test

April 21, 2024

[13]:
```
import os
import pandas as pd
import numpy as np
from tqdm.notebook import tqdm
from keras.models import Sequential, model_from_json
from keras.layers import Dense, Conv2D, Dropout, Flatten, MaxPooling2D
from keras.utils import to_categorical
from sklearn.preprocessing import LabelEncoder
from tensorflow.keras.preprocessing.image import img_to_array, load_img  #␣
 ↪Corrected import
from PIL import Image  # Importing Image from PIL for resizing
import matplotlib.pyplot as plt
```

[14]:
```
TRAIN_DIR = 'images/train'
TEST_DIR = 'images/test'
```

[15]:
```
def createdataframe(dir):
    image_paths = []
    labels = []
    for label in os.listdir(dir):
        for imagename in os.listdir(os.path.join(dir, label)):
            image_paths.append(os.path.join(dir, label, imagename))
            labels.append(label)
        print(label, "completed")
    return image_paths, labels
```

[16]:
```
def extract_features(images):
    features = []
    for image in tqdm(images):
        img = load_img(image, grayscale=True)
        img = img.resize((48, 48))  # Resize the image to a consistent size
        img = np.array(img)
        features.append(img)
    features = np.array(features)
    features = features.reshape(len(features), 48, 48, 1)
    return features
```

[17]:
```
train = pd.DataFrame()
train['image'], train['label'] = createdataframe(TRAIN_DIR)
```

```
angry completed
disgust completed
fear completed
happy completed
neutral completed
sad completed
surprise completed
```

[18]:
```
test = pd.DataFrame()
test['image'], test['label'] = createdataframe(TEST_DIR)
```

```
angry completed
disgust completed
fear completed
happy completed
neutral completed
sad completed
surprise completed
```

[19]:
```
train_features = extract_features(train['image'])
```

```
  0%|          | 0/28823 [00:00<?, ?it/s]
c:\Users\basum\anaconda3\envs\pythongpu\lib\site-
packages\keras\utils\image_utils.py:409: UserWarning: grayscale is deprecated.
Please use color_mode = "grayscale"
  warnings.warn(
```

[20]:
```
test_features = extract_features(test['image'])
```

```
  0%|          | 0/7068 [00:00<?, ?it/s]
```

[21]:
```
x_train = train_features/255.0
x_test = test_features/255.0
```

[22]:
```
le = LabelEncoder()
le.fit(train['label'])
```

[22]:
```
LabelEncoder()
```

[23]:
```
y_train = le.transform(train['label'])
y_test = le.transform(test['label'])
```

[24]:
```
y_train = to_categorical(y_train,num_classes = 7)
y_test = to_categorical(y_test,num_classes = 7)
```

[25]:
```
model = Sequential()
# convolutional layers
model.add(Conv2D(128, kernel_size=(3,3), activation='relu',␣
 ↪input_shape=(48,48,1)))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.4))

model.add(Conv2D(256, kernel_size=(3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.4))

model.add(Conv2D(512, kernel_size=(3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.4))

model.add(Conv2D(512, kernel_size=(3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.4))

model.add(Flatten())
# fully connected layers
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.3))
# output layer
model.add(Dense(7, activation='softmax'))
```

[26]:
```
model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics =␣
 ↪'accuracy' )
```

[27]:
```
model.fit(x= x_train,y = y_train, batch_size = 128, epochs = 100,␣
 ↪validation_data = (x_test,y_test))
```

```
Epoch 1/100
226/226 [==============================] - 12s 42ms/step - loss: 1.8237 -
accuracy: 0.2419 - val_loss: 1.8068 - val_accuracy: 0.2585
Epoch 2/100
226/226 [==============================] - 8s 38ms/step - loss: 1.7768 -
accuracy: 0.2596 - val_loss: 1.7117 - val_accuracy: 0.3035
Epoch 3/100
226/226 [==============================] - 8s 37ms/step - loss: 1.6624 -
accuracy: 0.3301 - val_loss: 1.5013 - val_accuracy: 0.4199
Epoch 4/100
226/226 [==============================] - 8s 37ms/step - loss: 1.5243 -
accuracy: 0.4062 - val_loss: 1.3713 - val_accuracy: 0.4733
Epoch 5/100
```

```
226/226 [==============================] - 8s 37ms/step - loss: 1.4539 -
accuracy: 0.4384 - val_loss: 1.3329 - val_accuracy: 0.4901
Epoch 6/100
226/226 [==============================] - 9s 38ms/step - loss: 1.4025 -
accuracy: 0.4612 - val_loss: 1.3039 - val_accuracy: 0.5045
Epoch 7/100
226/226 [==============================] - 8s 38ms/step - loss: 1.3636 -
accuracy: 0.4746 - val_loss: 1.2584 - val_accuracy: 0.5204
Epoch 8/100
226/226 [==============================] - 9s 38ms/step - loss: 1.3308 -
accuracy: 0.4916 - val_loss: 1.2209 - val_accuracy: 0.5334
Epoch 9/100
226/226 [==============================] - 9s 38ms/step - loss: 1.3010 -
accuracy: 0.5003 - val_loss: 1.2335 - val_accuracy: 0.5307
Epoch 10/100
226/226 [==============================] - 9s 38ms/step - loss: 1.2914 -
accuracy: 0.5054 - val_loss: 1.2097 - val_accuracy: 0.5405
Epoch 11/100
226/226 [==============================] - 9s 38ms/step - loss: 1.2696 -
accuracy: 0.5154 - val_loss: 1.1882 - val_accuracy: 0.5426
Epoch 12/100
226/226 [==============================] - 9s 38ms/step - loss: 1.2454 -
accuracy: 0.5242 - val_loss: 1.1527 - val_accuracy: 0.5584
Epoch 13/100
226/226 [==============================] - 9s 38ms/step - loss: 1.2369 -
accuracy: 0.5307 - val_loss: 1.1691 - val_accuracy: 0.5637
Epoch 14/100
226/226 [==============================] - 9s 38ms/step - loss: 1.2143 -
accuracy: 0.5385 - val_loss: 1.1294 - val_accuracy: 0.5727
Epoch 15/100
226/226 [==============================] - 9s 38ms/step - loss: 1.1989 -
accuracy: 0.5453 - val_loss: 1.1360 - val_accuracy: 0.5772
Epoch 16/100
226/226 [==============================] - 9s 38ms/step - loss: 1.1926 -
accuracy: 0.5451 - val_loss: 1.1219 - val_accuracy: 0.5792
Epoch 17/100
226/226 [==============================] - 9s 38ms/step - loss: 1.1840 -
accuracy: 0.5509 - val_loss: 1.1289 - val_accuracy: 0.5795
Epoch 18/100
226/226 [==============================] - 9s 38ms/step - loss: 1.1707 -
accuracy: 0.5538 - val_loss: 1.1287 - val_accuracy: 0.5720
Epoch 19/100
226/226 [==============================] - 9s 38ms/step - loss: 1.1568 -
accuracy: 0.5571 - val_loss: 1.1193 - val_accuracy: 0.5770
Epoch 20/100
226/226 [==============================] - 9s 38ms/step - loss: 1.1552 -
accuracy: 0.5595 - val_loss: 1.1029 - val_accuracy: 0.5847
Epoch 21/100
```

```
226/226 [==============================] - 9s 38ms/step - loss: 1.1379 -
accuracy: 0.5694 - val_loss: 1.0873 - val_accuracy: 0.5866
Epoch 22/100
226/226 [==============================] - 9s 38ms/step - loss: 1.1339 -
accuracy: 0.5684 - val_loss: 1.0953 - val_accuracy: 0.5855
Epoch 23/100
226/226 [==============================] - 9s 38ms/step - loss: 1.1281 -
accuracy: 0.5720 - val_loss: 1.0917 - val_accuracy: 0.5949
Epoch 24/100
226/226 [==============================] - 9s 38ms/step - loss: 1.1125 -
accuracy: 0.5801 - val_loss: 1.0871 - val_accuracy: 0.5988
Epoch 25/100
226/226 [==============================] - 9s 38ms/step - loss: 1.1116 -
accuracy: 0.5787 - val_loss: 1.0809 - val_accuracy: 0.5934
Epoch 26/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0979 -
accuracy: 0.5826 - val_loss: 1.0736 - val_accuracy: 0.6024
Epoch 27/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0904 -
accuracy: 0.5853 - val_loss: 1.0817 - val_accuracy: 0.5915
Epoch 28/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0850 -
accuracy: 0.5900 - val_loss: 1.0789 - val_accuracy: 0.6037
Epoch 29/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0722 -
accuracy: 0.5962 - val_loss: 1.0650 - val_accuracy: 0.6006
Epoch 30/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0638 -
accuracy: 0.5962 - val_loss: 1.0710 - val_accuracy: 0.6002
Epoch 31/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0615 -
accuracy: 0.5971 - val_loss: 1.0645 - val_accuracy: 0.6036
Epoch 32/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0528 -
accuracy: 0.6054 - val_loss: 1.0704 - val_accuracy: 0.6009
Epoch 33/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0464 -
accuracy: 0.6046 - val_loss: 1.0506 - val_accuracy: 0.6121
Epoch 34/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0399 -
accuracy: 0.6080 - val_loss: 1.0521 - val_accuracy: 0.6068
Epoch 35/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0359 -
accuracy: 0.6117 - val_loss: 1.0447 - val_accuracy: 0.6096
Epoch 36/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0236 -
accuracy: 0.6156 - val_loss: 1.0453 - val_accuracy: 0.6132
Epoch 37/100
```

```
226/226 [==============================] - 9s 38ms/step - loss: 1.0249 -
accuracy: 0.6120 - val_loss: 1.0543 - val_accuracy: 0.6071
Epoch 38/100
226/226 [==============================] - 9s 39ms/step - loss: 1.0144 -
accuracy: 0.6163 - val_loss: 1.0533 - val_accuracy: 0.6166
Epoch 39/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0071 -
accuracy: 0.6187 - val_loss: 1.0518 - val_accuracy: 0.6159
Epoch 40/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0037 -
accuracy: 0.6203 - val_loss: 1.0425 - val_accuracy: 0.6173
Epoch 41/100
226/226 [==============================] - 9s 38ms/step - loss: 1.0029 -
accuracy: 0.6243 - val_loss: 1.0477 - val_accuracy: 0.6135
Epoch 42/100
226/226 [==============================] - 9s 38ms/step - loss: 0.9909 -
accuracy: 0.6239 - val_loss: 1.0413 - val_accuracy: 0.6207
Epoch 43/100
226/226 [==============================] - 9s 38ms/step - loss: 0.9850 -
accuracy: 0.6314 - val_loss: 1.0459 - val_accuracy: 0.6146
Epoch 44/100
226/226 [==============================] - 9s 38ms/step - loss: 0.9822 -
accuracy: 0.6331 - val_loss: 1.0530 - val_accuracy: 0.6128
Epoch 45/100
226/226 [==============================] - 9s 38ms/step - loss: 0.9732 -
accuracy: 0.6358 - val_loss: 1.0473 - val_accuracy: 0.6159
Epoch 46/100
226/226 [==============================] - 9s 38ms/step - loss: 0.9712 -
accuracy: 0.6312 - val_loss: 1.0458 - val_accuracy: 0.6139
Epoch 47/100
226/226 [==============================] - 9s 38ms/step - loss: 0.9679 -
accuracy: 0.6376 - val_loss: 1.0447 - val_accuracy: 0.6173
Epoch 48/100
226/226 [==============================] - 10s 45ms/step - loss: 0.9585 -
accuracy: 0.6401 - val_loss: 1.0457 - val_accuracy: 0.6143
Epoch 49/100
226/226 [==============================] - 10s 46ms/step - loss: 0.9573 -
accuracy: 0.6420 - val_loss: 1.0470 - val_accuracy: 0.6121
Epoch 50/100
226/226 [==============================] - 10s 46ms/step - loss: 0.9534 -
accuracy: 0.6410 - val_loss: 1.0373 - val_accuracy: 0.6222
Epoch 51/100
226/226 [==============================] - 10s 46ms/step - loss: 0.9402 -
accuracy: 0.6451 - val_loss: 1.0488 - val_accuracy: 0.6244
Epoch 52/100
226/226 [==============================] - 10s 46ms/step - loss: 0.9494 -
accuracy: 0.6447 - val_loss: 1.0501 - val_accuracy: 0.6197
Epoch 53/100
```

```
226/226 [==============================] - 10s 46ms/step - loss: 0.9242 -
accuracy: 0.6557 - val_loss: 1.0393 - val_accuracy: 0.6217
Epoch 54/100
226/226 [==============================] - 10s 46ms/step - loss: 0.9264 -
accuracy: 0.6538 - val_loss: 1.0373 - val_accuracy: 0.6242
Epoch 55/100
226/226 [==============================] - 10s 46ms/step - loss: 0.9245 -
accuracy: 0.6545 - val_loss: 1.0365 - val_accuracy: 0.6238
Epoch 56/100
226/226 [==============================] - 10s 46ms/step - loss: 0.9187 -
accuracy: 0.6563 - val_loss: 1.0322 - val_accuracy: 0.6262
Epoch 57/100
226/226 [==============================] - 10s 46ms/step - loss: 0.9073 -
accuracy: 0.6616 - val_loss: 1.0267 - val_accuracy: 0.6246
Epoch 58/100
226/226 [==============================] - 10s 45ms/step - loss: 0.9072 -
accuracy: 0.6620 - val_loss: 1.0308 - val_accuracy: 0.6231
Epoch 59/100
226/226 [==============================] - 9s 38ms/step - loss: 0.9016 -
accuracy: 0.6643 - val_loss: 1.0269 - val_accuracy: 0.6283
Epoch 60/100
226/226 [==============================] - 9s 38ms/step - loss: 0.9009 -
accuracy: 0.6651 - val_loss: 1.0256 - val_accuracy: 0.6299
Epoch 61/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8996 -
accuracy: 0.6633 - val_loss: 1.0368 - val_accuracy: 0.6213
Epoch 62/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8807 -
accuracy: 0.6686 - val_loss: 1.0356 - val_accuracy: 0.6254
Epoch 63/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8838 -
accuracy: 0.6693 - val_loss: 1.0442 - val_accuracy: 0.6228
Epoch 64/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8789 -
accuracy: 0.6730 - val_loss: 1.0246 - val_accuracy: 0.6272
Epoch 65/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8761 -
accuracy: 0.6745 - val_loss: 1.0349 - val_accuracy: 0.6179
Epoch 66/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8788 -
accuracy: 0.6711 - val_loss: 1.0358 - val_accuracy: 0.6235
Epoch 67/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8671 -
accuracy: 0.6773 - val_loss: 1.0260 - val_accuracy: 0.6296
Epoch 68/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8631 -
accuracy: 0.6787 - val_loss: 1.0242 - val_accuracy: 0.6312
Epoch 69/100
```

```
226/226 [==============================] - 9s 38ms/step - loss: 0.8558 -
accuracy: 0.6816 - val_loss: 1.0223 - val_accuracy: 0.6299
Epoch 70/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8519 -
accuracy: 0.6841 - val_loss: 1.0439 - val_accuracy: 0.6317
Epoch 71/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8580 -
accuracy: 0.6814 - val_loss: 1.0325 - val_accuracy: 0.6314
Epoch 72/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8434 -
accuracy: 0.6865 - val_loss: 1.0334 - val_accuracy: 0.6347
Epoch 73/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8385 -
accuracy: 0.6883 - val_loss: 1.0219 - val_accuracy: 0.6370
Epoch 74/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8361 -
accuracy: 0.6904 - val_loss: 1.0215 - val_accuracy: 0.6297
Epoch 75/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8289 -
accuracy: 0.6921 - val_loss: 1.0264 - val_accuracy: 0.6351
Epoch 76/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8291 -
accuracy: 0.6942 - val_loss: 1.0200 - val_accuracy: 0.6321
Epoch 77/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8229 -
accuracy: 0.6959 - val_loss: 1.0242 - val_accuracy: 0.6268
Epoch 78/100
226/226 [==============================] - 9s 38ms/step - loss: 0.8233 -
accuracy: 0.6949 - val_loss: 1.0212 - val_accuracy: 0.6341
Epoch 79/100
226/226 [==============================] - 9s 39ms/step - loss: 0.8209 -
accuracy: 0.6967 - val_loss: 1.0347 - val_accuracy: 0.6302
Epoch 80/100
226/226 [==============================] - 10s 46ms/step - loss: 0.8053 -
accuracy: 0.7018 - val_loss: 1.0358 - val_accuracy: 0.6329
Epoch 81/100
226/226 [==============================] - 10s 46ms/step - loss: 0.8027 -
accuracy: 0.7035 - val_loss: 1.0283 - val_accuracy: 0.6405
Epoch 82/100
226/226 [==============================] - 10s 46ms/step - loss: 0.8085 -
accuracy: 0.7021 - val_loss: 1.0247 - val_accuracy: 0.6346
Epoch 83/100
226/226 [==============================] - 10s 46ms/step - loss: 0.8002 -
accuracy: 0.7040 - val_loss: 1.0432 - val_accuracy: 0.6337
Epoch 84/100
226/226 [==============================] - 10s 46ms/step - loss: 0.8045 -
accuracy: 0.7044 - val_loss: 1.0284 - val_accuracy: 0.6343
Epoch 85/100
```

```
226/226 [==============================] - 10s 46ms/step - loss: 0.7921 -
accuracy: 0.7063 - val_loss: 1.0295 - val_accuracy: 0.6391
Epoch 86/100
226/226 [==============================] - 10s 46ms/step - loss: 0.7896 -
accuracy: 0.7092 - val_loss: 1.0248 - val_accuracy: 0.6353
Epoch 87/100
226/226 [==============================] - 10s 46ms/step - loss: 0.7889 -
accuracy: 0.7097 - val_loss: 1.0232 - val_accuracy: 0.6347
Epoch 88/100
226/226 [==============================] - 10s 46ms/step - loss: 0.7871 -
accuracy: 0.7114 - val_loss: 1.0193 - val_accuracy: 0.6418
Epoch 89/100
226/226 [==============================] - 10s 46ms/step - loss: 0.7863 -
accuracy: 0.7087 - val_loss: 1.0233 - val_accuracy: 0.6430
Epoch 90/100
226/226 [==============================] - 10s 42ms/step - loss: 0.7836 -
accuracy: 0.7119 - val_loss: 1.0274 - val_accuracy: 0.6408
Epoch 91/100
226/226 [==============================] - 9s 38ms/step - loss: 0.7818 -
accuracy: 0.7157 - val_loss: 1.0258 - val_accuracy: 0.6412
Epoch 92/100
226/226 [==============================] - 9s 38ms/step - loss: 0.7755 -
accuracy: 0.7156 - val_loss: 1.0256 - val_accuracy: 0.6365
Epoch 93/100
226/226 [==============================] - 9s 38ms/step - loss: 0.7730 -
accuracy: 0.7182 - val_loss: 1.0283 - val_accuracy: 0.6353
Epoch 94/100
226/226 [==============================] - 9s 38ms/step - loss: 0.7721 -
accuracy: 0.7188 - val_loss: 1.0230 - val_accuracy: 0.6401
Epoch 95/100
226/226 [==============================] - 9s 38ms/step - loss: 0.7654 -
accuracy: 0.7233 - val_loss: 1.0445 - val_accuracy: 0.6350
Epoch 96/100
226/226 [==============================] - 9s 38ms/step - loss: 0.7616 -
accuracy: 0.7226 - val_loss: 1.0411 - val_accuracy: 0.6423
Epoch 97/100
226/226 [==============================] - 9s 38ms/step - loss: 0.7522 -
accuracy: 0.7258 - val_loss: 1.0286 - val_accuracy: 0.6370
Epoch 98/100
226/226 [==============================] - 9s 38ms/step - loss: 0.7654 -
accuracy: 0.7200 - val_loss: 1.0373 - val_accuracy: 0.6364
Epoch 99/100
226/226 [==============================] - 9s 38ms/step - loss: 0.7496 -
accuracy: 0.7240 - val_loss: 1.0341 - val_accuracy: 0.6384
Epoch 100/100
226/226 [==============================] - 9s 38ms/step - loss: 0.7439 -
accuracy: 0.7294 - val_loss: 1.0194 - val_accuracy: 0.6370
```

[27]:
```
<keras.callbacks.History at 0x281a3155060>
```

[28]:
```
model_json = model.to_json()
with open("emotiondetector.json",'w') as json_file:
    json_file.write(model_json)
model.save("emotiondetector.h5")
```

[29]:
```
label = ['angry','disgust','fear','happy','neutral','sad','surprise']
```

[30]:
```
def ef(image):
    img = load_img(image,grayscale =  True )
    feature = np.array(img)
    feature = feature.reshape(1,48,48,1)
    return feature/255.0
```

[31]:
```
image = 'images/train/sad/42.jpg'
print("original image is of sad")
img = ef(image)
pred = model.predict(img)
pred_label = label[pred.argmax()]
print("model prediction is ",pred_label)
```

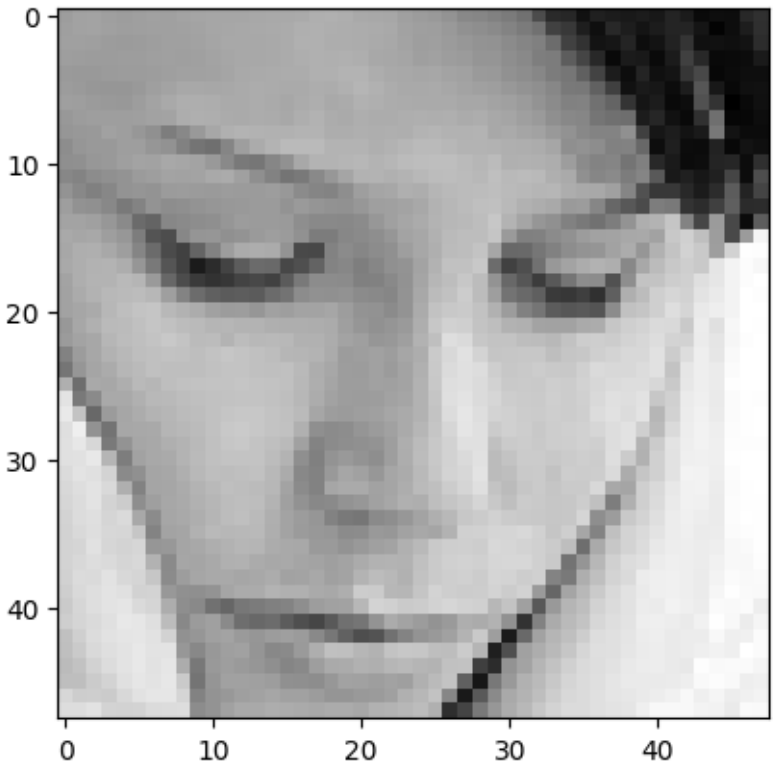
```
original image is of sad

c:\Users\basum\anaconda3\envs\pythongpu\lib\site-
packages\keras\utils\image_utils.py:409: UserWarning: grayscale is deprecated.
Please use color_mode = "grayscale"
  warnings.warn(

1/1 [==============================] - 1s 585ms/step
model prediction is  sad
```

[32]:
```
%matplotlib inline
```

[33]:
```
image = 'images/train/sad/42.jpg'
print("original image is of sad")
img = ef(image)
pred = model.predict(img)
pred_label = label[pred.argmax()]
print("model prediction is ",pred_label)
plt.imshow(img.reshape(48,48),cmap='gray')
```

```
original image is of sad
1/1 [==============================] - 0s 44ms/step
model prediction is  sad
```

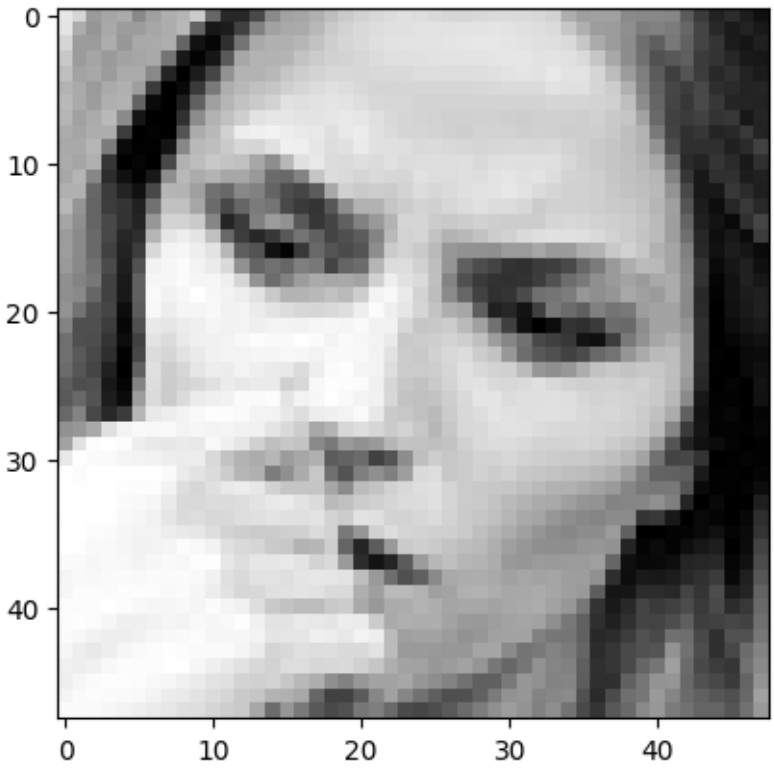
[33]:
```
<matplotlib.image.AxesImage at 0x282e0b665f0>
```



[34]:
```
image = 'images/train/fear/2.jpg'
print("original image is of fear")
img = ef(image)
pred = model.predict(img)
pred_label = label[pred.argmax()]
print("model prediction is ",pred_label)
plt.imshow(img.reshape(48,48),cmap='gray')
```

```
original image is of fear
1/1 [==============================] - 0s 38ms/step
model prediction is  fear
```

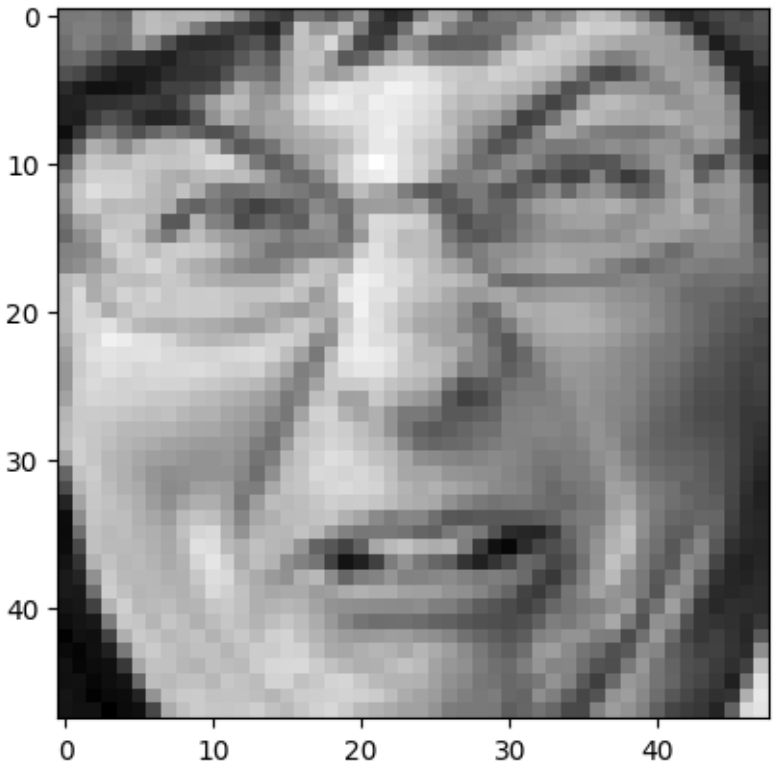
[34]: <matplotlib.image.AxesImage at 0x281f2a30f40>



[35]:
```
image = 'images/train/disgust/299.jpg'
print("original image is of disgust")
img = ef(image)
pred = model.predict(img)
pred_label = label[pred.argmax()]
print("model prediction is ",pred_label)
plt.imshow(img.reshape(48,48),cmap='gray')
```

```
original image is of disgust
1/1 [==============================] - 0s 45ms/step
model prediction is  disgust
```

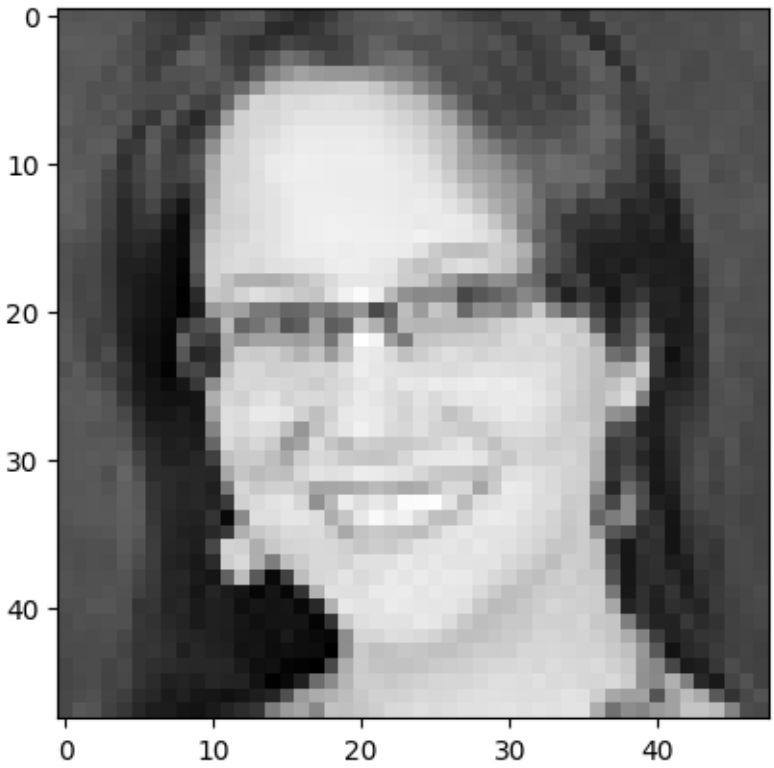
[35]: <matplotlib.image.AxesImage at 0x282e0b0b940>



[36]:
```
image = 'images/train/happy/7.jpg'
print("original image is of happy")
img = ef(image)
pred = model.predict(img)
pred_label = label[pred.argmax()]
print("model prediction is ",pred_label)
plt.imshow(img.reshape(48,48),cmap='gray')
```

```
original image is of happy
1/1 [==============================] - 0s 30ms/step
model prediction is  happy
```

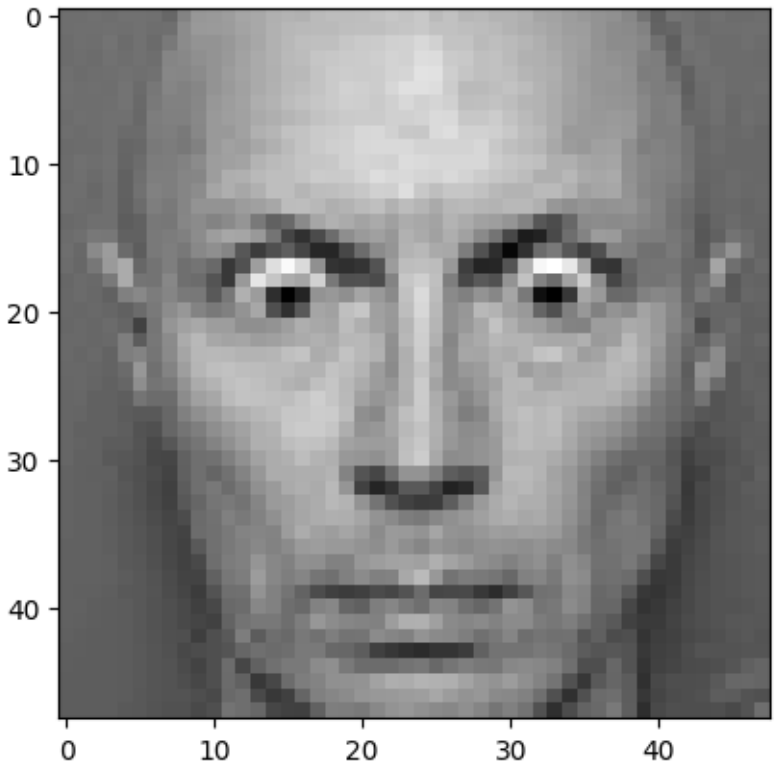
[36]: `<matplotlib.image.AxesImage at 0x281f33ad5a0>`



[37]:
```
image = 'images/train/surprise/15.jpg'
print("original image is of surprise")
img = ef(image)
pred = model.predict(img)
pred_label = label[pred.argmax()]
print("model prediction is ",pred_label)
plt.imshow(img.reshape(48,48),cmap='gray')
```

```
original image is of surprise
1/1 [==============================] - 0s 34ms/step
model prediction is  surprise
```

[37]:
```
<matplotlib.image.AxesImage at 0x281f29fa440>
```


0
10
20
30
40
0
10
20
30
40

[ ]: