In [1]:
```
import tkinter as tk
from tkinter import *
import cv2
from PIL import Image, ImageTk
import os
import numpy as np
import cv2
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D
from keras.optimizers import Adam
from keras.layers import MaxPooling2D
from keras.preprocessing.image import ImageDataGenerator
import threading
```

In [2]:
```
emotion_model = Sequential()

emotion_model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(48
emotion_model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))

emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))

emotion_model.add(Flatten())
emotion_model.add(Dense(1024, activation='relu'))
emotion_model.add(Dropout(0.5))
emotion_model.add(Dense(7, activation='softmax'))
emotion_model.load_weights('model.h5')
```

In [3]:
```
cv2.ocl.setUseOpenCL(False)

emotion_dict = {0: "Sad", 1: "Disgusted", 2: "Fearful", 3: "Happy", 4: "Neutral", 5

emoji_dist={0:"emojis\\sad.png",1:"emojis\\disgust.png",2:"emojis\\fear.png",3:"emo

global last_frame1
last1_frame1 = np.zeros((448,640,3), dtype = np.uint8)
global cap1
show_text = [0]
global frame_number
```

In [4]:
```
def show_subject():
    cap1 = cv2.VideoCapture(0)
    if not cap1.isOpened():
        print("Can't open the camera")
    global frame_number
    length = int(cap1.get(cv2.CAP_PROP_FRAME_COUNT))
    frame_number += 1
    if frame_number >= length:
        exit()
    cap1.set(1, frame_number)
    flag1, frame1 = cap1.read()
    frame1 = cv2.resize(frame1,(600,500))
    bounding_box = cv2.CascadeClassifier("opencv-master\\opencv-master\\data\\haar
    gray_frame = cv2.cvtColor(frame1, cv2.COLOR_BGR2GRAY)
    num_faces = bounding_box.detectMultiScale(gray_frame, scaleFactor = 1.3, minNe
```

```
    for (x,y,w,h) in num_faces:
        cv2.rectangle(frame1, (x,y-50), (x+w, y+h+10), (255,0,0), 2)
        roi_gray_frame = gray_frame[y:y + h, x:x + w]
        cropped_img = np.expand_dims(np.expand_dims(cv2.resize(roi_gray_frame, (48,
        prediction = emotion_model.predict(cropped_img)
        maxindex = int(np.argmax(prediction))
        cv2.putText(frame1, emotion_dict[maxindex], (x+20, y-60), cv2.FONT_HERSHEY_
        show_text[0] = maxindex

    if flag1 is None:
        print("Major error!")
    elif flag1:
        global last_frame1
        last_frame1 = frame1.copy()
        pic = cv2.cvtColor(last_frame1,cv2.COLOR_BGR2RGB)
        img = Image.fromarray(pic)
        imgtk = ImageTk.PhotoImage(image=img)
        lmain.imgtk = imgtk
        lmain.configure(image = imgtk)
        root.update()
        lmain.after(10,show_subject)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        exit()
```

In [5]:

```
def show_avatar():
    frame2 = cv2.imread(emoji_dist[show_text[0]])
    pic2 = cv2.cvtColor(frame2,cv2.COLOR_BGR2RGB)
    img2 = Image.fromarray(frame2)
    imgtk2 = ImageTk.PhotoImage(image=img2)
    lmain2.imgtk2 = imgtk2
    lmain3.configure(text=emotion_dict[show_text[0]],font=('arial',45,'bold'))

    lmain2.configure(image = imgtk2)
    root.update()
    lmain2.after(10,show_avatar())
```

In [ ]:

```
if __name__ == '__main__':
    frame_number = 0
    root=tk.Tk()
    lmain = tk.Label(master=root,padx=50,bd=10)
    lmain2 = tk.Label(master=root,bd=10)
    lmain3=tk.Label(master=root,bd=10,fg="#CDCDCD",bg='black')
    lmain.pack(side=LEFT)
    lmain.place(x=50,y=250)
    lmain3.pack()
    lmain3.place(x=960,y=250)
    lmain2.pack(side=RIGHT)
    lmain2.place(x=900,y=350)

    root.title("Photo To Emoji")
    root.geometry("1400x900+100+10")
    root['bg']='black'
    exitButton = Button(root, text='Quit',fg="red",command=root.destroy,font=('aria
    threading.Thread(target = show_subject()).start()
    threading.Thread(target = show_avatar()).start()
    root.mainloop()
```

```
1/1 [==============================] - 0s 251ms/step
1/1 [==============================] - 0s 21ms/step
```

In [ ]: