine-learning/projects/capstone/saved_models/weights.
best.from_scratch.hdf5
Epoch 2/100
 - 70s - loss: 2.2862 - acc: 0.1525 - val_loss: 2.2725 - val_acc: 0.1579

Epoch 00002: val_loss improved from 2.29956 to 2.27248, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 3/100
 - 68s - loss: 2.2251 - acc: 0.1854 - val_loss: 2.1722 - val_acc: 0.1891

Epoch 00003: val_loss improved from 2.27248 to 2.17218, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 4/100
 - 68s - loss: 2.1112 - acc: 0.1939 - val_loss: 2.0552 - val_acc: 0.1891

Epoch 00004: val_loss improved from 2.17218 to 2.05517, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 5/100
 - 68s - loss: 1.9981 - acc: 0.1987 - val_loss: 1.9516 - val_acc: 0.1931

Epoch 00005: val_loss improved from 2.05517 to 1.95163, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 6/100
 - 69s - loss: 1.9218 - acc: 0.2012 - val_loss: 1.8945 - val_acc: 0.1961

Epoch 00006: val_loss improved from 1.95163 to 1.89453, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 7/100
 - 71s - loss: 1.8716 - acc: 0.2036 - val_loss: 1.8669 - val_acc: 0.1974
```

```
Epoch 00007: val_loss improved from 1.89453 to 1.86694, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 8/100
 - 71s - loss: 1.8361 - acc: 0.2027 - val_loss: 1.8359 - val_acc: 0.2016

Epoch 00008: val_loss improved from 1.86694 to 1.83589, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 9/100
 - 70s - loss: 1.8143 - acc: 0.2067 - val_loss: 1.8129 - val_acc: 0.1957

Epoch 00009: val_loss improved from 1.83589 to 1.81295, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 10/100
 - 70s - loss: 1.7975 - acc: 0.2026 - val_loss: 1.8098 - val_acc: 0.2115

Epoch 00010: val_loss improved from 1.81295 to 1.80984, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 11/100
 - 71s - loss: 1.7856 - acc: 0.2073 - val_loss: 1.8262 - val_acc: 0.2056

Epoch 00011: val_loss did not improve
Epoch 12/100
 - 68s - loss: 1.7739 - acc: 0.2056 - val_loss: 1.8137 - val_acc: 0.2010

Epoch 00012: val_loss did not improve
Epoch 13/100
 - 68s - loss: 1.7642 - acc: 0.2106 - val_loss: 1.7759 - val_acc: 0.2082

Epoch 00013: val_loss improved from 1.80984 to 1.77587, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 14/100
 - 71s - loss: 1.7574 - acc: 0.2073 - val_loss: 1.8056 - val_acc: 0.2010

Epoch 00014: val_loss did not improve
Epoch 15/100
 - 72s - loss: 1.7467 - acc: 0.2156 - val_loss: 1.7647 - val_acc: 0.1957

Epoch 00015: val_loss improved from 1.77587 to 1.76467, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 16/100
 - 68s - loss: 1.7370 - acc: 0.2137 - val_loss: 1.7427 - val_acc: 0.2105

Epoch 00016: val_loss improved from 1.76467 to 1.74270, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 17/100
 - 69s - loss: 1.7297 - acc: 0.2153 - val_loss: 1.7382 - val_acc: 0.2049

Epoch 00017: val_loss improved from 1.74270 to 1.73818, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
```

```
hts.best.from_scratch.hdf5
Epoch 18/100
 - 68s - loss: 1.7239 - acc: 0.2230 - val_loss: 1.7367 - val_acc: 0.2188

Epoch 00018: val_loss improved from 1.73818 to 1.73666, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 19/100
 - 70s - loss: 1.7212 - acc: 0.2167 - val_loss: 1.7800 - val_acc: 0.2141

Epoch 00019: val_loss did not improve
Epoch 20/100
 - 67s - loss: 1.7147 - acc: 0.2269 - val_loss: 1.7198 - val_acc: 0.2303

Epoch 00020: val_loss improved from 1.73666 to 1.71979, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 21/100
 - 68s - loss: 1.7105 - acc: 0.2284 - val_loss: 1.7202 - val_acc: 0.2240

Epoch 00021: val_loss did not improve
Epoch 22/100
 - 67s - loss: 1.7017 - acc: 0.2368 - val_loss: 1.7184 - val_acc: 0.2385

Epoch 00022: val_loss improved from 1.71979 to 1.71843, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 23/100
 - 67s - loss: 1.6922 - acc: 0.2399 - val_loss: 1.7055 - val_acc: 0.2411

Epoch 00023: val_loss improved from 1.71843 to 1.70546, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 24/100
 - 67s - loss: 1.6817 - acc: 0.2546 - val_loss: 1.6994 - val_acc: 0.2461

Epoch 00024: val_loss improved from 1.70546 to 1.69937, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 25/100
 - 67s - loss: 1.6741 - acc: 0.2660 - val_loss: 1.6865 - val_acc: 0.2572

Epoch 00025: val_loss improved from 1.69937 to 1.68649, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 26/100
 - 67s - loss: 1.6632 - acc: 0.2663 - val_loss: 1.6935 - val_acc: 0.2694

Epoch 00026: val_loss did not improve
Epoch 27/100
 - 67s - loss: 1.6492 - acc: 0.2887 - val_loss: 1.6794 - val_acc: 0.2753

Epoch 00027: val_loss improved from 1.68649 to 1.67941, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 28/100
 - 67s - loss: 1.6360 - acc: 0.2909 - val_loss: 1.6562 - val_acc: 0.2987
```

```
Epoch 00028: val_loss improved from 1.67941 to 1.65621, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 29/100
 - 67s - loss: 1.6244 - acc: 0.3094 - val_loss: 1.6521 - val_acc: 0.2875

Epoch 00029: val_loss improved from 1.65621 to 1.65211, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 30/100
 - 67s - loss: 1.6094 - acc: 0.3174 - val_loss: 1.6341 - val_acc: 0.3010

Epoch 00030: val_loss improved from 1.65211 to 1.63414, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 31/100
 - 67s - loss: 1.5922 - acc: 0.3290 - val_loss: 1.6071 - val_acc: 0.3191

Epoch 00031: val_loss improved from 1.63414 to 1.60709, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 32/100
 - 67s - loss: 1.5738 - acc: 0.3292 - val_loss: 1.6054 - val_acc: 0.3161

Epoch 00032: val_loss improved from 1.60709 to 1.60544, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 33/100
 - 67s - loss: 1.5587 - acc: 0.3436 - val_loss: 1.6027 - val_acc: 0.3342

Epoch 00033: val_loss improved from 1.60544 to 1.60273, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 34/100
 - 67s - loss: 1.5436 - acc: 0.3461 - val_loss: 1.5810 - val_acc: 0.3207

Epoch 00034: val_loss improved from 1.60273 to 1.58105, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 35/100
 - 67s - loss: 1.5277 - acc: 0.3545 - val_loss: 1.5782 - val_acc: 0.3352

Epoch 00035: val_loss improved from 1.58105 to 1.57820, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 36/100
 - 67s - loss: 1.5127 - acc: 0.3620 - val_loss: 1.5492 - val_acc: 0.3490

Epoch 00036: val_loss improved from 1.57820 to 1.54918, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 37/100
 - 67s - loss: 1.4983 - acc: 0.3674 - val_loss: 1.5687 - val_acc: 0.3178

Epoch 00037: val_loss did not improve
Epoch 38/100
```

```
 - 67s - loss: 1.4839 - acc: 0.3762 - val_loss: 1.5131 - val_acc: 0.3829

Epoch 00038: val_loss improved from 1.54918 to 1.51314, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 39/100
 - 67s - loss: 1.4643 - acc: 0.3866 - val_loss: 1.5049 - val_acc: 0.3589

Epoch 00039: val_loss improved from 1.51314 to 1.50492, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 40/100
 - 67s - loss: 1.4454 - acc: 0.3913 - val_loss: 1.4864 - val_acc: 0.3609

Epoch 00040: val_loss improved from 1.50492 to 1.48640, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 41/100
 - 67s - loss: 1.4258 - acc: 0.4133 - val_loss: 1.4538 - val_acc: 0.3776

Epoch 00041: val_loss improved from 1.48640 to 1.45376, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 42/100
 - 67s - loss: 1.4010 - acc: 0.4153 - val_loss: 1.4501 - val_acc: 0.4026

Epoch 00042: val_loss improved from 1.45376 to 1.45012, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 43/100
 - 67s - loss: 1.3762 - acc: 0.4139 - val_loss: 1.4389 - val_acc: 0.3842

Epoch 00043: val_loss improved from 1.45012 to 1.43889, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 44/100
 - 67s - loss: 1.3524 - acc: 0.4229 - val_loss: 1.3930 - val_acc: 0.4079

Epoch 00044: val_loss improved from 1.43889 to 1.39297, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 45/100
 - 67s - loss: 1.3350 - acc: 0.4321 - val_loss: 1.3732 - val_acc: 0.4174

Epoch 00045: val_loss improved from 1.39297 to 1.37317, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 46/100
 - 67s - loss: 1.3142 - acc: 0.4417 - val_loss: 1.3611 - val_acc: 0.4141

Epoch 00046: val_loss improved from 1.37317 to 1.36108, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 47/100
 - 67s - loss: 1.2995 - acc: 0.4422 - val_loss: 1.3522 - val_acc: 0.4197

Epoch 00047: val_loss improved from 1.36108 to 1.35225, saving model to
```

```
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 48/100
 - 67s - loss: 1.2822 - acc: 0.4465 - val_loss: 1.3254 - val_acc: 0.4322

Epoch 00048: val_loss improved from 1.35225 to 1.32541, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 49/100
 - 67s - loss: 1.2702 - acc: 0.4509 - val_loss: 1.3306 - val_acc: 0.4161

Epoch 00049: val_loss did not improve
Epoch 50/100
 - 67s - loss: 1.2538 - acc: 0.4678 - val_loss: 1.2986 - val_acc: 0.4365

Epoch 00050: val_loss improved from 1.32541 to 1.29865, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 51/100
 - 67s - loss: 1.2393 - acc: 0.4533 - val_loss: 1.2865 - val_acc: 0.4411

Epoch 00051: val_loss improved from 1.29865 to 1.28647, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 52/100
 - 67s - loss: 1.2248 - acc: 0.4665 - val_loss: 1.2865 - val_acc: 0.4365

Epoch 00052: val_loss did not improve
Epoch 53/100
 - 67s - loss: 1.2120 - acc: 0.4721 - val_loss: 1.3700 - val_acc: 0.3681

Epoch 00053: val_loss did not improve
Epoch 54/100
 - 67s - loss: 1.1983 - acc: 0.4772 - val_loss: 1.2746 - val_acc: 0.4428

Epoch 00054: val_loss improved from 1.28647 to 1.27456, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 55/100
 - 67s - loss: 1.1875 - acc: 0.4734 - val_loss: 1.2770 - val_acc: 0.4372

Epoch 00055: val_loss did not improve
Epoch 56/100
 - 67s - loss: 1.1786 - acc: 0.4780 - val_loss: 1.2480 - val_acc: 0.4497

Epoch 00056: val_loss improved from 1.27456 to 1.24803, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 57/100
 - 67s - loss: 1.1632 - acc: 0.4854 - val_loss: 1.2562 - val_acc: 0.4451

Epoch 00057: val_loss did not improve
Epoch 58/100
 - 67s - loss: 1.1517 - acc: 0.4852 - val_loss: 1.2357 - val_acc: 0.4516

Epoch 00058: val_loss improved from 1.24803 to 1.23569, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
```

```
hts.best.from_scratch.hdf5
Epoch 59/100
 - 67s - loss: 1.1381 - acc: 0.4988 - val_loss: 1.2278 - val_acc: 0.4615

Epoch 00059: val_loss improved from 1.23569 to 1.22784, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 60/100
 - 67s - loss: 1.1286 - acc: 0.4942 - val_loss: 1.2320 - val_acc: 0.4602

Epoch 00060: val_loss did not improve
Epoch 61/100
 - 67s - loss: 1.1192 - acc: 0.4945 - val_loss: 1.2320 - val_acc: 0.4497

Epoch 00061: val_loss did not improve
Epoch 62/100
 - 67s - loss: 1.1058 - acc: 0.5012 - val_loss: 1.2099 - val_acc: 0.4562

Epoch 00062: val_loss improved from 1.22784 to 1.20994, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 63/100
 - 67s - loss: 1.0968 - acc: 0.5090 - val_loss: 1.2042 - val_acc: 0.4737

Epoch 00063: val_loss improved from 1.20994 to 1.20415, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 64/100
 - 67s - loss: 1.0881 - acc: 0.5101 - val_loss: 1.1796 - val_acc: 0.4944

Epoch 00064: val_loss improved from 1.20415 to 1.17958, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 65/100
 - 67s - loss: 1.0765 - acc: 0.5126 - val_loss: 1.2122 - val_acc: 0.4691

Epoch 00065: val_loss did not improve
Epoch 66/100
 - 67s - loss: 1.0682 - acc: 0.5250 - val_loss: 1.2082 - val_acc: 0.4704

Epoch 00066: val_loss did not improve
Epoch 67/100
 - 67s - loss: 1.0612 - acc: 0.5217 - val_loss: 1.1800 - val_acc: 0.4938

Epoch 00067: val_loss did not improve
Epoch 68/100
 - 67s - loss: 1.0514 - acc: 0.5231 - val_loss: 1.2113 - val_acc: 0.4572

Epoch 00068: val_loss did not improve
Epoch 69/100
 - 67s - loss: 1.0431 - acc: 0.5331 - val_loss: 1.1602 - val_acc: 0.4895

Epoch 00069: val_loss improved from 1.17958 to 1.16016, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 70/100
 - 67s - loss: 1.0324 - acc: 0.5400 - val_loss: 1.1695 - val_acc: 0.4845
```

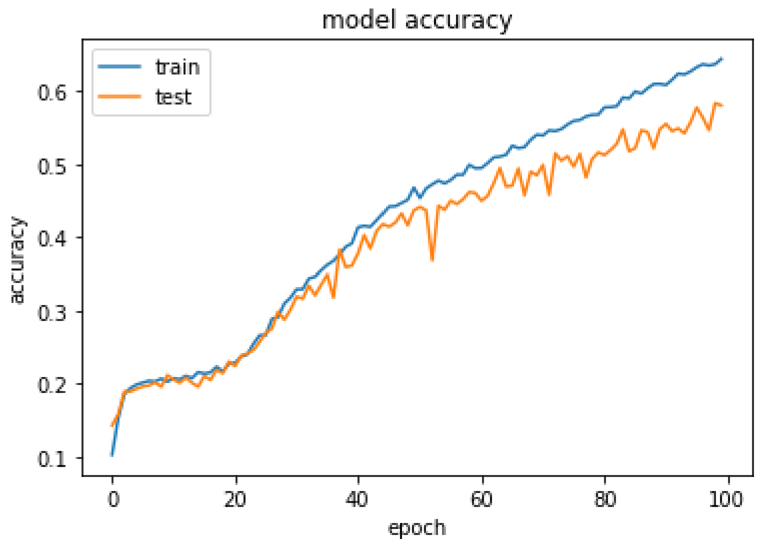
```
Epoch 00070: val_loss did not improve
Epoch 71/100
 - 67s - loss: 1.0240 - acc: 0.5390 - val_loss: 1.1363 - val_acc: 0.4987

Epoch 00071: val_loss improved from 1.16016 to 1.13635, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 72/100
 - 67s - loss: 1.0172 - acc: 0.5460 - val_loss: 1.1686 - val_acc: 0.4579

Epoch 00072: val_loss did not improve
Epoch 73/100
 - 67s - loss: 1.0099 - acc: 0.5451 - val_loss: 1.1348 - val_acc: 0.5145

Epoch 00073: val_loss improved from 1.13635 to 1.13483, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 74/100
 - 67s - loss: 1.0013 - acc: 0.5479 - val_loss: 1.1386 - val_acc: 0.5046

Epoch 00074: val_loss did not improve
Epoch 75/100
 - 67s - loss: 0.9948 - acc: 0.5541 - val_loss: 1.1173 - val_acc: 0.5105

Epoch 00075: val_loss improved from 1.13483 to 1.11733, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 76/100
 - 67s - loss: 0.9881 - acc: 0.5590 - val_loss: 1.1186 - val_acc: 0.4964

Epoch 00076: val_loss did not improve
Epoch 77/100
 - 67s - loss: 0.9783 - acc: 0.5602 - val_loss: 1.1238 - val_acc: 0.5138

Epoch 00077: val_loss did not improve
Epoch 78/100
 - 67s - loss: 0.9703 - acc: 0.5650 - val_loss: 1.1426 - val_acc: 0.4816

Epoch 00078: val_loss did not improve
Epoch 79/100
 - 67s - loss: 0.9601 - acc: 0.5673 - val_loss: 1.1515 - val_acc: 0.5069

Epoch 00079: val_loss did not improve
Epoch 80/100
 - 67s - loss: 0.9613 - acc: 0.5676 - val_loss: 1.1102 - val_acc: 0.5158

Epoch 00080: val_loss improved from 1.11733 to 1.11021, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 81/100
 - 67s - loss: 0.9525 - acc: 0.5775 - val_loss: 1.1051 - val_acc: 0.5122

Epoch 00081: val_loss improved from 1.11021 to 1.10512, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 82/100
```

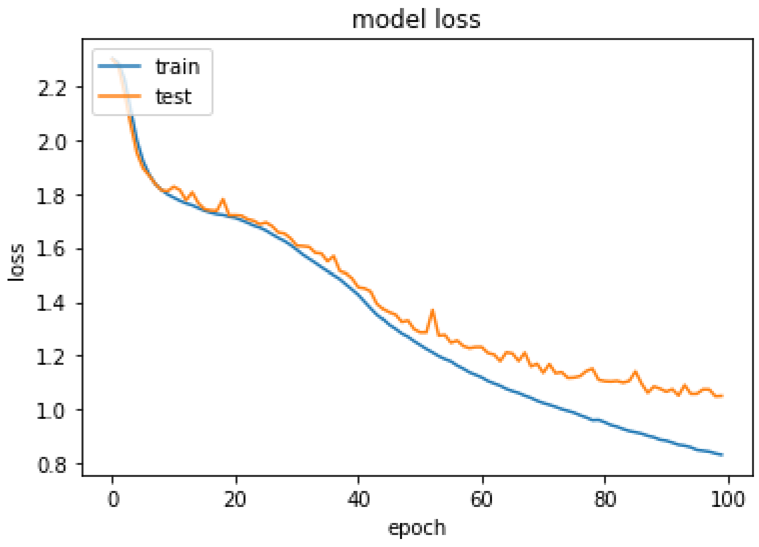
```
 - 67s - loss: 0.9422 - acc: 0.5779 - val_loss: 1.1038 - val_acc: 0.5191

Epoch 00082: val_loss improved from 1.10512 to 1.10383, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 83/100
 - 67s - loss: 0.9359 - acc: 0.5791 - val_loss: 1.1060 - val_acc: 0.5276

Epoch 00083: val_loss did not improve
Epoch 84/100
 - 67s - loss: 0.9273 - acc: 0.5910 - val_loss: 1.0997 - val_acc: 0.5474

Epoch 00084: val_loss improved from 1.10383 to 1.09974, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 85/100
 - 67s - loss: 0.9199 - acc: 0.5897 - val_loss: 1.1060 - val_acc: 0.5174

Epoch 00085: val_loss did not improve
Epoch 86/100
 - 67s - loss: 0.9152 - acc: 0.5993 - val_loss: 1.1416 - val_acc: 0.5217

Epoch 00086: val_loss did not improve
Epoch 87/100
 - 67s - loss: 0.9102 - acc: 0.5964 - val_loss: 1.0957 - val_acc: 0.5461

Epoch 00087: val_loss improved from 1.09974 to 1.09569, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 88/100
 - 67s - loss: 0.9022 - acc: 0.6035 - val_loss: 1.0619 - val_acc: 0.5437

Epoch 00088: val_loss improved from 1.09569 to 1.06194, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 89/100
 - 67s - loss: 0.8966 - acc: 0.6094 - val_loss: 1.0853 - val_acc: 0.5214

Epoch 00089: val_loss did not improve
Epoch 90/100
 - 67s - loss: 0.8876 - acc: 0.6097 - val_loss: 1.0772 - val_acc: 0.5477

Epoch 00090: val_loss did not improve
Epoch 91/100
 - 67s - loss: 0.8839 - acc: 0.6081 - val_loss: 1.0662 - val_acc: 0.5549

Epoch 00091: val_loss did not improve
Epoch 92/100
 - 67s - loss: 0.8768 - acc: 0.6154 - val_loss: 1.0750 - val_acc: 0.5451

Epoch 00092: val_loss did not improve
Epoch 93/100
 - 67s - loss: 0.8688 - acc: 0.6238 - val_loss: 1.0518 - val_acc: 0.5487

Epoch 00093: val_loss improved from 1.06194 to 1.05179, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
```

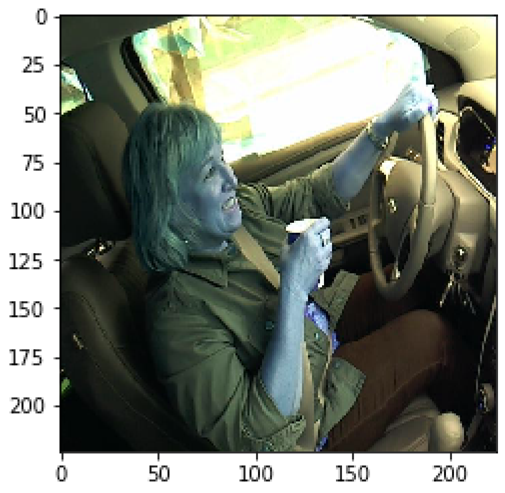
```
Epoch 94/100
 - 67s - loss: 0.8654 - acc: 0.6225 - val_loss: 1.0907 - val_acc: 0.5421

Epoch 00094: val_loss did not improve
Epoch 95/100
 - 67s - loss: 0.8594 - acc: 0.6267 - val_loss: 1.0581 - val_acc: 0.5569

Epoch 00095: val_loss did not improve
Epoch 96/100
 - 67s - loss: 0.8496 - acc: 0.6325 - val_loss: 1.0584 - val_acc: 0.5773

Epoch 00096: val_loss did not improve
Epoch 97/100
 - 67s - loss: 0.8464 - acc: 0.6365 - val_loss: 1.0745 - val_acc: 0.5635

Epoch 00097: val_loss did not improve
Epoch 98/100
 - 67s - loss: 0.8433 - acc: 0.6349 - val_loss: 1.0742 - val_acc: 0.5464

Epoch 00098: val_loss did not improve
Epoch 99/100
 - 68s - loss: 0.8366 - acc: 0.6366 - val_loss: 1.0488 - val_acc: 0.5832

Epoch 00099: val_loss improved from 1.05179 to 1.04881, saving model to
C:/Users/pushkar/ML/machine-learning/projects/capstone/saved_models/weig
hts.best.from_scratch.hdf5
Epoch 100/100
 - 67s - loss: 0.8316 - acc: 0.6438 - val_loss: 1.0504 - val_acc: 0.5803

Epoch 00100: val_loss did not improve
```

In [12]:
```
import matplotlib.pyplot as plt
import numpy as py

print (history)

# 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()
```

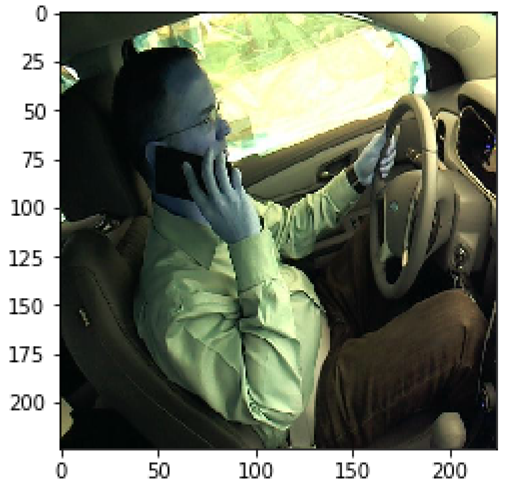
```
<keras.callbacks.History object at 0x000001CCA48CCD68>
```



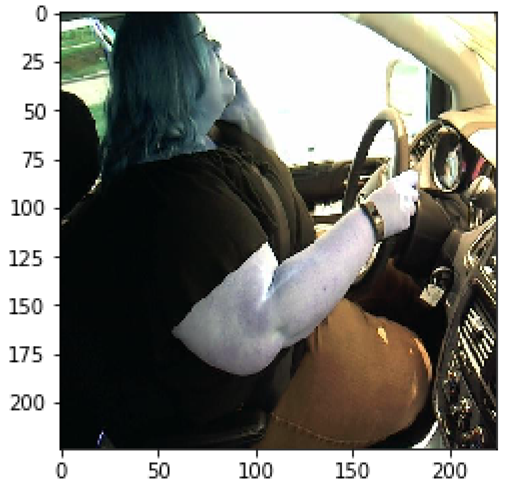




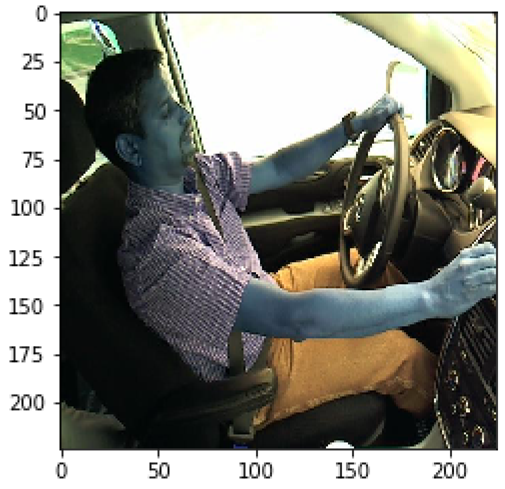
```
Predicted class drinking
Actual Class drinking
```



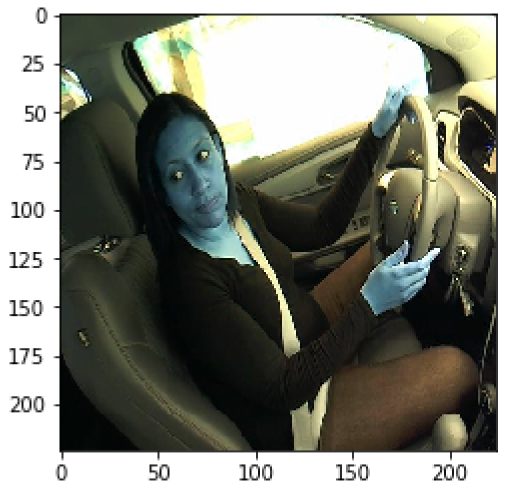
```
Predicted class reaching behind
Actual Class talking on the phone - right
```



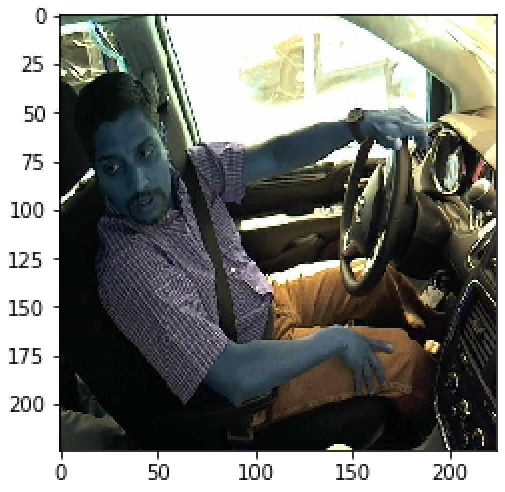
```
Predicted class texting - left
Actual Class talking on the phone - left
```



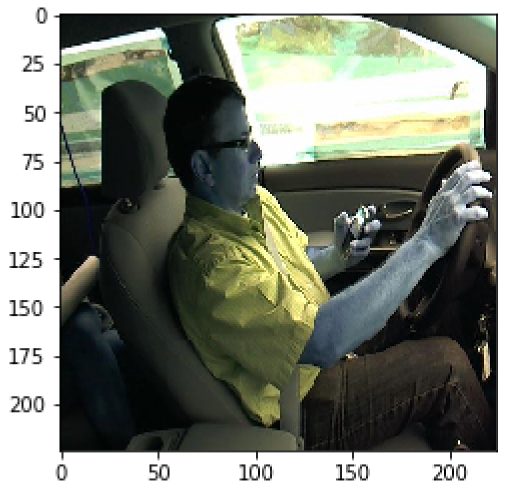
```
Predicted class operating the radio
Actual Class operating the radio
```



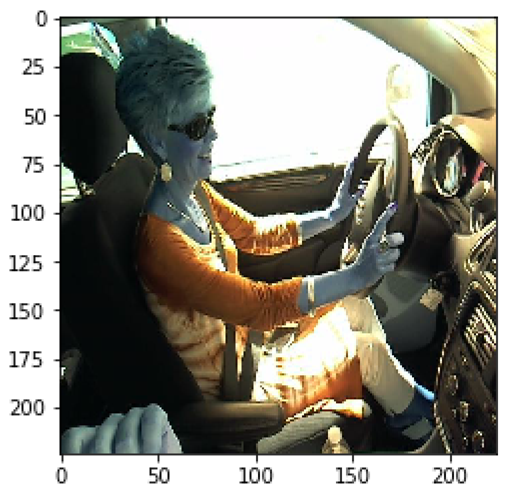
```
Predicted class talking to passenger
Actual Class talking to passenger
```



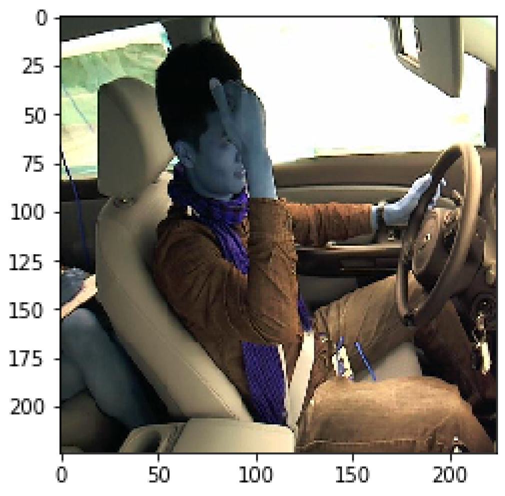
```
Predicted class talking to passenger
Actual Class talking to passenger
```



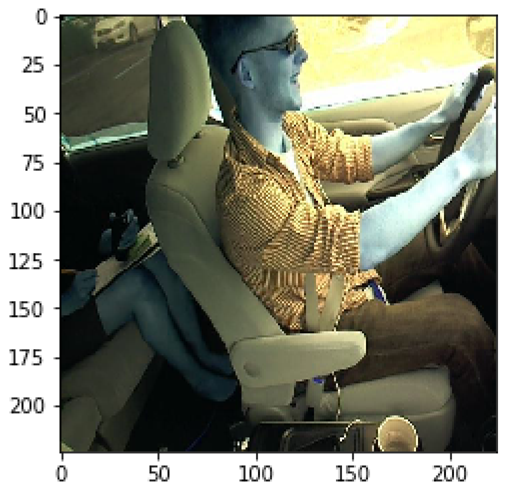
```
Predicted class talking on the phone - left
Actual Class texting - left
```



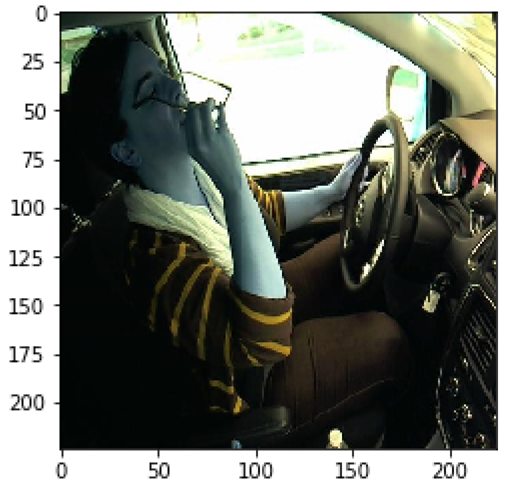
```
Predicted class texting - left
Actual Class safe driving
```



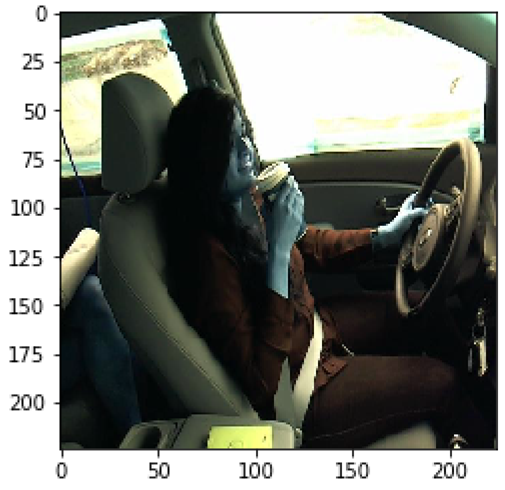
```
Predicted class hair and makeup
Actual Class hair and makeup
```



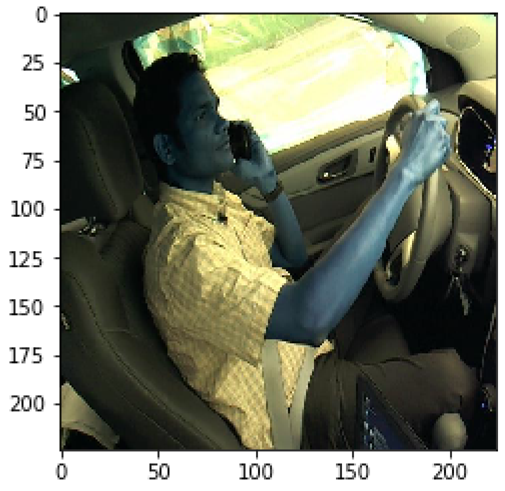
```
Predicted class safe driving
Actual Class safe driving
```



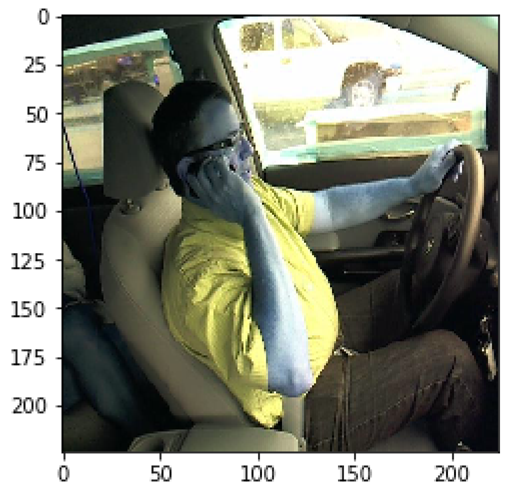
```
Predicted class drinking
Actual Class hair and makeup
```



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



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



```
Predicted class reaching behind
Actual Class talking on the phone - right
Test accuracy: 11.8947%
```

```
Out[12]: 11.894736842105264
```