Open in Colab
(https://colab.research.google.com/gist/prathamshroff/9efd66398e13327114d4f5ad61d064aa/pratham-s-driver-distraction-detector.ipynb)

3-D Driver Distraction Detection

In [1]:
```
#@title Run this to download data and prepare our environment! { display-mode: "form" }


import cv2
import numpy as np

def label_to_numpy(labels):
  final_labels = np.zeros((len(labels), 4))
  for i in range(len(labels)):
    label = labels[i]
    if label == 'Attentive':
      final_labels[i,:] = np.array([1, 0, 0, 0])
    if label == 'DrinkingCoffee':
      final_labels[i,:] = np.array([0, 1, 0, 0])
    if label == 'UsingMirror':
      final_labels[i,:] = np.array([0, 0, 1, 0])
    if label == 'UsingRadio':
      final_labels[i,:] = np.array([0, 0, 0, 1])
  return final_labels

def augment(data, augmenter):
  if len(data.shape) == 3:
    return augmenter.augment_image(data)
  if len(data.shape) == 4:
    return augmenter.augment_images(data)

def rotate(data, rotate):
  fun = augmenters.Affine(rotate = rotate)
  return augment(data, fun)

def shear(data, shear):
  fun = augmenters.Affine(shear = shear)
  return augment(data, fun)

def scale(data, scale):
  fun = augmenters.Affine(scale = shear)
  return augment(data, fun)

def flip_left_right(data):
  fun = augmenters.Fliplr()
  return augment(data, fun)

def flip_up_down(data):
```

```
  fun = augmenters.Flipud()
  return augment(data, fun)

def remove_color(data, channel):
  new_data = data.copy()
  if len(data.shape) == 3:
    new_data[:,:,channel] = 0
    return new_data
  if len(data.shape) == 4:
    new_data[:,:,:,channel] = 0
    return new_data

class pkg:
  #### DOWNLOADING AND LOADING DATA
  def get_metadata(metadata_path, which_splits = ['train', 'test'
]):
    '''returns metadata dataframe which contains columns of:
       * index: index of data into numpy data
       * class: class of image
       * split: which dataset split is this a part of?
    '''
    metadata = pd.read_csv(metadata_path)
    keep_idx = metadata['split'].isin(which_splits)
    metadata = metadata[keep_idx]

    # Get dataframes for each class.
    df_coffee_train = metadata[(metadata['class'] == 'DrinkingCof
fee') & \
                           (metadata['split'] == 'train')]
    df_coffee_test = metadata[(metadata['class'] == 'DrinkingCoff
ee') & \
                           (metadata['split'] == 'test')]
    df_mirror_train = metadata[(metadata['class'] == 'UsingMirror
') & \
                           (metadata['split'] == 'train')]
    df_mirror_test = metadata[(metadata['class'] == 'UsingMirror'
) & \
                           (metadata['split'] == 'test')]
    df_attentive_train = metadata[(metadata['class'] == 'Attentiv
e') & \
                           (metadata['split'] == 'train')]
    df_attentive_test = metadata[(metadata['class'] == 'Attentive
') & \
                           (metadata['split'] == 'test')]
    df_radio_train = metadata[(metadata['class'] == 'UsingRadio')
& \
                           (metadata['split'] == 'train')]
    df_radio_test = metadata[(metadata['class'] == 'UsingRadio')
& \
                           (metadata['split'] == 'test')]

    # Get number of items in class with lowest number of images.
    num_samples_train = min(df_coffee_train.shape[0], \
                            df_mirror_train.shape[0], \
```

```
                              df_attentive_train.shape[0], \
                              df_radio_train.shape[0])
    num_samples_test = min(df_coffee_test.shape[0], \
                              df_mirror_test.shape[0], \
                              df_attentive_test.shape[0], \
                              df_radio_test.shape[0])

    # Resample each of the classes and concatenate the images.
    metadata_train = pd.concat([df_coffee_train.sample(num_sample
s_train), \
                            df_mirror_train.sample(num_samples_trai
n), \
                            df_attentive_train.sample(num_samples_t
rain), \
                            df_radio_train.sample(num_samples_train
) ])
    metadata_test = pd.concat([df_coffee_test.sample(num_samples_
test), \
                            df_mirror_test.sample(num_samples_test)
, \
                            df_attentive_test.sample(num_samples_te
st), \
                            df_radio_test.sample(num_samples_test)
])

    metadata = pd.concat( [metadata_train, metadata_test] )

    return metadata

  def get_data_split(split_name, flatten, all_data, metadata, ima
ge_shape):
    '''
    returns images (data), labels from folder of format [image_fo
lder]/[split_name]/[class_name]/
    flattens if flatten option is True
    '''
    # Get dataframes for each class.
    df_coffee_train = metadata[(metadata['class'] == 'DrinkingCof
fee') & \
                           (metadata['split'] == 'train')]
    df_coffee_test = metadata[(metadata['class'] == 'DrinkingCoff
ee') & \
                           (metadata['split'] == 'test')]
    df_mirror_train = metadata[(metadata['class'] == 'UsingMirror
') & \
                           (metadata['split'] == 'train')]
    df_mirror_test = metadata[(metadata['class'] == 'UsingMirror'
) & \
                           (metadata['split'] == 'test')]
    df_attentive_train = metadata[(metadata['class'] == 'Attentiv
e') & \
                           (metadata['split'] == 'train')]
    df_attentive_test = metadata[(metadata['class'] == 'Attentive
```

```
') & \
                          (metadata['split'] == 'test')]
    df_radio_train = metadata[(metadata['class'] == 'UsingRadio')
& \
                          (metadata['split'] == 'train')]
    df_radio_test = metadata[(metadata['class'] == 'UsingRadio')
& \
                          (metadata['split'] == 'test')]

    # Get number of items in class with lowest number of images.
    num_samples_train = min(df_coffee_train.shape[0], \
                            df_mirror_train.shape[0], \
                            df_attentive_train.shape[0], \
                            df_radio_train.shape[0])
    num_samples_test = min(df_coffee_test.shape[0], \
                            df_mirror_test.shape[0], \
                            df_attentive_test.shape[0], \
                            df_radio_test.shape[0])

    # Resample each of the classes and concatenate the images.
    metadata_train = pd.concat([df_coffee_train.sample(num_sample
s_train), \
                            df_mirror_train.sample(num_samples_trai
n), \
                            df_attentive_train.sample(num_samples_t
rain), \
                            df_radio_train.sample(num_samples_train
) ])
    metadata_test = pd.concat([df_coffee_test.sample(num_samples_
test), \
                            df_mirror_test.sample(num_samples_test)
, \
                            df_attentive_test.sample(num_samples_te
st), \
                            df_radio_test.sample(num_samples_test)
])

    metadata = pd.concat( [metadata_train, metadata_test] )

    sub_df = metadata[metadata['split'].isin([split_name])]
    index  = sub_df['index'].values
    labels = sub_df['class'].values
    data = all_data[index,:]
    if flatten:
      data = data.reshape([-1, np.product(image_shape)])
    return data, labels

  def get_train_data(flatten, all_data, metadata, image_shape):
    return get_data_split('train', flatten, all_data, metadata, i
mage_shape)

  def get_test_data(flatten, all_data, metadata, image_shape):
    return get_data_split('test', flatten, all_data, metadata, im
```

```
age_shape)

  def get_field_data(flatten, all_data, metadata, image_shape):
    return get_data_split('field', flatten, all_data, metadata, i
mage_shape)

class helpers:
  #### PLOTTING
  def plot_one_image(data, labels = [], index = None, image_shape
= [64,64,3]):
    '''
    if data is a single image, display that image

    if data is a 4d stack of images, display that image
    '''
    ### cv2.imshow('image', data)


    num_dims   = len(data.shape)
    num_labels = len(labels)
    target_shape = (64,64,3)
    # reshape data if necessary
    if num_dims == 1:
      data = data.reshape(target_shape)
    if num_dims == 2:
      data = data.reshape(np.vstack[-1, image_shape])
    num_dims   = len(data.shape)

    # check if single or multiple images
    if num_dims == 3:
      if num_labels > 1:
        print('Multiple labels does not make sense for single ima
ge.')
        return

      label = labels
      if num_labels == 0:
        label = ''
      image = data

    if num_dims == 4:
      image = data[index, :]
      label = labels[index]

    # plot image of interest
    print('Label: %s'%label)
    plt.imshow(image)
    plt.show()



  #### QUERYING AND COMBINING DATA
  def get_misclassified_data(data, labels, predictions):
```

```
    '''
    Gets the data and labels that are misclassified in a classifi
cation task
    Returns:
    -missed_data
    -missed_labels
    -predicted_labels (corresponding to missed_labels)
    -missed_index (indices of items in original dataset)
    '''
    missed_index     = np.where(np.abs(predictions.squeeze() - la
bels.squeeze()) > 0)[0]
    missed_labels    = labels[missed_index]
    missed_data      = data[missed_index,:]
    predicted_labels = predictions[missed_index]
    return missed_data, missed_labels, predicted_labels, missed_i
ndex

  def combine_data(data_list, labels_list):
    return np.concatenate(data_list, axis = 0), np.concatenate(la
bels_list, axis = 0)

  def model_to_string(model):
    import re
    stringlist = []
    model.summary(print_fn=lambda x: stringlist.append(x))
    sms = "\n".join(stringlist)
    sms = re.sub('_\d\d\d','', sms)
    sms = re.sub('_\d\d','', sms)
    sms = re.sub('_\d','', sms)
    return sms

  def plot_acc(history, ax = None, xlabel = 'Epoch #'):
    # i'm sorry for this function's code. i am so sorry.
    history = history.history
    history.update({'epoch':list(range(len(history['val_acc'])))}
)
    history = pd.DataFrame.from_dict(history)

    best_epoch = history.sort_values(by = 'val_acc', ascending =
False).iloc[0]['epoch']

    if not ax:
      f, ax = plt.subplots(1,1)
    sns.lineplot(x = 'epoch', y = 'val_acc', data = history, labe
l = 'Validation', ax = ax)
    sns.lineplot(x = 'epoch', y = 'acc', data = history, label =
'Training', ax = ax)
    ax.axhline(0.25, linestyle = '--',color='red', label = 'Chanc
e')
    ax.axvline(x = best_epoch, linestyle = '--', color = 'green',
label = 'Best Epoch')
    ax.legend(loc = 1)
    ax.set_ylim([0.01, 1])
```

```
    ax.set_xlabel(xlabel)
    ax.set_ylabel('Accuracy (Fraction)')

    plt.show()

class models:
  def DenseClassifier(hidden_layer_sizes, nn_params, dropout = 1)
:
    model = Sequential()
    model.add(Flatten(input_shape = nn_params['input_shape']))
    for ilayer in hidden_layer_sizes:
      model.add(Dense(ilayer, activation = 'relu'))
      if dropout:
        model.add(Dropout(dropout))
    model.add(Dense(units = nn_params['output_neurons'], activati
on = nn_params['output_activation']))
    model.compile(loss=nn_params['loss'],
                  optimizer=optimizers.SGD(lr=1e-4, momentum=0.95
),
                  metrics=['accuracy'])
    return model

  def CNNClassifier(num_hidden_layers, nn_params, dropout = 1):
    model = Sequential()

    model.add(Conv2D(32, (3, 3), input_shape=nn_params['input_sha
pe'], padding = 'same'))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    for i in range(num_hidden_layers-1):
        model.add(Conv2D(32, (3, 3), padding = 'same'))
        model.add(Activation('relu'))
        model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Flatten())

    model.add(Dense(units = 128, activation = 'relu'))
    model.add(Dropout(dropout))

    model.add(Dense(units = 64, activation = 'relu'))


    model.add(Dense(units = nn_params['output_neurons'], activati
on = nn_params['output_activation']))

    # initiate RMSprop optimizer
    opt = keras.optimizers.rmsprop(lr=1e-4, decay=1e-6)

    # Let's train the model using RMSprop
    model.compile(loss=nn_params['loss'],
                  optimizer=opt,
```

```
                  metrics=['accuracy'])
    return model

  def TransferClassifier(name, nn_params, trainable = True):
    expert_dict = {'VGG16': VGG16,
                   'VGG19': VGG19,
                   'ResNet50':ResNet50,
                   'DenseNet121':DenseNet121}

    expert_conv = expert_dict[name](weights = 'imagenet',
                                    include_top = False
,
                                    input_shape = nn_pa
rams['input_shape'])
    for layer in expert_conv.layers:
      layer.trainable = trainable

    expert_model = Sequential()
    expert_model.add(expert_conv)
    expert_model.add(GlobalAveragePooling2D())

    expert_model.add(Dense(128, activation = 'relu'))
    expert_model.add(Dropout(0.3))

    expert_model.add(Dense(64, activation = 'relu'))

    expert_model.add(Dense(nn_params['output_neurons'], activatio
n = nn_params['output_activation']))

    expert_model.compile(loss = nn_params['loss'],
                  optimizer = optimizers.SGD(lr=1e-4, momentum=0.
95),
                  metrics=['accuracy'])

    return expert_model

import gdown
import zipfile

import os
import numpy as np
import pandas as pd

import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn import model_selection
```

```
from collections import Counter

import keras
from keras.models import Sequential
from keras.layers import Activation, MaxPooling2D, Dropout, Flatt
en, Reshape, Dense, Conv2D, GlobalAveragePooling2D
from keras.wrappers.scikit_learn import KerasClassifier
import keras.optimizers as optimizers
from keras.callbacks import ModelCheckpoint

from keras.applications import VGG16, VGG19, ResNet50, DenseNet12
1

from imgaug import augmenters

### defining project variables
# file variables
image_data_url       = 'https://drive.google.com/uc?id=1qmTuUyn05
25-612yS-wkp8gHB72Wv_XP'
metadata_url         = 'https://drive.google.com/uc?id=1OfKnq3uIT
29sXjWSZqOOpceig8Ul24OW'
image_data_path      = './image_data.npy'
metadata_path        = './metadata.csv'
image_shape          = (64, 64, 3)

# neural net parameters
nn_params = {}
nn_params['input_shape']       = image_shape
nn_params['output_neurons']    = 4
nn_params['loss']              = 'categorical_crossentropy'
nn_params['output_activation'] = 'softmax'

###
gdown.download(image_data_url, image_data_path , True)
gdown.download(metadata_url, metadata_path , True)


### pre-loading all data of interest
_all_data = np.load('image_data.npy')
_metadata = pkg.get_metadata(metadata_path, ['train','test','fiel
d'])

### preparing definitions
# downloading and loading data
get_data_split = pkg.get_data_split
get_metadata    = lambda :                  pkg.get_metadata(metad
ata_path, ['train','test'])
get_train_data  = lambda flatten = False : pkg.get_train_data(fla
tten = flatten, all_data = _all_data, metadata = _metadata, image
_shape = image_shape)
get_test_data   = lambda flatten = False : pkg.get_test_data(flat
ten = flatten, all_data = _all_data, metadata = _metadata, image_
```

```
shape = image_shape)
get_field_data  = lambda flatten = False : pkg.get_field_data(fla
tten = flatten, all_data = _all_data, metadata = _metadata, image
_shape = image_shape)

# plotting
plot_one_image = lambda data, labels = [], index = None: helpers.
plot_one_image(data = data, labels = labels, index = index, image
_shape = image_shape);
plot_acc       = lambda history: helpers.plot_acc(history)

# querying and combining data
model_to_string        = lambda model: helpers.model_to_string(mo
del)
get_misclassified_data = helpers.get_misclassified_data;
combine_data           = helpers.combine_data;

# models with input parameters
DenseClassifier     = lambda hidden_layer_sizes: models.DenseClas
sifier(hidden_layer_sizes = hidden_layer_sizes, nn_params = nn_pa
rams);
CNNClassifier       = lambda num_hidden_layers: models.CNNClassif
ier(num_hidden_layers, nn_params = nn_params);
TransferClassifier  = lambda name: models.TransferClassifier(name
= name, nn_params = nn_params);

monitor = ModelCheckpoint('./model.h5', monitor='val_acc', verbos
e=0, save_best_only=True, save_weights_only=False, mode='auto', p
eriod=1)


# prepare more

! pwd
! pip3 install scipy==1.1.0
! pip install git+https://github.com/raghakot/keras-vis.git -U

! pip3 install scipy==1.1.0
! pip install git+https://github.com/raghakot/keras-vis.git -U
```

```
Using TensorFlow backend.
/content
Requirement already satisfied: scipy==1.1.0 in /usr/local/lib/pyt
hon3.6/dist-packages (1.1.0)
Requirement already satisfied: numpy>=1.8.2 in /usr/local/lib/pyt
hon3.6/dist-packages (from scipy==1.1.0) (1.16.4)
Collecting git+https://github.com/raghakot/keras-vis.git
  Cloning https://github.com/raghakot/keras-vis.git to /tmp/pip-r
eq-build-nnxgl58q
  Running command git clone -q https://github.com/raghakot/keras-
vis.git /tmp/pip-req-build-nnxgl58q
Requirement already satisfied, skipping upgrade: keras>=2.0 in /u
sr/local/lib/python3.6/dist-packages (from keras-vis==0.5.0) (2.2
```

```
.4)
Requirement already satisfied, skipping upgrade: six in /usr/loca
l/lib/python3.6/dist-packages (from keras-vis==0.5.0) (1.12.0)
Requirement already satisfied, skipping upgrade: scikit-image in
/usr/local/lib/python3.6/dist-packages (from keras-vis==0.5.0) (0
.15.0)
Requirement already satisfied, skipping upgrade: matplotlib in /u
sr/local/lib/python3.6/dist-packages (from keras-vis==0.5.0) (3.0
.3)
Requirement already satisfied, skipping upgrade: h5py in /usr/loc
al/lib/python3.6/dist-packages (from keras-vis==0.5.0) (2.8.0)
Requirement already satisfied, skipping upgrade: keras-applicatio
ns>=1.0.6 in /usr/local/lib/python3.6/dist-packages (from keras>=
2.0->keras-vis==0.5.0) (1.0.8)
Requirement already satisfied, skipping upgrade: scipy>=0.14 in /
usr/local/lib/python3.6/dist-packages (from keras>=2.0->keras-vis
==0.5.0) (1.1.0)
Requirement already satisfied, skipping upgrade: keras-preprocess
ing>=1.0.5 in /usr/local/lib/python3.6/dist-packages (from keras>
=2.0->keras-vis==0.5.0) (1.1.0)
Requirement already satisfied, skipping upgrade: numpy>=1.9.1 in
/usr/local/lib/python3.6/dist-packages (from keras>=2.0->keras-vi
s==0.5.0) (1.16.4)
Requirement already satisfied, skipping upgrade: pyyaml in /usr/l
ocal/lib/python3.6/dist-packages (from keras>=2.0->keras-vis==0.5
.0) (3.13)
Requirement already satisfied, skipping upgrade: pillow>=4.3.0 in
/usr/local/lib/python3.6/dist-packages (from scikit-image->keras-
vis==0.5.0) (4.3.0)
Requirement already satisfied, skipping upgrade: networkx>=2.0 in
/usr/local/lib/python3.6/dist-packages (from scikit-image->keras-
vis==0.5.0) (2.3)
Requirement already satisfied, skipping upgrade: PyWavelets>=0.4.
0 in /usr/local/lib/python3.6/dist-packages (from scikit-image->k
eras-vis==0.5.0) (1.0.3)
Requirement already satisfied, skipping upgrade: imageio>=2.0.1 i
n /usr/local/lib/python3.6/dist-packages (from scikit-image->kera
s-vis==0.5.0) (2.4.1)
Requirement already satisfied, skipping upgrade: kiwisolver>=1.0.
1 in /usr/local/lib/python3.6/dist-packages (from matplotlib->ker
as-vis==0.5.0) (1.1.0)
Requirement already satisfied, skipping upgrade: python-dateutil>
=2.1 in /usr/local/lib/python3.6/dist-packages (from matplotlib->
keras-vis==0.5.0) (2.5.3)
Requirement already satisfied, skipping upgrade: cycler>=0.10 in
/usr/local/lib/python3.6/dist-packages (from matplotlib->keras-vi
s==0.5.0) (0.10.0)
Requirement already satisfied, skipping upgrade: pyparsing!=2.0.4
,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.6/dist-package
s (from matplotlib->keras-vis==0.5.0) (2.4.0)
Requirement already satisfied, skipping upgrade: olefile in /usr/
local/lib/python3.6/dist-packages (from pillow>=4.3.0->scikit-ima
ge->keras-vis==0.5.0) (0.46)
Requirement already satisfied, skipping upgrade: decorator>=4.3.0
```

```
in /usr/local/lib/python3.6/dist-packages (from networkx>=2.0->sc
ikit-image->keras-vis==0.5.0) (4.4.0)
Requirement already satisfied, skipping upgrade: setuptools in /u
sr/local/lib/python3.6/dist-packages (from kiwisolver>=1.0.1->mat
plotlib->keras-vis==0.5.0) (41.0.1)
Building wheels for collected packages: keras-vis
  Building wheel for keras-vis (setup.py) ... done
  Stored in directory: /tmp/pip-ephem-wheel-cache-wl02721c/wheels
/c5/ae/e7/b34d1cb48b1898f606a5cce08ebc9521fa0588f37f1e590d9f
Successfully built keras-vis
Installing collected packages: keras-vis
  Found existing installation: keras-vis 0.5.0
    Uninstalling keras-vis-0.5.0:
      Successfully uninstalled keras-vis-0.5.0
Successfully installed keras-vis-0.5.0
Requirement already satisfied: scipy==1.1.0 in /usr/local/lib/pyt
hon3.6/dist-packages (1.1.0)
Requirement already satisfied: numpy>=1.8.2 in /usr/local/lib/pyt
hon3.6/dist-packages (from scipy==1.1.0) (1.16.4)
Collecting git+https://github.com/raghakot/keras-vis.git
  Cloning https://github.com/raghakot/keras-vis.git to /tmp/pip-r
eq-build-0x6txl8h
  Running command git clone -q https://github.com/raghakot/keras-
vis.git /tmp/pip-req-build-0x6txl8h
Requirement already satisfied, skipping upgrade: keras>=2.0 in /u
sr/local/lib/python3.6/dist-packages (from keras-vis==0.5.0) (2.2
.4)
Requirement already satisfied, skipping upgrade: six in /usr/loca
l/lib/python3.6/dist-packages (from keras-vis==0.5.0) (1.12.0)
Requirement already satisfied, skipping upgrade: scikit-image in
/usr/local/lib/python3.6/dist-packages (from keras-vis==0.5.0) (0
.15.0)
Requirement already satisfied, skipping upgrade: matplotlib in /u
sr/local/lib/python3.6/dist-packages (from keras-vis==0.5.0) (3.0
.3)
Requirement already satisfied, skipping upgrade: h5py in /usr/loc
al/lib/python3.6/dist-packages (from keras-vis==0.5.0) (2.8.0)
Requirement already satisfied, skipping upgrade: scipy>=0.14 in /
usr/local/lib/python3.6/dist-packages (from keras>=2.0->keras-vis
==0.5.0) (1.1.0)
Requirement already satisfied, skipping upgrade: keras-applicatio
ns>=1.0.6 in /usr/local/lib/python3.6/dist-packages (from keras>=
2.0->keras-vis==0.5.0) (1.0.8)
Requirement already satisfied, skipping upgrade: keras-preprocess
ing>=1.0.5 in /usr/local/lib/python3.6/dist-packages (from keras>
=2.0->keras-vis==0.5.0) (1.1.0)
Requirement already satisfied, skipping upgrade: pyyaml in /usr/l
ocal/lib/python3.6/dist-packages (from keras>=2.0->keras-vis==0.5
.0) (3.13)
Requirement already satisfied, skipping upgrade: numpy>=1.9.1 in
/usr/local/lib/python3.6/dist-packages (from keras>=2.0->keras-vi
s==0.5.0) (1.16.4)
Requirement already satisfied, skipping upgrade: PyWavelets>=0.4.
0 in /usr/local/lib/python3.6/dist-packages (from scikit-image->k
```

```
eras-vis==0.5.0) (1.0.3)
Requirement already satisfied, skipping upgrade: imageio>=2.0.1 i
n /usr/local/lib/python3.6/dist-packages (from scikit-image->kera
s-vis==0.5.0) (2.4.1)
Requirement already satisfied, skipping upgrade: pillow>=4.3.0 in
/usr/local/lib/python3.6/dist-packages (from scikit-image->keras-
vis==0.5.0) (4.3.0)
Requirement already satisfied, skipping upgrade: networkx>=2.0 in
/usr/local/lib/python3.6/dist-packages (from scikit-image->keras-
vis==0.5.0) (2.3)
Requirement already satisfied, skipping upgrade: python-dateutil>
=2.1 in /usr/local/lib/python3.6/dist-packages (from matplotlib->
keras-vis==0.5.0) (2.5.3)
Requirement already satisfied, skipping upgrade: kiwisolver>=1.0.
1 in /usr/local/lib/python3.6/dist-packages (from matplotlib->ker
as-vis==0.5.0) (1.1.0)
Requirement already satisfied, skipping upgrade: cycler>=0.10 in
/usr/local/lib/python3.6/dist-packages (from matplotlib->keras-vi
s==0.5.0) (0.10.0)
Requirement already satisfied, skipping upgrade: pyparsing!=2.0.4
,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/lib/python3.6/dist-package
s (from matplotlib->keras-vis==0.5.0) (2.4.0)
Requirement already satisfied, skipping upgrade: olefile in /usr/
local/lib/python3.6/dist-packages (from pillow>=4.3.0->scikit-ima
ge->keras-vis==0.5.0) (0.46)
Requirement already satisfied, skipping upgrade: decorator>=4.3.0
in /usr/local/lib/python3.6/dist-packages (from networkx>=2.0->sc
ikit-image->keras-vis==0.5.0) (4.4.0)
Requirement already satisfied, skipping upgrade: setuptools in /u
sr/local/lib/python3.6/dist-packages (from kiwisolver>=1.0.1->mat
plotlib->keras-vis==0.5.0) (41.0.1)
Building wheels for collected packages: keras-vis
  Building wheel for keras-vis (setup.py) ... done
  Stored in directory: /tmp/pip-ephem-wheel-cache-z2m3utdz/wheels
/c5/ae/e7/b34d1cb48b1898f606a5cce08ebc9521fa0588f37f1e590d9f
Successfully built keras-vis
Installing collected packages: keras-vis
  Found existing installation: keras-vis 0.5.0
    Uninstalling keras-vis-0.5.0:
      Successfully uninstalled keras-vis-0.5.0
Successfully installed keras-vis-0.5.0
```

In [0]:
```
#@title Run this to train your data! { display-mode: "form" }

from vis.visualization import visualize_saliency, visualize_cam

# train test splitting

train_data, train_labels = get_train_data(flatten=True)
test_data, test_labels = get_test_data(flatten=True)

train_data = train_data.reshape([-1, 64, 64, 3])
test_data = test_data.reshape([-1, 64, 64, 3])

train_labels = label_to_numpy(train_labels)
test_labels = label_to_numpy(test_labels)

# model making

vgg_model = TransferClassifier(name = 'VGG16')
vgg_model.compile(loss='categorical_crossentropy', optimizer = op
timizers.SGD(lr=1e-3, momentum=0.95), metrics = ['accuracy'])
history = vgg_model.fit(train_data, train_labels, epochs = 5, val
idation_data = (test_data, test_labels), shuffle = True, callback
s = [monitor])
```

```
WARNING: Logging before flag parsing goes to stderr.
W0711 15:57:59.059623 140694134564736 deprecation_wrapper.py:119]
From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorf
low_backend.py:74: The name tf.get_default_graph is deprecated. P
lease use tf.compat.v1.get_default_graph instead.

W0711 15:57:59.095239 140694134564736 deprecation_wrapper.py:119]
From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorf
low_backend.py:517: The name tf.placeholder is deprecated. Please
use tf.compat.v1.placeholder instead.

W0711 15:57:59.103902 140694134564736 deprecation_wrapper.py:119]
From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorf
low_backend.py:4138: The name tf.random_uniform is deprecated. Pl
ease use tf.random.uniform instead.

W0711 15:57:59.150730 140694134564736 deprecation_wrapper.py:119]
From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorf
low_backend.py:3976: The name tf.nn.max_pool is deprecated. Pleas
e use tf.nn.max_pool2d instead.

Downloading data from https://github.com/fchollet/deep-learning-m
odels/releases/download/v0.1/vgg16_weights_tf_dim_ordering_tf_ker
nels_notop.h5
58892288/58889256 [==============================] - 5s 0us/step
```

```
W0711 15:58:06.335328 140694134564736 deprecation_wrapper.py:119] From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:174: The name tf.get_default_session is deprecated. Please use tf.compat.v1.get_default_session instead.

W0711 15:58:06.340554 140694134564736 deprecation_wrapper.py:119] From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:181: The name tf.ConfigProto is deprecated. Please use tf.compat.v1.ConfigProto instead.

W0711 15:58:09.822615 140694134564736 deprecation.py:506] From /usr/local/lib/python3.6/dist-packages/keras/backend/tensorflow_backend.py:3445: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
W0711 15:58:09.890247 140694134564736 deprecation_wrapper.py:119] From /usr/local/lib/python3.6/dist-packages/keras/optimizers.py:790: The name tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

W0711 15:58:10.086007 140694134564736 deprecation.py:323] From /usr/local/lib/python3.6/dist-packages/tensorflow/python/ops/math_grad.py:1250: add_dispatch_support.<locals>.wrapper (from tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
Train on 6724 samples, validate on 920 samples
Epoch 1/5
6724/6724 [==============================] - 31s 5ms/step - loss: 0.4249 - acc: 0.8275 - val_loss: 0.9123 - val_acc: 0.7228
Epoch 2/5
6724/6724 [==============================] - 25s 4ms/step - loss: 0.0805 - acc: 0.9784 - val_loss: 0.7622 - val_acc: 0.8435
Epoch 3/5
6688/6724 [============================>.] - ETA: 0s - loss: 0.0374 - acc: 0.9900
```

In [0]:
```
from google.colab import files
uploaded = files.upload()
```

In [0]:
```
#@title Run your program to view the prediction now! { display-mode: "form" }

for fn in uploaded.keys():
  img_name = fn
img = cv2.imread(img_name)
img = cv2.resize(img,(64,64))
img = img.reshape([-1, 64, 64, 3])
print(img_name)

classes = vgg_model.predict(img)

if np.array_equal(classes, np.array([[1, 0, 0, 0]])):
  output = 'Attentive'
elif np.array_equal(classes, np.array([[0., 1., 0., 0.]])):
  output = 'Drinking Coffee'
elif np.array_equal(classes, np.array([[0, 0, 1, 0]])):
  output = 'Using Mirror'
elif np.array_equal(classes, np.array([[0, 0, 0, 1]])):
  output = 'Using Radio'
else:
  output = 'Error. Please Try Again Later or Use Another Image'

if output == 'Attentive':
  print(output)
else:
  print("Distracted.", "Probably", output)
```