In [1]:
```
import numpy as np
import cv2
from detectfaces import get_faces
from keras.models import load_model
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, BatchNormalization
from keras.layers import Conv2D, MaxPooling2D, Activation
from keras import backend as K
import matplotlib.pyplot as plt

img_rows, img_cols = 48, 48

# emotion = ['Angry', 'Happy', 'Sad', 'Surprise', 'Neutral']
emotion = ['Angry', "Disgust", 'Fear', 'Happy', 'Sad', 'Surprise', 'Neutral']
font = cv2.FONT_HERSHEY_SIMPLEX

model = []
print('Loading Models...')
print('0/3')
for i in range(2):
    m = load_model('saved_model/' + 'cnn'+str(i)+'.h5')
    print(str(i+1) + '/3')
    model.append(m)

m = load_model('saved_model/ensemble.h5')
model.append(m)
print('3/3')

print("Loading Complete!")

def predict(x):
    x_rev = np.flip(x, 1)
    x = x.astype('float32')
    x_rev = x_rev.astype('float32')
    x /= 255
    x_rev /= 255
    p = np.zeros((1, 14))
    p[:,0:7] = model[0].predict(x.reshape(1,48,48,1))
    p[:,7:14] = model[1].predict(x_rev.reshape(1,48,48,1))
    pre = model[2].predict(p)
    return pre

cap = cv2.VideoCapture(0)
if not cap.isOpened():
    cap.open()

while(True):
    ret, img = cap.read()

    faces = get_faces(img, method='dnn')
    for i,(face,x,y,w,h) in enumerate(faces):
        pre = predict(face)
        k = np.argmax(pre)
        tl=(x,y)
        br=(x+w,y+h)
        img = cv2.rectangle(img, tl, br, (0, 255, 0), 2)
```

```
        txt = emotion[k] + ' [' + str(int(pre[0,k]*100)) + '%]'
        cv2.putText(img, txt, tl, font, 0.5,(0,255,0),1,cv2.LINE_AA)
        # cv2.imshow(str(i), face)

    cv2.imshow('Camera', img)
    if cv2.waitKey(20) & 0xFF == ord('q'):
        break

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

```
Using TensorFlow backend.

Loading Models...
0/3
1/3
2/3
3/3
Loading Complete!

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-1-8233a2094c48> in <module>
     50     faces = get_faces(img, method='dnn')
     51     for i,(face,x,y,w,h) in enumerate(faces):
---> 52         pre = predict(face)
     53         k = np.argmax(pre)
     54         tl=(x,y)

<ipython-input-1-8233a2094c48> in predict(x)
     38     p[:,0:7] = model[0].predict(x.reshape(1,48,48,1))
     39     p[:,7:14] = model[1].predict(x_rev.reshape(1,48,48,1))
---> 40     pre = model[2].predict(p)
     41     return pre
     42

~\Anaconda3\envs\tf-gpu\lib\site-packages\keras\engine\training.py in predict
(self, x, batch_size, verbose, steps)
   1147                                 'argument.')
   1148         # Validate user data.
-> 1149         x, _, _ = self._standardize_user_data(x)
   1150         if self.stateful:
   1151             if x[0].shape[0] > batch_size and x[0].shape[0] % batch_s
ize != 0:

~\Anaconda3\envs\tf-gpu\lib\site-packages\keras\engine\training.py in _standa
rdize_user_data(self, x, y, sample_weight, class_weight, check_array_lengths,
batch_size)
    749             feed_input_shapes,
    750             check_batch_axis=False,  # Don't enforce the batch size.
--> 751             exception_prefix='input')
    752
    753         if y is not None:

~\Anaconda3\envs\tf-gpu\lib\site-packages\keras\engine\training_utils.py in s
tandardize_input_data(data, names, shapes, check_batch_axis, exception_prefi
x)
    126                         ': expected ' + names[i] + ' to have ' +
    127                         str(len(shape)) + ' dimensions, but got array
'
--> 128                         'with shape ' + str(data_shape))
    129                 if not check_batch_axis:
    130                     data_shape = data_shape[1:]

ValueError: Error when checking input: expected batch_normalization_8_input t
o have 4 dimensions, but got array with shape (1, 14)
```