In [1]:
```
from __future__ import print_function
import keras
from keras.preprocessing.image import ImageDataGenerator,img_to_array
from keras.models import Sequential
from keras.layers import Dense,Dropout,Activation,Flatten,BatchNormalization
from keras.layers import Conv2D,MaxPooling2D
import os
from keras.optimizers import RMSprop,SGD,Adam
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau
import numpy as np
```

In [2]:
```
import cv2
from keras.models import load_model
```

In [3]:
```
num_classes = 5
img_rows,img_cols = 48,48
batch_size = 32

train_data_dir = 'C:/Users/suman/Downloads/images/train'
validation_data_dir = 'C:/Users/suman/Downloads/images/validation'
```

In [1]:
```
# from google.colab import drive
# drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call d
rive.mount("/content/drive", force_remount=True).
```

In [ ]:
```
# num_classes = 5
# img_rows,img_cols = 48,48
# batch_size = 32

# train_data_dir = '/Users/durgeshthakur/Deep Learning Stuff/Emotion Classificati
# validation_data_dir = '/Users/durgeshthakur/Deep Learning Stuff/Emotion Classif
```

In [4]:
```
train_datagen = ImageDataGenerator(
                    rescale=1./255,
                    rotation_range=30,
                    shear_range=0.3,
                    zoom_range=0.3,
                    width_shift_range=0.4,
                    height_shift_range=0.4,
                    horizontal_flip=True,
                    fill_mode='nearest')

validation_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
                    train_data_dir,
                    color_mode='grayscale',
                    target_size=(img_rows,img_cols),
                    batch_size=batch_size,
                    class_mode='categorical',
                    shuffle=True)

validation_generator = validation_datagen.flow_from_directory(
                            validation_data_dir,
                            color_mode='grayscale',
                            target_size=(img_rows,img_cols),
                            batch_size=batch_size,
                            class_mode='categorical',
                            shuffle=True)
```

```
Found 24282 images belonging to 5 classes.
Found 5937 images belonging to 5 classes.
```

In [5]:
```
model = Sequential()
```

In [6]:
```
# Block-1

model.add(Conv2D(32,(3,3),padding='same',kernel_initializer='he_normal',input_sha
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(32,(3,3),padding='same',kernel_initializer='he_normal',input_sha
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))
```

In [7]:
```
# Block-2

model.add(Conv2D(64,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(64,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))
```

In [8]:
```
# Block-3

model.add(Conv2D(128,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(128,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))
```

In [9]:
```
# Block-4

model.add(Conv2D(256,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Conv2D(256,(3,3),padding='same',kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))
```

In [10]:
```
# Block-5

model.add(Flatten())
model.add(Dense(64,kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
```

In [11]:
```
# Block-6

model.add(Dense(64,kernel_initializer='he_normal'))
model.add(Activation('elu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
```

In [12]:
```
# Block-7

model.add(Dense(num_classes,kernel_initializer='he_normal'))
model.add(Activation('softmax'))

print(model.summary())
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 48, 48, 32)        320
_________________________________________________________________
activation (Activation)      (None, 48, 48, 32)        0
_________________________________________________________________
batch_normalization (BatchNo (None, 48, 48, 32)        128
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 48, 48, 32)        9248
_________________________________________________________________
activation_1 (Activation)    (None, 48, 48, 32)        0
_________________________________________________________________
batch_normalization_1 (Batch (None, 48, 48, 32)        128
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 24, 24, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 24, 24, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 24, 24, 64)        18496
_________________________________________________________________
activation_2 (Activation)    (None, 24, 24, 64)        0
_________________________________________________________________
batch_normalization_2 (Batch (None, 24, 24, 64)        256
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 24, 24, 64)        36928
_________________________________________________________________
activation_3 (Activation)    (None, 24, 24, 64)        0
_________________________________________________________________
batch_normalization_3 (Batch (None, 24, 24, 64)        256
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 12, 12, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 12, 12, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 12, 12, 128)       73856
_________________________________________________________________
activation_4 (Activation)    (None, 12, 12, 128)       0
_________________________________________________________________
batch_normalization_4 (Batch (None, 12, 12, 128)       512
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 12, 12, 128)       147584
_________________________________________________________________
activation_5 (Activation)    (None, 12, 12, 128)       0
_________________________________________________________________
batch_normalization_5 (Batch (None, 12, 12, 128)       512
_________________________________________________________________
```

```
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 128)         0
_________________________________________________________________
dropout_2 (Dropout)          (None, 6, 6, 128)         0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 6, 6, 256)         295168
_________________________________________________________________
activation_6 (Activation)    (None, 6, 6, 256)         0
_________________________________________________________________
batch_normalization_6 (Batch (None, 6, 6, 256)         1024
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 6, 6, 256)         590080
_________________________________________________________________
activation_7 (Activation)    (None, 6, 6, 256)         0
_________________________________________________________________
batch_normalization_7 (Batch (None, 6, 6, 256)         1024
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 3, 3, 256)         0
_________________________________________________________________
dropout_3 (Dropout)          (None, 3, 3, 256)         0
_________________________________________________________________
flatten (Flatten)            (None, 2304)              0
_________________________________________________________________
dense (Dense)                (None, 64)                147520
_________________________________________________________________
activation_8 (Activation)    (None, 64)                0
_________________________________________________________________
batch_normalization_8 (Batch (None, 64)                256
_________________________________________________________________
dropout_4 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                4160
_________________________________________________________________
activation_9 (Activation)    (None, 64)                0
_________________________________________________________________
batch_normalization_9 (Batch (None, 64)                256
_________________________________________________________________
dropout_5 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 5)                 325
_________________________________________________________________
activation_10 (Activation)   (None, 5)                 0
=================================================================
Total params: 1,328,037
Trainable params: 1,325,861
Non-trainable params: 2,176
_________________________________________________________________
None
```

In [13]:

```
checkpoint = ModelCheckpoint('Emotion_little_vgg.h5',
                             monitor='val_loss',
                             mode='min',
                             save_best_only=True,
                             verbose=1)

earlystop = EarlyStopping(monitor='val_loss',
                          min_delta=0,
                          patience=3,
                          verbose=1,
                          restore_best_weights=True
                          )

reduce_lr = ReduceLROnPlateau(monitor='val_loss',
                              factor=0.2,
                              patience=3,
                              verbose=1,
                              min_delta=0.0001)

callbacks = [earlystop,checkpoint,reduce_lr]
```

In [14]:
```
model.compile(loss='categorical_crossentropy',
              optimizer = Adam(lr=0.001),
              metrics=['accuracy'])

nb_train_samples = 24176
nb_validation_samples = 3006
epochs=25

history=model.fit_generator(
                train_generator,
                steps_per_epoch=nb_train_samples//batch_size,
                epochs=epochs,
                callbacks=callbacks,
                validation_data=validation_generator,
                validation_steps=nb_validation_samples//batch_size)
```

```
E:\Anaconda\lib\site-packages\tensorflow\python\keras\engine\training.py:1844:
UserWarning: `Model.fit_generator` is deprecated and will be removed in a futur
e version. Please use `Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '

Epoch 1/25
755/755 [==============================] - 527s 693ms/step - loss: 2.1226 - a
ccuracy: 0.2223 - val_loss: 1.5537 - val_accuracy: 0.3004

Epoch 00001: val_loss improved from inf to 1.55368, saving model to Emotion_l
ittle_vgg.h5
Epoch 2/25
755/755 [==============================] - 489s 648ms/step - loss: 1.5976 - a
ccuracy: 0.2638 - val_loss: 1.5351 - val_accuracy: 0.2957

Epoch 00002: val_loss improved from 1.55368 to 1.53515, saving model to Emoti
on_little_vgg.h5
Epoch 3/25
755/755 [==============================] - 472s 626ms/step - loss: 1.5500 - a
ccuracy: 0.2992 - val_loss: 1.4711 - val_accuracy: 0.3663

Epoch 00003: val_loss improved from 1.53515 to 1.47106, saving model to Emoti
on_little_vgg.h5
Epoch 4/25
755/755 [==============================] - 448s 593ms/step - loss: 1.5238 - a
ccuracy: 0.3159 - val_loss: 1.4299 - val_accuracy: 0.3760

Epoch 00004: val_loss improved from 1.47106 to 1.42988, saving model to Emoti
on_little_vgg.h5
Epoch 5/25
755/755 [==============================] - 425s 563ms/step - loss: 1.4573 - a
ccuracy: 0.3628 - val_loss: 1.4743 - val_accuracy: 0.4113

Epoch 00005: val_loss did not improve from 1.42988
Epoch 6/25
755/755 [==============================] - 407s 538ms/step - loss: 1.3687 - a
ccuracy: 0.4204 - val_loss: 1.1898 - val_accuracy: 0.5134

Epoch 00006: val_loss improved from 1.42988 to 1.18985, saving model to Emoti
on_little_vgg.h5
Epoch 7/25
```

```
755/755 [==============================] - 366s 485ms/step - loss: 1.2857 - a
ccuracy: 0.4619 - val_loss: 1.0016 - val_accuracy: 0.5857

Epoch 00007: val_loss improved from 1.18985 to 1.00161, saving model to Emoti
on_little_vgg.h5
Epoch 8/25
755/755 [==============================] - 399s 529ms/step - loss: 1.2136 - a
ccuracy: 0.4997 - val_loss: 0.9423 - val_accuracy: 0.6179

Epoch 00008: val_loss improved from 1.00161 to 0.94225, saving model to Emoti
on_little_vgg.h5
Epoch 9/25
755/755 [==============================] - 504s 668ms/step - loss: 1.1890 - a
ccuracy: 0.5240 - val_loss: 0.9160 - val_accuracy: 0.6300

Epoch 00009: val_loss improved from 0.94225 to 0.91598, saving model to Emoti
on_little_vgg.h5
Epoch 10/25
755/755 [==============================] - 442s 586ms/step - loss: 1.1616 - a
ccuracy: 0.5296 - val_loss: 0.8710 - val_accuracy: 0.6546

Epoch 00010: val_loss improved from 0.91598 to 0.87101, saving model to Emoti
on_little_vgg.h5
Epoch 11/25
755/755 [==============================] - 498s 660ms/step - loss: 1.1225 - a
ccuracy: 0.5466 - val_loss: 0.8900 - val_accuracy: 0.6552

Epoch 00011: val_loss did not improve from 0.87101
Epoch 12/25
755/755 [==============================] - 459s 607ms/step - loss: 1.1108 - a
ccuracy: 0.5520 - val_loss: 0.8770 - val_accuracy: 0.6475

Epoch 00012: val_loss did not improve from 0.87101
Epoch 13/25
755/755 [==============================] - 465s 616ms/step - loss: 1.1060 - a
ccuracy: 0.5595 - val_loss: 0.8604 - val_accuracy: 0.6663

Epoch 00013: val_loss improved from 0.87101 to 0.86040, saving model to Emoti
on_little_vgg.h5
Epoch 14/25
755/755 [==============================] - 459s 608ms/step - loss: 1.0713 - a
ccuracy: 0.5769 - val_loss: 0.9210 - val_accuracy: 0.6502

Epoch 00014: val_loss did not improve from 0.86040
Epoch 15/25
755/755 [==============================] - 463s 613ms/step - loss: 1.0588 - a
ccuracy: 0.5866 - val_loss: 0.8454 - val_accuracy: 0.6801

Epoch 00015: val_loss improved from 0.86040 to 0.84535, saving model to Emoti
on_little_vgg.h5
Epoch 16/25
755/755 [==============================] - 446s 591ms/step - loss: 1.0450 - a
ccuracy: 0.5949 - val_loss: 0.7866 - val_accuracy: 0.6956

Epoch 00016: val_loss improved from 0.84535 to 0.78664, saving model to Emoti
on_little_vgg.h5
Epoch 17/25
```

```
755/755 [==============================] - 468s 620ms/step - loss: 1.0310 - a
ccuracy: 0.5960 - val_loss: 0.8247 - val_accuracy: 0.6825

Epoch 00017: val_loss did not improve from 0.78664
Epoch 18/25
755/755 [==============================] - 522s 692ms/step - loss: 1.0346 - a
ccuracy: 0.5975 - val_loss: 0.8774 - val_accuracy: 0.6653

Epoch 00018: val_loss did not improve from 0.78664
Epoch 19/25
755/755 [==============================] - 586s 776ms/step - loss: 1.0213 - a
ccuracy: 0.6064 - val_loss: 0.8038 - val_accuracy: 0.6818
Restoring model weights from the end of the best epoch.

Epoch 00019: val_loss did not improve from 0.78664

Epoch 00019: ReduceLROnPlateau reducing learning rate to 0.000200000009499490
26.
Epoch 00019: early stopping
```

In [3]:
```
face_classifier = cv2.CascadeClassifier(r'C:\Users\suman\Downloads\haarcascade_fr
classifier =load_model(r'C:\Users\suman\Downloads\Emotion_little_vgg.h5')

class_labels = ['Angry','Happy','Neutral','Sad','Surprise']

cap = cv2.VideoCapture(0)
```

In [4]:
```
from PIL import *
```

In [5]:
```
while True:
    # Grab a single frame of video
    ret, frame = cap.read()
    labels = []
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray,1.3,5)

    for (x,y,w,h) in faces:
        cv2.rectangle(frame,(x,y),(x+w,y+h),(255,0,0),2)
        roi_gray = gray[y:y+h,x:x+w]
        roi_gray = cv2.resize(roi_gray,(48,48),interpolation=cv2.INTER_AREA)
    # rect,face,image = face_detector(frame)


        if np.sum([roi_gray])!=0:
            roi = roi_gray.astype('float')/255.0
            roi = img_to_array(roi)
            roi = np.expand_dims(roi,axis=0)

        # make a prediction on the ROI, then lookup the class

            preds = classifier.predict(roi)[0]
            label=class_labels[preds.argmax()]
            label_position = (x,y)
            cv2.putText(frame,label,label_position,cv2.FONT_HERSHEY_SIMPLEX,2,(0,
        else:
            cv2.putText(frame,'No Face Found',(20,60),cv2.FONT_HERSHEY_SIMPLEX,2,
    cv2.imshow('Emotion Detector',frame)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

In [ ]:

In [ ]:

In [ ]: