In [2]:

```
import keras
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dense,Dropout,Activation,Flatten,BatchNormalization
from keras.layers import Conv2D,MaxPooling2D
import os
```

In [3]:

```
num_classes = 7
img_rows,img_cols = 48,48
batch_size = 64
```

In [4]:

```
train_data_dir = '/Users/varun/Documents/Deep_Learning/data/train'
validation_data_dir = '/Users/varun/Documents/Deep_Learning/data/test'
```

In [5]:

```
train_datagen = ImageDataGenerator(rescale = 1./255)

validation_datagen = ImageDataGenerator(rescale=1./255)
```

In [6]:

```
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 28709 images belonging to 7 classes.
Found 7178 images belonging to 7 classes.
```

In [7]:

```
model = Sequential()

model.add(Conv2D(32,(3,3),padding='same',kernel_initializer='he_normal',input_shape=(img_rows,img_cols,1)))
model.add(Activation('relu'))
model.add(BatchNormalization())
model.add(Conv2D(32,(3,3),padding='same',kernel_initializer='he_normal',input_shape=(img_rows,img_cols,1)))
model.add(Activation('relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.2))
```

In [8]:

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

In [9]:

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

In [10]:

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

In [11]:
```
model.add(Flatten())
model.add(Dense(64,kernel_initializer='he_normal'))
model.add(Activation('relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
```

In [12]:
```
model.add(Dense(64,kernel_initializer='he_normal'))
model.add(Activation('relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
```

In [13]:
```
model.add(Dense(num_classes,kernel_initializer='he_normal'))
model.add(Activation('softmax'))
```

In [14]:
```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_1 (Conv2D)            (None, 48, 48, 32)        320       
_________________________________________________________________
activation_1 (Activation)    (None, 48, 48, 32)        0         
_________________________________________________________________
batch_normalization_1 (Batch (None, 48, 48, 32)        128       
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 48, 48, 32)        9248      
_________________________________________________________________
activation_2 (Activation)    (None, 48, 48, 32)        0         
_________________________________________________________________
batch_normalization_2 (Batch (None, 48, 48, 32)        128       
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 24, 24, 32)        0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 24, 24, 32)        0         
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 24, 24, 64)        18496     
_________________________________________________________________
activation_3 (Activation)    (None, 24, 24, 64)        0         
_________________________________________________________________
batch_normalization_3 (Batch (None, 24, 24, 64)        256       
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 24, 24, 64)        36928     
_________________________________________________________________
activation_4 (Activation)    (None, 24, 24, 64)        0         
_________________________________________________________________
batch_normalization_4 (Batch (None, 24, 24, 64)        256       
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 12, 12, 64)        0         
_________________________________________________________________
dropout_2 (Dropout)          (None, 12, 12, 64)        0         
```

```
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 12, 12, 128)       73856
_________________________________________________________________
activation_5 (Activation)    (None, 12, 12, 128)       0
_________________________________________________________________
batch_normalization_5 (Batch (None, 12, 12, 128)       512
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 12, 12, 128)       147584
_________________________________________________________________
activation_6 (Activation)    (None, 12, 12, 128)       0
_________________________________________________________________
batch_normalization_6 (Batch (None, 12, 12, 128)       512
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 6, 6, 128)         0
_________________________________________________________________
dropout_3 (Dropout)          (None, 6, 6, 128)         0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 6, 6, 256)         295168
_________________________________________________________________
activation_7 (Activation)    (None, 6, 6, 256)         0
_________________________________________________________________
batch_normalization_7 (Batch (None, 6, 6, 256)         1024
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 6, 6, 256)         590080
_________________________________________________________________
activation_8 (Activation)    (None, 6, 6, 256)         0
_________________________________________________________________
batch_normalization_8 (Batch (None, 6, 6, 256)         1024
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 3, 3, 256)         0
_________________________________________________________________
dropout_4 (Dropout)          (None, 3, 3, 256)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 2304)              0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                147520
_________________________________________________________________
activation_9 (Activation)    (None, 64)                0
_________________________________________________________________
batch_normalization_9 (Batch (None, 64)                256
_________________________________________________________________
dropout_5 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                4160
_________________________________________________________________
activation_10 (Activation)   (None, 64)                0
_________________________________________________________________
batch_normalization_10 (Batc (None, 64)                256
_________________________________________________________________
dropout_6 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 7)                 455
_________________________________________________________________
```

```
activation_11 (Activation)   (None, 7)                 0
=================================================================
Total params: 1,328,167
Trainable params: 1,325,991
Non-trainable params: 2,176

_________________________________________________________________
```

In [17]:

```
from keras.optimizers import RMSprop

model.compile(loss='categorical_crossentropy',
              optimizer = RMSprop(lr=0.001),
              metrics=['accuracy'])
```

In [18]:

```
nb_train_samples = train_generator.samples
nb_validation_samples = validation_generator.samples
epochs=25

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

```
Epoch 1/25
448/448 [==============================] - 955s 2s/step - loss: 1.
4669 - accuracy: 0.4346 - val_loss: 1.3395 - val_accuracy: 0.4926
Epoch 2/25
448/448 [==============================] - 883s 2s/step - loss: 1.
3036 - accuracy: 0.5072 - val_loss: 1.0971 - val_accuracy: 0.5371
Epoch 3/25
448/448 [==============================] - 566s 1s/step - loss: 1.
2160 - accuracy: 0.5447 - val_loss: 1.3655 - val_accuracy: 0.5526
Epoch 4/25
448/448 [==============================] - 553s 1s/step - loss: 1.
1525 - accuracy: 0.5749 - val_loss: 1.0365 - val_accuracy: 0.5783
Epoch 5/25
448/448 [==============================] - 581s 1s/step - loss: 1.
0892 - accuracy: 0.6013 - val_loss: 0.9968 - val_accuracy: 0.5932
Epoch 6/25
448/448 [==============================] - 553s 1s/step - loss: 1.
0240 - accuracy: 0.6278 - val_loss: 0.8856 - val_accuracy: 0.6067
Epoch 7/25
448/448 [==============================] - 558s 1s/step - loss: 0.
9675 - accuracy: 0.6496 - val_loss: 1.2115 - val_accuracy: 0.6023
Epoch 8/25
448/448 [==============================] - 607s 1s/step - loss: 0.
9165 - accuracy: 0.6723 - val_loss: 0.8414 - val_accuracy: 0.6134
Epoch 9/25
448/448 [==============================] - 630s 1s/step - loss: 0.
8656 - accuracy: 0.6942 - val_loss: 1.0902 - val_accuracy: 0.6143
Epoch 10/25
```

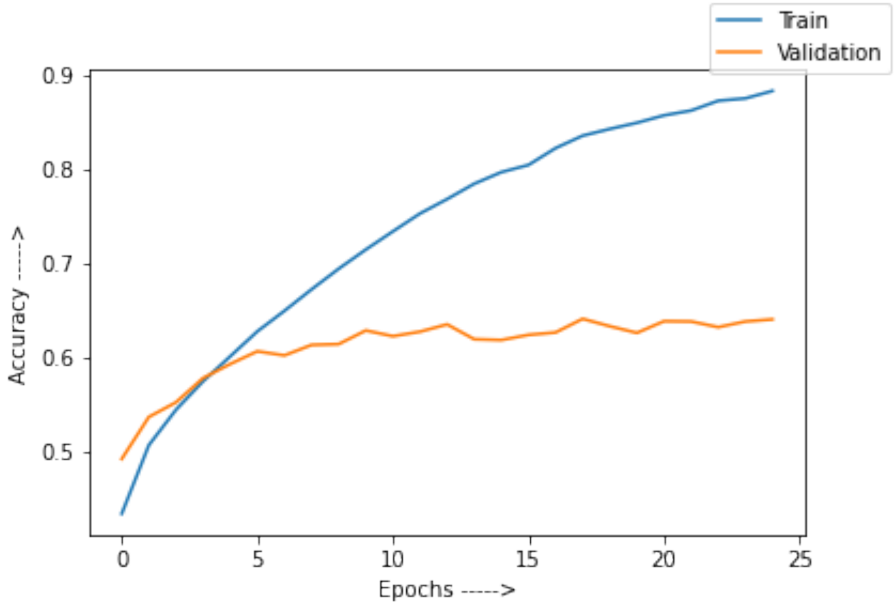
```
448/448 [==============================] - 563s 1s/step - loss: 0.
8154 - accuracy: 0.7145 - val_loss: 1.0121 - val_accuracy: 0.6288
Epoch 11/25
448/448 [==============================] - 560s 1s/step - loss: 0.
7702 - accuracy: 0.7338 - val_loss: 1.2011 - val_accuracy: 0.6227
Epoch 12/25
448/448 [==============================] - 554s 1s/step - loss: 0.
7284 - accuracy: 0.7528 - val_loss: 1.0205 - val_accuracy: 0.6275
Epoch 13/25
448/448 [==============================] - 552s 1s/step - loss: 0.
6765 - accuracy: 0.7681 - val_loss: 1.0300 - val_accuracy: 0.6352
Epoch 14/25
448/448 [==============================] - 552s 1s/step - loss: 0.
6369 - accuracy: 0.7843 - val_loss: 1.2528 - val_accuracy: 0.6195
Epoch 15/25
448/448 [==============================] - 551s 1s/step - loss: 0.
5975 - accuracy: 0.7966 - val_loss: 1.1270 - val_accuracy: 0.6186
Epoch 16/25
448/448 [==============================] - 551s 1s/step - loss: 0.
5721 - accuracy: 0.8042 - val_loss: 0.9713 - val_accuracy: 0.6241
Epoch 17/25
448/448 [==============================] - 570s 1s/step - loss: 0.
5333 - accuracy: 0.8221 - val_loss: 1.3069 - val_accuracy: 0.6267
Epoch 18/25
448/448 [==============================] - 562s 1s/step - loss: 0.
4976 - accuracy: 0.8353 - val_loss: 1.0557 - val_accuracy: 0.6411
Epoch 19/25
448/448 [==============================] - 558s 1s/step - loss: 0.
4805 - accuracy: 0.8424 - val_loss: 1.1246 - val_accuracy: 0.6333
Epoch 20/25
448/448 [==============================] - 599s 1s/step - loss: 0.
4557 - accuracy: 0.8489 - val_loss: 1.0690 - val_accuracy: 0.6262
Epoch 21/25
448/448 [==============================] - 610s 1s/step - loss: 0.
4307 - accuracy: 0.8568 - val_loss: 1.7513 - val_accuracy: 0.6386
Epoch 22/25
448/448 [==============================] - 560s 1s/step - loss: 0.
4138 - accuracy: 0.8619 - val_loss: 1.3285 - val_accuracy: 0.6383
Epoch 23/25
448/448 [==============================] - 554s 1s/step - loss: 0.
3993 - accuracy: 0.8723 - val_loss: 1.2025 - val_accuracy: 0.6323
Epoch 24/25
448/448 [==============================] - 551s 1s/step - loss: 0.
3729 - accuracy: 0.8748 - val_loss: 1.2708 - val_accuracy: 0.6383
Epoch 25/25
448/448 [==============================] - 549s 1s/step - loss: 0.
3650 - accuracy: 0.8828 - val_loss: 1.3415 - val_accuracy: 0.6406
```

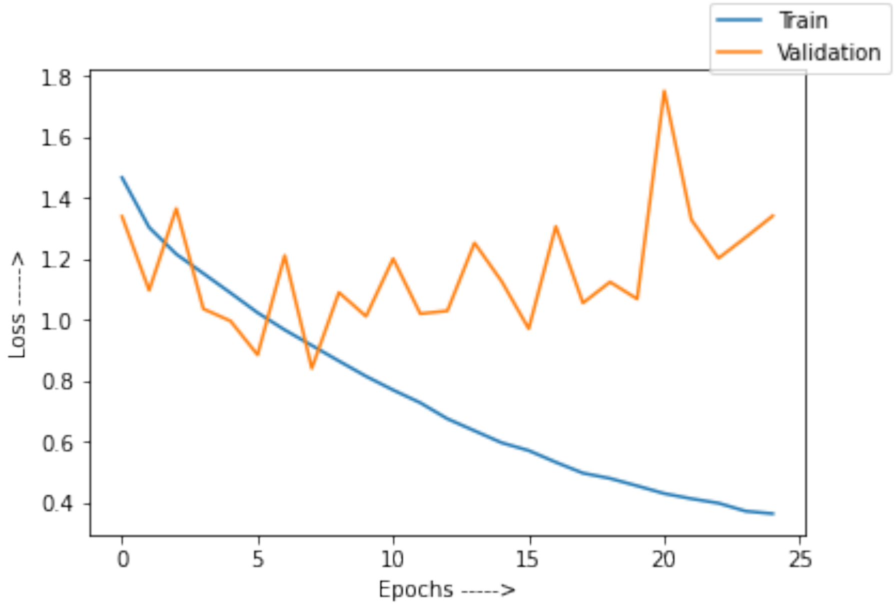
In [19]:

```
model.save('emorecgwoaug.h5')
```

In [21]:

```
from matplotlib import pyplot as plt

acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']


ax1 = plt.figure(0)
plt.plot(acc,label = 'Train')
plt.plot(val_acc, label = 'Validation')
plt.xlabel('Epochs ----->')
plt.ylabel('Accuracy ----->')
leg = ax1.legend()


ax2 = plt.figure(1)
plt.plot(loss,label = 'Train')
plt.plot(val_loss,label = 'Validation')
plt.xlabel('Epochs ----->')
plt.ylabel('Loss ----->')
leg = ax2.legend()
```





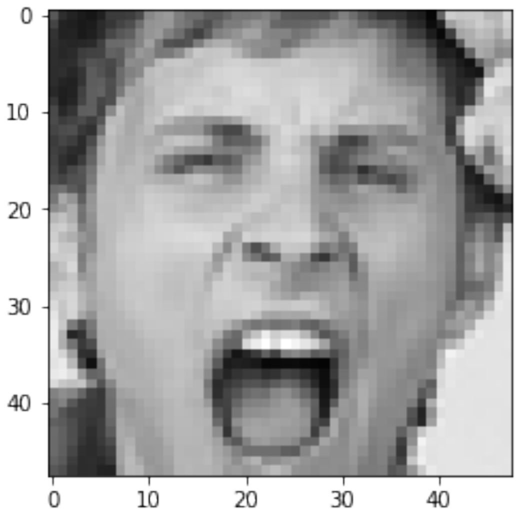
```
In [32]: import cv2

         import numpy as np

         import tensorflow as tf
```

```
In [23]: classes = ['angry','disguisted','fearful','happy','neutral','sad','suprised']
```

In [69]:

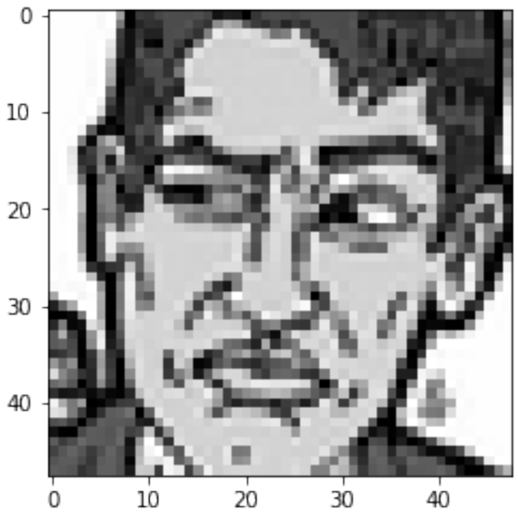
```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/angry/im3.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: angry
```



```
In [70]: I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/angry/im0.png')

         ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

         I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


         IG = tf.cast(I1, tf.float32)

         plt.imshow(ID)

         print("IG shape: " + str(IG.shape))

         IG = IG/255

         IGP = np.expand_dims(IG,axis = 0)

         IGP = np.expand_dims(IGP,axis = 3)

         print("IGP Shape:" + str(IGP.shape))

         predictions = model.predict(IGP)

         index = np.argmax(predictions)

         print("Predicted Emotion is: " + str(classes[index]))
```

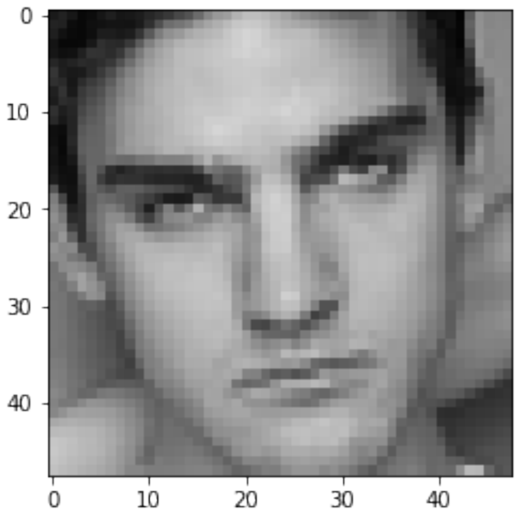
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: angry
```



In [71]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/angry/im10.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

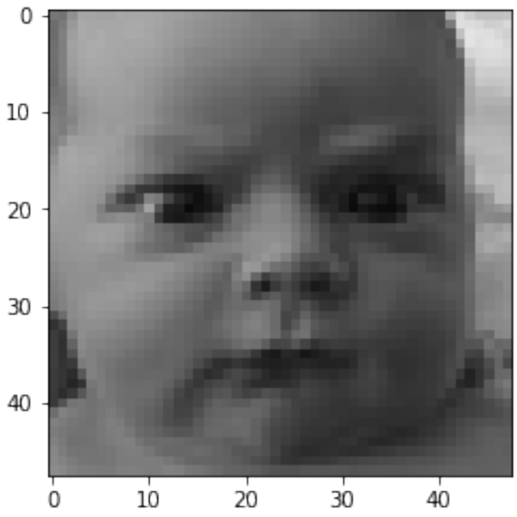
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: angry
```



In [72]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/angry/im123.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

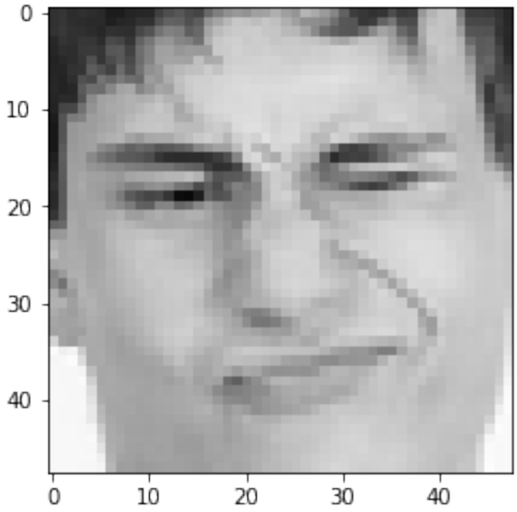
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: angry
```



In [78]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/disgusted/im5.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

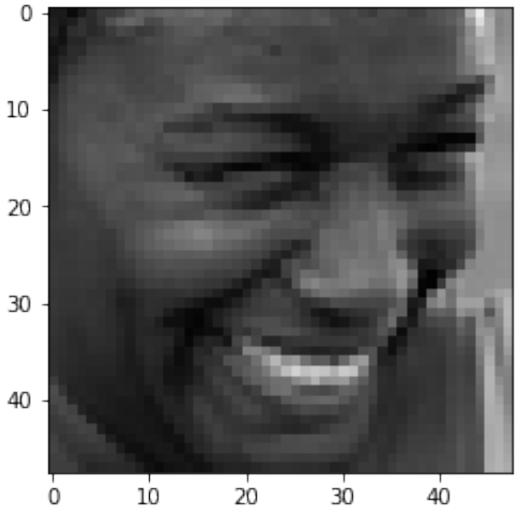
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: disguisted
```



In [79]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/happy/im5.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

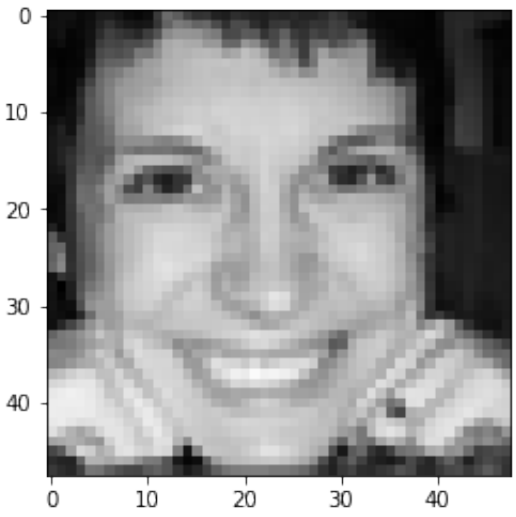
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: happy
```



In [80]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/happy/im516.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

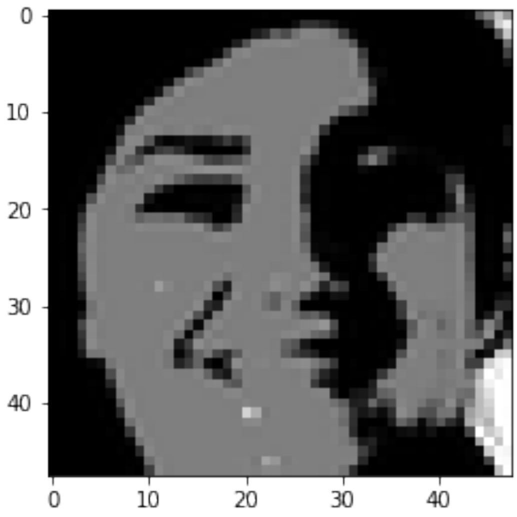
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: happy
```



In [81]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/happy/im312.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

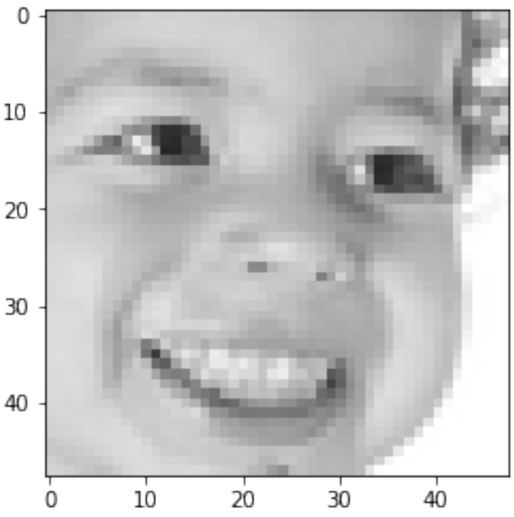
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: happy
```



In [82]:

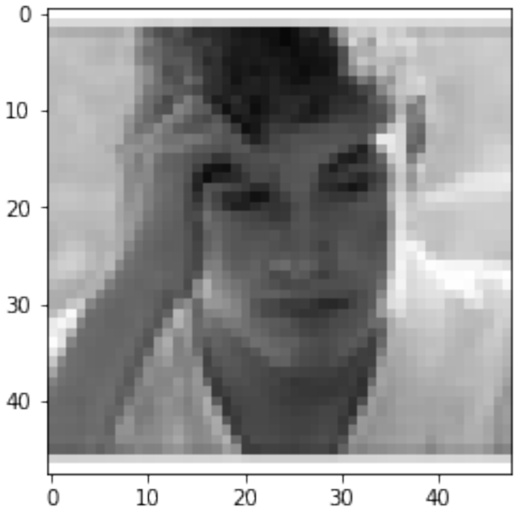
```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/happy/im500.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: happy
```



In [83]:
```
#Failed


I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/fearful/im1.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

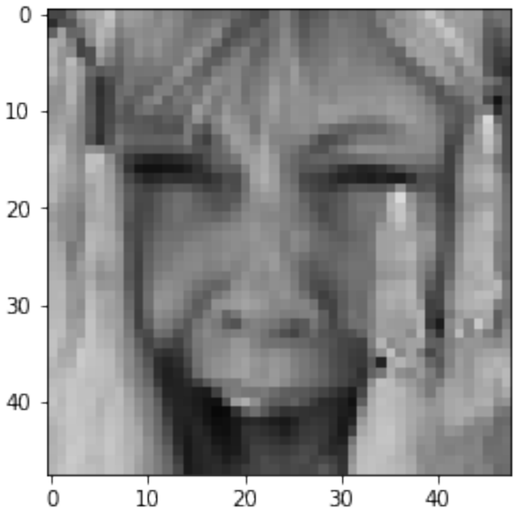
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: sad
```



In [84]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/fea
rful/im0.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

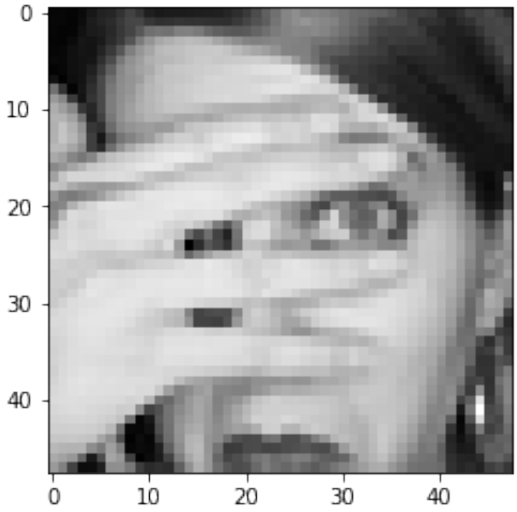
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: fearful
```



In [85]:

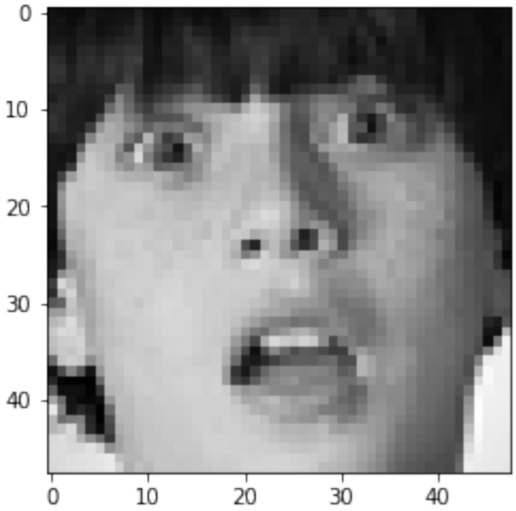
```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/fearful/im10.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: fearful
```



In [86]:
```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/fearful/im24.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

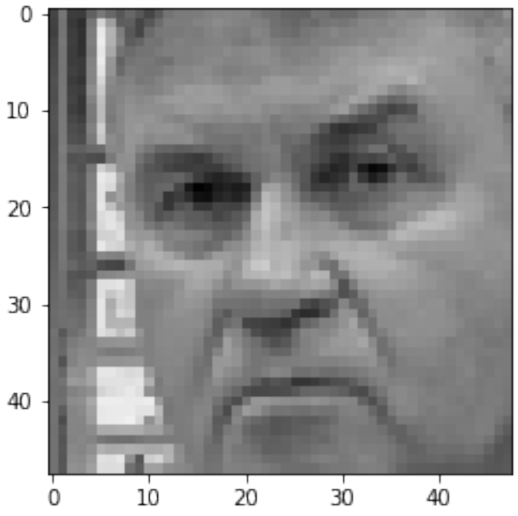
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: fearful
```



In [88]:

```
#Failed


I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/sad
/im0.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

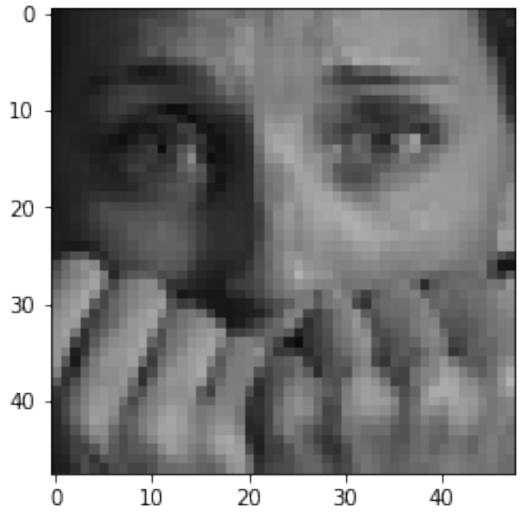
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: angry
```



In [90]:

```
#Failed


I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/sad
/im10.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

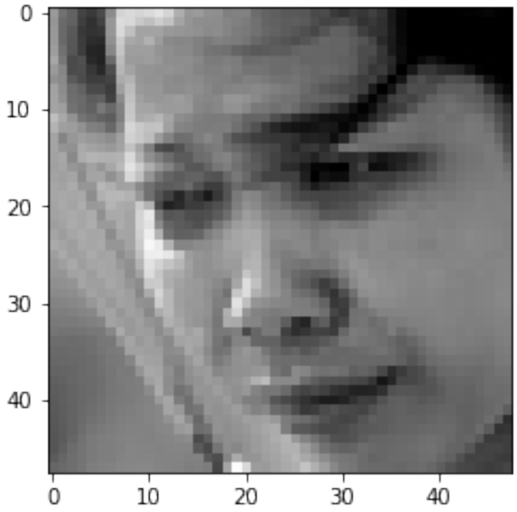
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: fearful
```



In [92]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/sad/im100.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

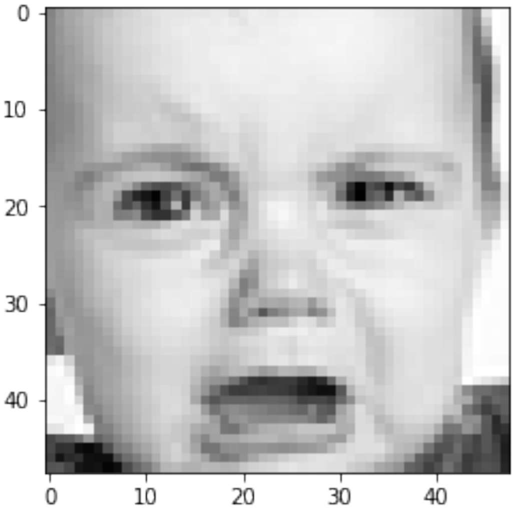
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: sad
```



In [93]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/sad/im20.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

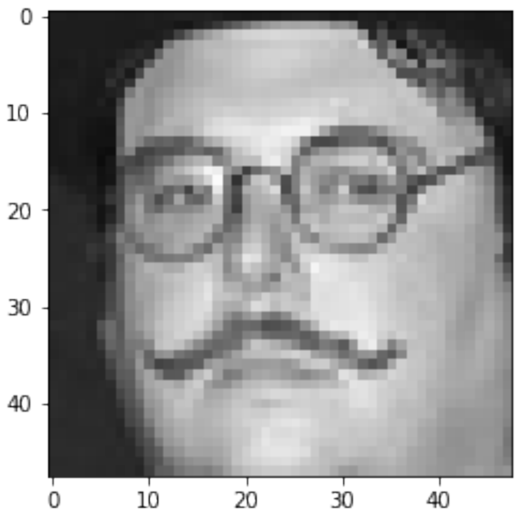
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: sad
```



In [94]:

```
#Failed

I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/sad
/im110.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

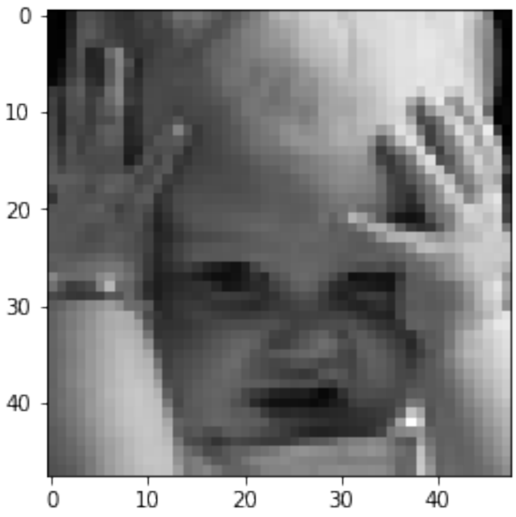
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: neutral
```



In [95]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/sad/im40.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

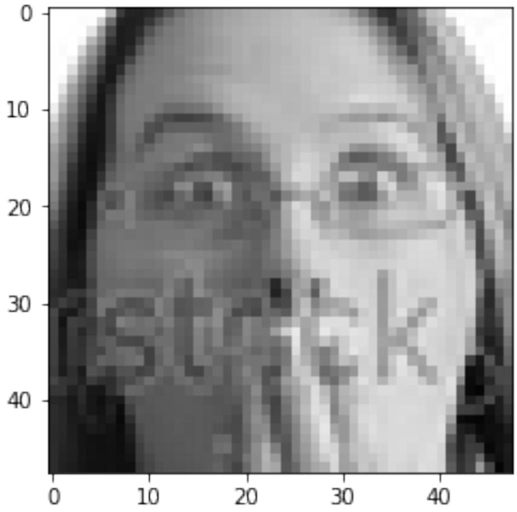
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: sad
```



In [96]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/surprised/im0.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

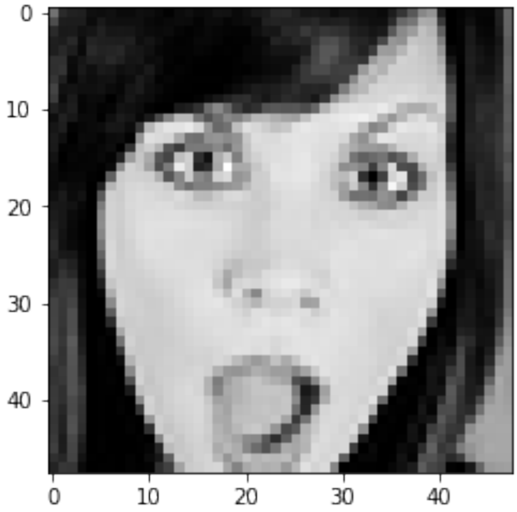
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: suprised
```



In [97]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/surprised/im10.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

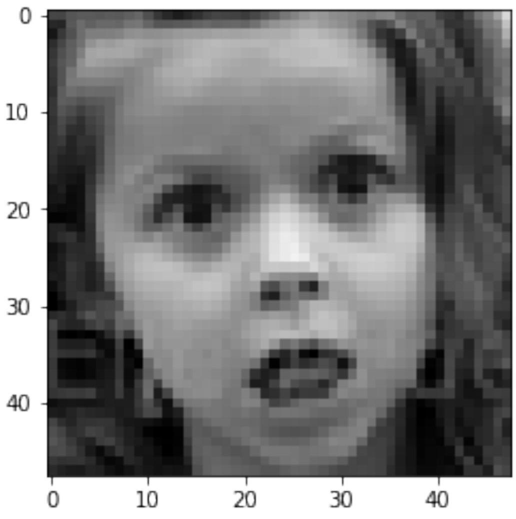
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: suprised
```



In [98]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/surprised/im100.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

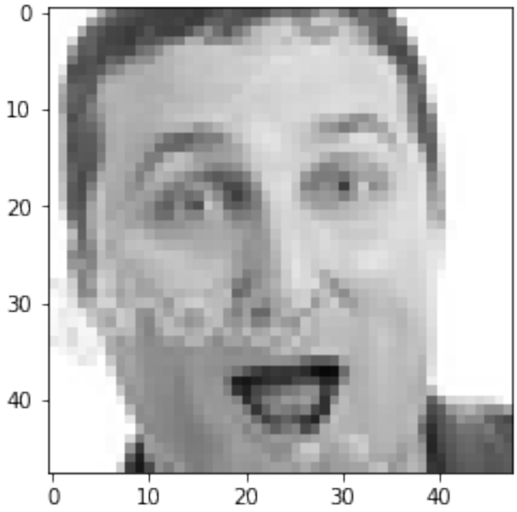
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: suprised
```



In [99]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/surprised/im20.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

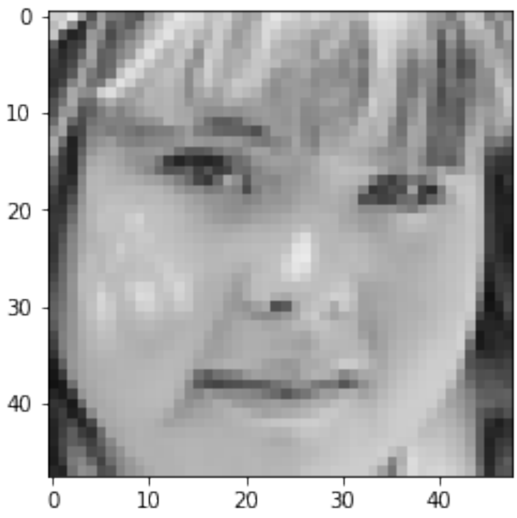
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: suprised
```



In [101]:

```python
#Failed

I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/neutral/im0.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

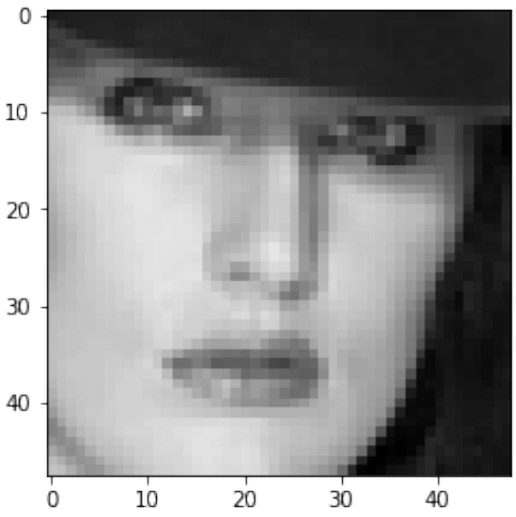
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: fearful
```



In [103]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/neutral/im100.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

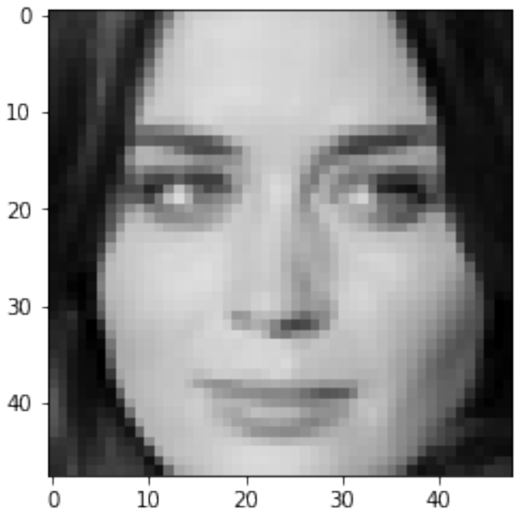
```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: neutral
```



In [104]:

```
I1 = cv2.imread('/Users/varun/Documents/Deep_Learning/data/test/neutral/im20.png')

ID = cv2.cvtColor(I1,cv2.COLOR_BGR2RGB)

I1 = cv2.cvtColor(I1,cv2.COLOR_BGR2GRAY)


IG = tf.cast(I1, tf.float32)

plt.imshow(ID)

print("IG shape: " + str(IG.shape))

IG = IG/255

IGP = np.expand_dims(IG,axis = 0)

IGP = np.expand_dims(IGP,axis = 3)

print("IGP Shape:" + str(IGP.shape))

predictions = model.predict(IGP)

index = np.argmax(predictions)

print("Predicted Emotion is: " + str(classes[index]))
```

```
IG shape: (48, 48)
IGP Shape:(1, 48, 48, 1)
Predicted Emotion is: neutral
```



In [ ]: