facial-emotion-recognition

January 31, 2024

0.1 Import Modules

[28]:
```python
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
import random
from tqdm.notebook import tqdm
warnings.filterwarnings('ignore')
%matplotlib inline

import tensorflow as tf
from tensorflow.keras.utils import to_categorical
from keras.preprocessing.image import load_img
from keras.models import Sequential
from keras.layers import Dense, Conv2D, Dropout, Flatten, MaxPooling2D
```

0.2 Load the Dataset

[2]:
```python
TRAIN_DIR = '../input/facial-expression-dataset/train/train/'
TEST_DIR = '../input/facial-expression-dataset/test/test/'
```

[3]:
```python
def load_dataset(directory):
    image_paths = []
    labels = []

    for label in os.listdir(directory):
        for filename in os.listdir(directory+label):
            image_path = os.path.join(directory, label, filename)
            image_paths.append(image_path)
            labels.append(label)

        print(label, "Completed")

    return image_paths, labels
```

[5]:
```python
## convert into dataframe
train = pd.DataFrame()
train['image'], train['label'] = load_dataset(TRAIN_DIR)
# shuffle the dataset
train = train.sample(frac=1).reset_index(drop=True)
train.head()
```

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

[5]:

	image	label
0	../input/facial-expression-dataset/train/train…	surprise
1	../input/facial-expression-dataset/train/train…	neutral
2	../input/facial-expression-dataset/train/train…	sad
3	../input/facial-expression-dataset/train/train…	happy
4	../input/facial-expression-dataset/train/train…	angry

[6]:
```python
test = pd.DataFrame()
test['image'], test['label'] = load_dataset(TEST_DIR)
test.head()
```

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

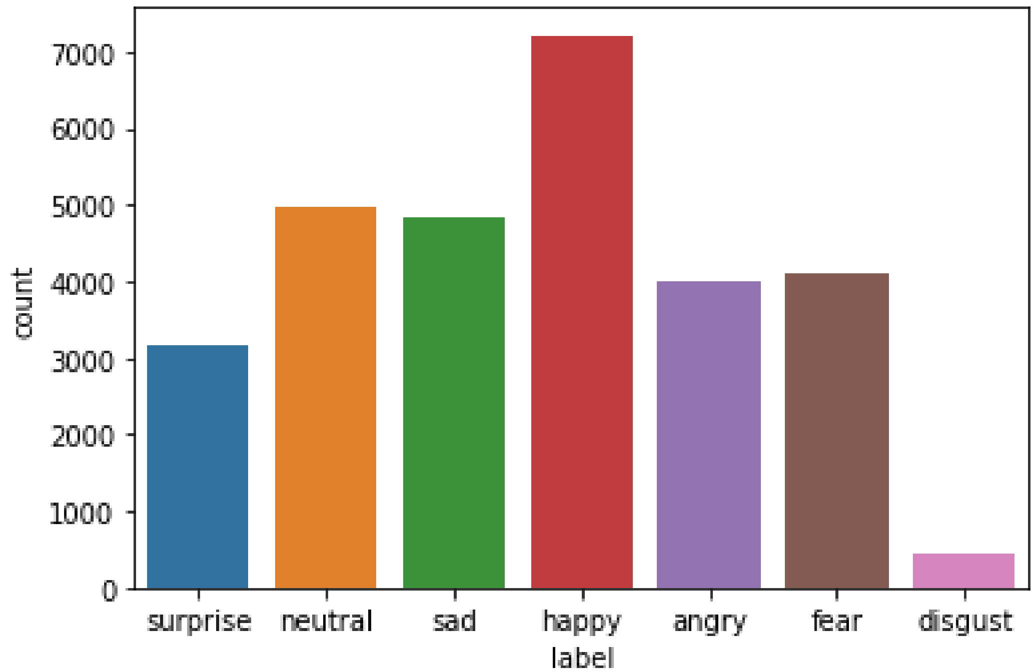
[6]:

	image	label
0	../input/facial-expression-dataset/test/test/s…	surprise
1	../input/facial-expression-dataset/test/test/s…	surprise
2	../input/facial-expression-dataset/test/test/s…	surprise
3	../input/facial-expression-dataset/test/test/s…	surprise
4	../input/facial-expression-dataset/test/test/s…	surprise

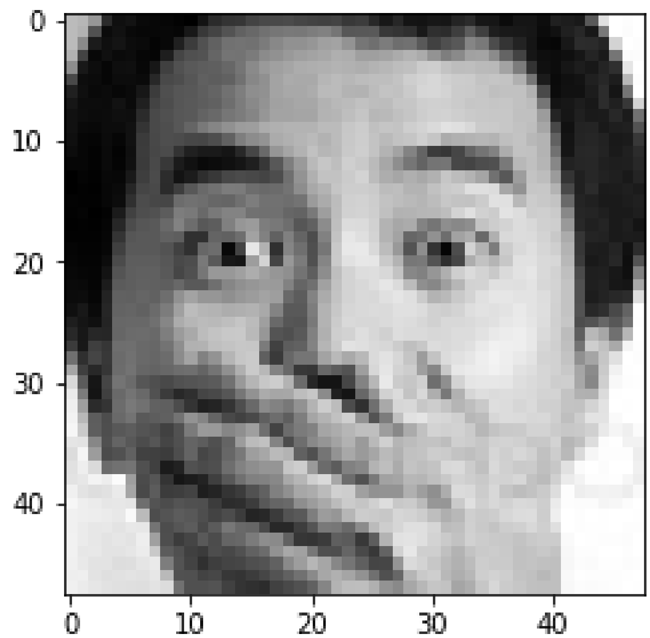
0.3 Exploratory Data Analysis

[7]:
```python
sns.countplot(train['label'])
```

[7]:
```
<AxesSubplot:xlabel='label', ylabel='count'>
```



[9]:
```
from PIL import Image
img = Image.open(train['image'][0])
plt.imshow(img, cmap='gray');
```



[13]:

```
# to display grid of images
plt.figure(figsize=(20,20))
files = train.iloc[0:25]

for index, file, label in files.itertuples():
    plt.subplot(5, 5, index+1)
    img = load_img(file)
    img = np.array(img)
    plt.imshow(img)
    plt.title(label)
    plt.axis('off')
```

0.4 Feature Extraction

[14]:
```
def extract_features(images):
    features = []
    for image in tqdm(images):
        img = load_img(image, grayscale=True)
        img = np.array(img)
        features.append(img)
    features = np.array(features)
    features = features.reshape(len(features), 48, 48, 1)
    return features
```

[15]:
```
train_features = extract_features(train['image'])
```

```
0%|          | 0/28709 [00:00<?, ?it/s]
```

[16]:
```
test_features = extract_features(test['image'])
```

```
0%|          | 0/7178 [00:00<?, ?it/s]
```

[17]:
```
## normalize the image
x_train = train_features/255.0
x_test = test_features/255.0
```

[18]:
```
## convert label to integer
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
le.fit(train['label'])
y_train = le.transform(train['label'])
y_test = le.transform(test['label'])
```

[21]:
```
y_train = to_categorical(y_train, num_classes=7)
y_test = to_categorical(y_test, num_classes=7)
```

[23]:
```
y_train[0]
```

[23]:
```
array([0., 0., 0., 0., 0., 0., 1.], dtype=float32)
```

[22]:
```
# config
input_shape = (48, 48, 1)
output_class = 7
```

0.5 Model Creation

[25]:
```
model = Sequential()
# convolutional layers
model.add(Conv2D(128, kernel_size=(3,3), activation='relu',␣
↪input_shape=input_shape))
```

```
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(output_class, activation='softmax'))

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

[26]:
```
# train the model
history = model.fit(x=x_train, y=y_train, batch_size=128, epochs=100,␣
 ↪validation_data=(x_test, y_test))
```

```
2022-03-08 09:11:04.206355: I
tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] None of the MLIR
Optimization Passes are enabled (registered 2)

Epoch 1/100

2022-03-08 09:11:05.604183: I tensorflow/stream_executor/cuda/cuda_dnn.cc:369]
Loaded cuDNN version 8005

225/225 [==============================] - 14s 31ms/step - loss: 1.8218 -
accuracy: 0.2478 - val_loss: 1.8159 - val_accuracy: 0.2471
Epoch 2/100
225/225 [==============================] - 6s 29ms/step - loss: 1.8052 -
accuracy: 0.2524 - val_loss: 1.7856 - val_accuracy: 0.2536
Epoch 3/100
225/225 [==============================] - 7s 29ms/step - loss: 1.7432 -
accuracy: 0.2815 - val_loss: 1.6658 - val_accuracy: 0.3238
Epoch 4/100
```

```
225/225 [==============================] - 7s 29ms/step - loss: 1.6653 -
accuracy: 0.3316 - val_loss: 1.6392 - val_accuracy: 0.3487
Epoch 5/100
225/225 [==============================] - 6s 29ms/step - loss: 1.5647 -
accuracy: 0.3882 - val_loss: 1.4292 - val_accuracy: 0.4551
Epoch 6/100
225/225 [==============================] - 7s 29ms/step - loss: 1.4872 -
accuracy: 0.4198 - val_loss: 1.3768 - val_accuracy: 0.4628
Epoch 7/100
225/225 [==============================] - 6s 29ms/step - loss: 1.4452 -
accuracy: 0.4432 - val_loss: 1.3394 - val_accuracy: 0.4816
Epoch 8/100
225/225 [==============================] - 6s 29ms/step - loss: 1.4051 -
accuracy: 0.4593 - val_loss: 1.2939 - val_accuracy: 0.4975
Epoch 9/100
225/225 [==============================] - 7s 29ms/step - loss: 1.3732 -
accuracy: 0.4733 - val_loss: 1.2677 - val_accuracy: 0.5123
Epoch 10/100
225/225 [==============================] - 6s 29ms/step - loss: 1.3417 -
accuracy: 0.4838 - val_loss: 1.2567 - val_accuracy: 0.5195
Epoch 11/100
225/225 [==============================] - 6s 29ms/step - loss: 1.3189 -
accuracy: 0.4936 - val_loss: 1.2226 - val_accuracy: 0.5297
Epoch 12/100
225/225 [==============================] - 6s 29ms/step - loss: 1.3063 -
accuracy: 0.5017 - val_loss: 1.2336 - val_accuracy: 0.5294
Epoch 13/100
225/225 [==============================] - 6s 29ms/step - loss: 1.2849 -
accuracy: 0.5076 - val_loss: 1.1776 - val_accuracy: 0.5492
Epoch 14/100
225/225 [==============================] - 7s 29ms/step - loss: 1.2665 -
accuracy: 0.5198 - val_loss: 1.1635 - val_accuracy: 0.5566
Epoch 15/100
225/225 [==============================] - 6s 29ms/step - loss: 1.2532 -
accuracy: 0.5224 - val_loss: 1.1527 - val_accuracy: 0.5559
Epoch 16/100
225/225 [==============================] - 6s 29ms/step - loss: 1.2382 -
accuracy: 0.5283 - val_loss: 1.1492 - val_accuracy: 0.5568
Epoch 17/100
225/225 [==============================] - 6s 29ms/step - loss: 1.2277 -
accuracy: 0.5344 - val_loss: 1.1441 - val_accuracy: 0.5639
Epoch 18/100
225/225 [==============================] - 7s 29ms/step - loss: 1.2091 -
accuracy: 0.5413 - val_loss: 1.1344 - val_accuracy: 0.5665
Epoch 19/100
225/225 [==============================] - 7s 29ms/step - loss: 1.1989 -
accuracy: 0.5445 - val_loss: 1.1263 - val_accuracy: 0.5698
Epoch 20/100
```

```
225/225 [==============================] - 6s 29ms/step - loss: 1.1978 -
accuracy: 0.5453 - val_loss: 1.1237 - val_accuracy: 0.5702
Epoch 21/100
225/225 [==============================] - 6s 29ms/step - loss: 1.1810 -
accuracy: 0.5502 - val_loss: 1.1084 - val_accuracy: 0.5741
Epoch 22/100
225/225 [==============================] - 6s 29ms/step - loss: 1.1712 -
accuracy: 0.5583 - val_loss: 1.1100 - val_accuracy: 0.5762
Epoch 23/100
225/225 [==============================] - 6s 29ms/step - loss: 1.1635 -
accuracy: 0.5582 - val_loss: 1.0952 - val_accuracy: 0.5779
Epoch 24/100
225/225 [==============================] - 7s 29ms/step - loss: 1.1544 -
accuracy: 0.5636 - val_loss: 1.1017 - val_accuracy: 0.5741
Epoch 25/100
225/225 [==============================] - 6s 29ms/step - loss: 1.1455 -
accuracy: 0.5676 - val_loss: 1.0890 - val_accuracy: 0.5829
Epoch 26/100
225/225 [==============================] - 6s 29ms/step - loss: 1.1366 -
accuracy: 0.5721 - val_loss: 1.0854 - val_accuracy: 0.5822
Epoch 27/100
225/225 [==============================] - 6s 29ms/step - loss: 1.1228 -
accuracy: 0.5773 - val_loss: 1.0724 - val_accuracy: 0.5935
Epoch 28/100
225/225 [==============================] - 6s 29ms/step - loss: 1.1224 -
accuracy: 0.5755 - val_loss: 1.0771 - val_accuracy: 0.5889
Epoch 29/100
225/225 [==============================] - 7s 30ms/step - loss: 1.1160 -
accuracy: 0.5811 - val_loss: 1.0764 - val_accuracy: 0.5919
Epoch 30/100
225/225 [==============================] - 7s 29ms/step - loss: 1.1161 -
accuracy: 0.5796 - val_loss: 1.0702 - val_accuracy: 0.5914
Epoch 31/100
225/225 [==============================] - 6s 29ms/step - loss: 1.1035 -
accuracy: 0.5836 - val_loss: 1.0816 - val_accuracy: 0.5864
Epoch 32/100
225/225 [==============================] - 6s 29ms/step - loss: 1.0941 -
accuracy: 0.5926 - val_loss: 1.0675 - val_accuracy: 0.5963
Epoch 33/100
225/225 [==============================] - 6s 29ms/step - loss: 1.0892 -
accuracy: 0.5892 - val_loss: 1.0678 - val_accuracy: 0.5946
Epoch 34/100
225/225 [==============================] - 7s 29ms/step - loss: 1.0886 -
accuracy: 0.5887 - val_loss: 1.0627 - val_accuracy: 0.6010
Epoch 35/100
225/225 [==============================] - 7s 29ms/step - loss: 1.0653 -
accuracy: 0.6000 - val_loss: 1.0540 - val_accuracy: 0.5954
Epoch 36/100
```

```
225/225 [==============================] - 6s 29ms/step - loss: 1.0660 -
accuracy: 0.5983 - val_loss: 1.0544 - val_accuracy: 0.6025
Epoch 37/100
225/225 [==============================] - 6s 29ms/step - loss: 1.0673 -
accuracy: 0.5986 - val_loss: 1.0633 - val_accuracy: 0.5949
Epoch 38/100
225/225 [==============================] - 6s 28ms/step - loss: 1.0485 -
accuracy: 0.6078 - val_loss: 1.0528 - val_accuracy: 0.5982
Epoch 39/100
225/225 [==============================] - 6s 29ms/step - loss: 1.0419 -
accuracy: 0.6083 - val_loss: 1.0494 - val_accuracy: 0.6046
Epoch 40/100
225/225 [==============================] - 7s 29ms/step - loss: 1.0392 -
accuracy: 0.6074 - val_loss: 1.0483 - val_accuracy: 0.6070
Epoch 41/100
225/225 [==============================] - 6s 29ms/step - loss: 1.0396 -
accuracy: 0.6082 - val_loss: 1.0508 - val_accuracy: 0.6059
Epoch 42/100
225/225 [==============================] - 6s 29ms/step - loss: 1.0320 -
accuracy: 0.6121 - val_loss: 1.0501 - val_accuracy: 0.6011
Epoch 43/100
225/225 [==============================] - 6s 29ms/step - loss: 1.0233 -
accuracy: 0.6161 - val_loss: 1.0453 - val_accuracy: 0.6099
Epoch 44/100
225/225 [==============================] - 6s 29ms/step - loss: 1.0147 -
accuracy: 0.6167 - val_loss: 1.0366 - val_accuracy: 0.6073
Epoch 45/100
225/225 [==============================] - 7s 29ms/step - loss: 1.0097 -
accuracy: 0.6230 - val_loss: 1.0467 - val_accuracy: 0.6048
Epoch 46/100
225/225 [==============================] - 7s 29ms/step - loss: 1.0191 -
accuracy: 0.6184 - val_loss: 1.0497 - val_accuracy: 0.6057
Epoch 47/100
225/225 [==============================] - 6s 29ms/step - loss: 1.0009 -
accuracy: 0.6241 - val_loss: 1.0445 - val_accuracy: 0.6119
Epoch 48/100
225/225 [==============================] - 6s 29ms/step - loss: 0.9992 -
accuracy: 0.6242 - val_loss: 1.0459 - val_accuracy: 0.6076
Epoch 49/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9926 -
accuracy: 0.6292 - val_loss: 1.0314 - val_accuracy: 0.6071
Epoch 50/100
225/225 [==============================] - 7s 30ms/step - loss: 0.9900 -
accuracy: 0.6272 - val_loss: 1.0436 - val_accuracy: 0.6055
Epoch 51/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9812 -
accuracy: 0.6349 - val_loss: 1.0347 - val_accuracy: 0.6117
Epoch 52/100
```

```
225/225 [==============================] - 7s 29ms/step - loss: 0.9747 -
accuracy: 0.6349 - val_loss: 1.0333 - val_accuracy: 0.6120
Epoch 53/100
225/225 [==============================] - 6s 29ms/step - loss: 0.9731 -
accuracy: 0.6376 - val_loss: 1.0249 - val_accuracy: 0.6108
Epoch 54/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9672 -
accuracy: 0.6365 - val_loss: 1.0322 - val_accuracy: 0.6141
Epoch 55/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9626 -
accuracy: 0.6434 - val_loss: 1.0242 - val_accuracy: 0.6156
Epoch 56/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9481 -
accuracy: 0.6465 - val_loss: 1.0284 - val_accuracy: 0.6127
Epoch 57/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9576 -
accuracy: 0.6449 - val_loss: 1.0284 - val_accuracy: 0.6152
Epoch 58/100
225/225 [==============================] - 7s 30ms/step - loss: 0.9499 -
accuracy: 0.6460 - val_loss: 1.0208 - val_accuracy: 0.6160
Epoch 59/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9435 -
accuracy: 0.6452 - val_loss: 1.0327 - val_accuracy: 0.6074
Epoch 60/100
225/225 [==============================] - 7s 30ms/step - loss: 0.9449 -
accuracy: 0.6467 - val_loss: 1.0199 - val_accuracy: 0.6211
Epoch 61/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9341 -
accuracy: 0.6538 - val_loss: 1.0220 - val_accuracy: 0.6181
Epoch 62/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9347 -
accuracy: 0.6516 - val_loss: 1.0261 - val_accuracy: 0.6179
Epoch 63/100
225/225 [==============================] - 6s 29ms/step - loss: 0.9232 -
accuracy: 0.6582 - val_loss: 1.0252 - val_accuracy: 0.6172
Epoch 64/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9212 -
accuracy: 0.6594 - val_loss: 1.0255 - val_accuracy: 0.6176
Epoch 65/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9238 -
accuracy: 0.6555 - val_loss: 1.0198 - val_accuracy: 0.6193
Epoch 66/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9115 -
accuracy: 0.6615 - val_loss: 1.0285 - val_accuracy: 0.6179
Epoch 67/100
225/225 [==============================] - 7s 30ms/step - loss: 0.9076 -
accuracy: 0.6616 - val_loss: 1.0253 - val_accuracy: 0.6232
Epoch 68/100
```

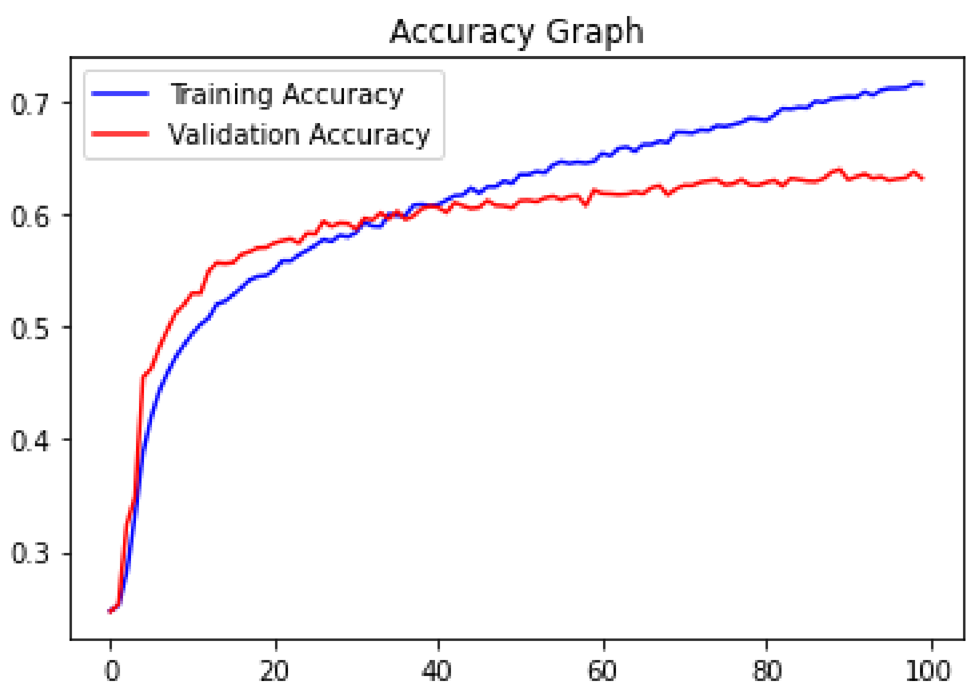
```
225/225 [==============================] - 6s 29ms/step - loss: 0.9027 -
accuracy: 0.6646 - val_loss: 1.0196 - val_accuracy: 0.6251
Epoch 69/100
225/225 [==============================] - 7s 29ms/step - loss: 0.9028 -
accuracy: 0.6634 - val_loss: 1.0252 - val_accuracy: 0.6169
Epoch 70/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8905 -
accuracy: 0.6726 - val_loss: 1.0166 - val_accuracy: 0.6222
Epoch 71/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8910 -
accuracy: 0.6724 - val_loss: 1.0226 - val_accuracy: 0.6254
Epoch 72/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8885 -
accuracy: 0.6717 - val_loss: 1.0107 - val_accuracy: 0.6252
Epoch 73/100
225/225 [==============================] - 6s 29ms/step - loss: 0.8775 -
accuracy: 0.6746 - val_loss: 1.0078 - val_accuracy: 0.6282
Epoch 74/100
225/225 [==============================] - 6s 29ms/step - loss: 0.8832 -
accuracy: 0.6741 - val_loss: 1.0181 - val_accuracy: 0.6293
Epoch 75/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8732 -
accuracy: 0.6784 - val_loss: 1.0121 - val_accuracy: 0.6303
Epoch 76/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8729 -
accuracy: 0.6776 - val_loss: 1.0181 - val_accuracy: 0.6258
Epoch 77/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8624 -
accuracy: 0.6793 - val_loss: 1.0195 - val_accuracy: 0.6268
Epoch 78/100
225/225 [==============================] - 6s 29ms/step - loss: 0.8577 -
accuracy: 0.6814 - val_loss: 1.0172 - val_accuracy: 0.6305
Epoch 79/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8570 -
accuracy: 0.6850 - val_loss: 1.0207 - val_accuracy: 0.6258
Epoch 80/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8553 -
accuracy: 0.6842 - val_loss: 1.0134 - val_accuracy: 0.6254
Epoch 81/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8550 -
accuracy: 0.6830 - val_loss: 1.0204 - val_accuracy: 0.6279
Epoch 82/100
225/225 [==============================] - 6s 29ms/step - loss: 0.8450 -
accuracy: 0.6876 - val_loss: 1.0171 - val_accuracy: 0.6298
Epoch 83/100
225/225 [==============================] - 6s 29ms/step - loss: 0.8396 -
accuracy: 0.6934 - val_loss: 1.0373 - val_accuracy: 0.6248
Epoch 84/100
```

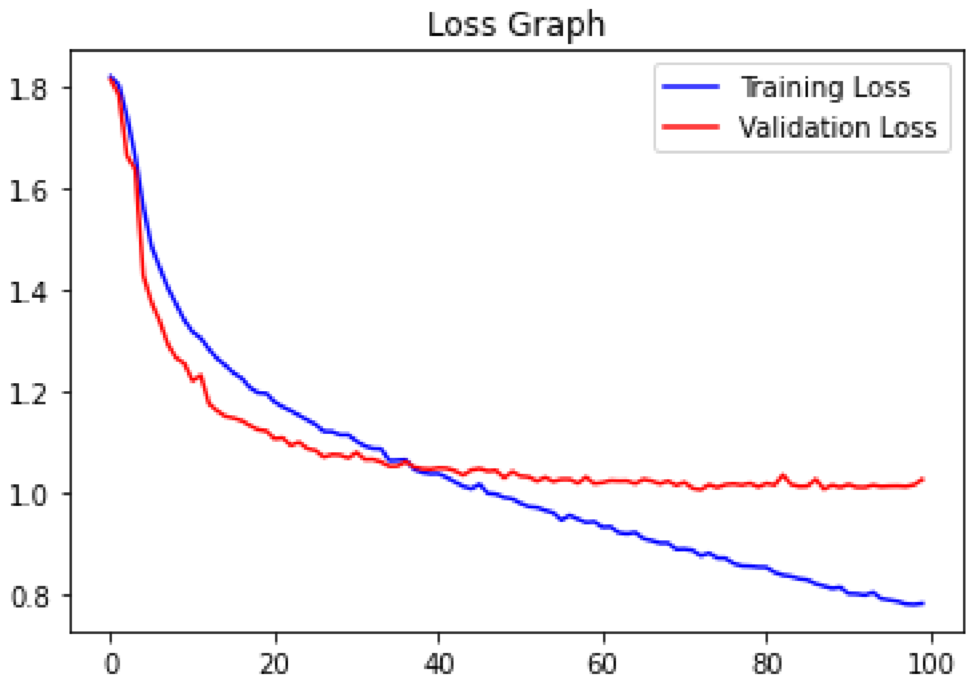
```
225/225 [==============================] - 6s 29ms/step - loss: 0.8372 -
accuracy: 0.6930 - val_loss: 1.0185 - val_accuracy: 0.6315
Epoch 85/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8328 -
accuracy: 0.6945 - val_loss: 1.0143 - val_accuracy: 0.6305
Epoch 86/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8305 -
accuracy: 0.6939 - val_loss: 1.0149 - val_accuracy: 0.6291
Epoch 87/100
225/225 [==============================] - 6s 29ms/step - loss: 0.8228 -
accuracy: 0.6999 - val_loss: 1.0286 - val_accuracy: 0.6287
Epoch 88/100
225/225 [==============================] - 6s 29ms/step - loss: 0.8185 -
accuracy: 0.6990 - val_loss: 1.0092 - val_accuracy: 0.6312
Epoch 89/100
225/225 [==============================] - 6s 29ms/step - loss: 0.8137 -
accuracy: 0.7020 - val_loss: 1.0170 - val_accuracy: 0.6371
Epoch 90/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8160 -
accuracy: 0.7028 - val_loss: 1.0126 - val_accuracy: 0.6395
Epoch 91/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8035 -
accuracy: 0.7037 - val_loss: 1.0184 - val_accuracy: 0.6304
Epoch 92/100
225/225 [==============================] - 6s 29ms/step - loss: 0.8029 -
accuracy: 0.7034 - val_loss: 1.0130 - val_accuracy: 0.6329
Epoch 93/100
225/225 [==============================] - 7s 29ms/step - loss: 0.8005 -
accuracy: 0.7082 - val_loss: 1.0127 - val_accuracy: 0.6354
Epoch 94/100
225/225 [==============================] - 6s 29ms/step - loss: 0.8059 -
accuracy: 0.7052 - val_loss: 1.0169 - val_accuracy: 0.6317
Epoch 95/100
225/225 [==============================] - 7s 29ms/step - loss: 0.7935 -
accuracy: 0.7096 - val_loss: 1.0140 - val_accuracy: 0.6329
Epoch 96/100
225/225 [==============================] - 7s 29ms/step - loss: 0.7909 -
accuracy: 0.7113 - val_loss: 1.0149 - val_accuracy: 0.6297
Epoch 97/100
225/225 [==============================] - 6s 29ms/step - loss: 0.7882 -
accuracy: 0.7113 - val_loss: 1.0152 - val_accuracy: 0.6311
Epoch 98/100
225/225 [==============================] - 6s 29ms/step - loss: 0.7829 -
accuracy: 0.7118 - val_loss: 1.0143 - val_accuracy: 0.6319
Epoch 99/100
225/225 [==============================] - 6s 29ms/step - loss: 0.7819 -
accuracy: 0.7156 - val_loss: 1.0176 - val_accuracy: 0.6372
Epoch 100/100
```

225/225 [==============================] - 6s 29ms/step - loss: 0.7840 - accuracy: 0.7151 - val_loss: 1.0282 - val_accuracy: 0.6317

0.6 Plot the Results

[30]:
```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
epochs = range(len(acc))

plt.plot(epochs, acc, 'b', label='Training Accuracy')
plt.plot(epochs, val_acc, 'r', label='Validation Accuracy')
plt.title('Accuracy Graph')
plt.legend()
plt.figure()

loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(len(acc))

plt.plot(epochs, loss, 'b', label='Training Loss')
plt.plot(epochs, val_loss, 'r', label='Validation Loss')
plt.title('Loss Graph')
plt.legend()

plt.show()
```

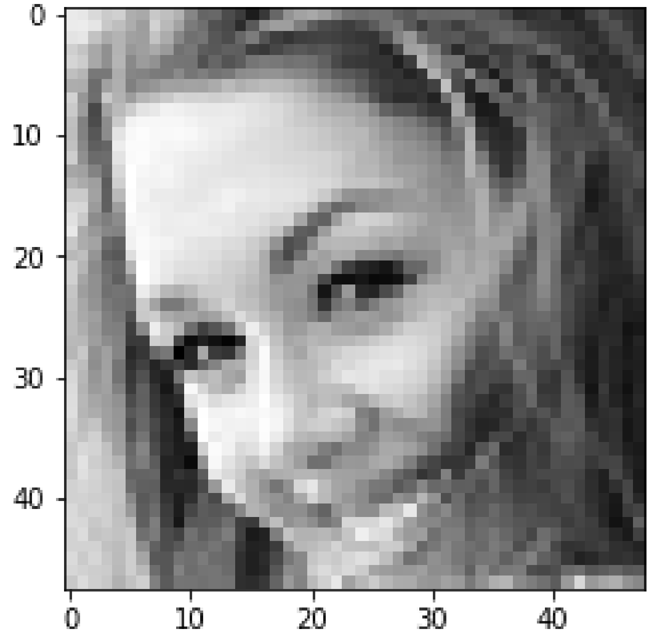




0.7 Test with Image Data

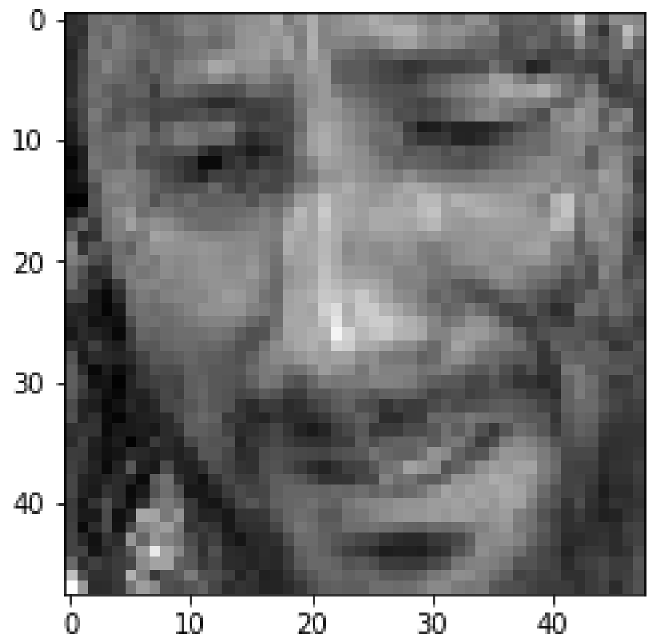
[40]:
```
image_index = random.randint(0, len(test))
print("Original Output:", test['label'][image_index])
pred = model.predict(x_test[image_index].reshape(1, 48, 48, 1))
prediction_label = le.inverse_transform([pred.argmax()])[0]
print("Predicted Output:", prediction_label)
plt.imshow(x_test[image_index].reshape(48, 48), cmap='gray');
```

```
Original Output: happy
Predicted Output: happy
```



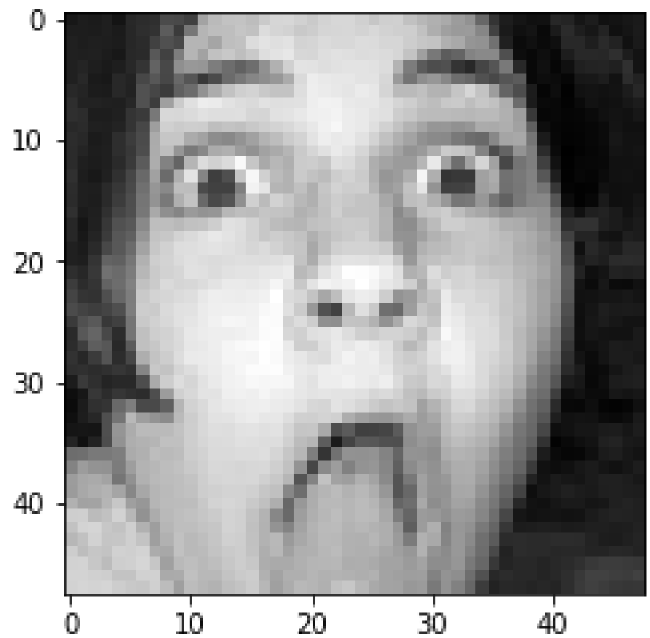
[43]:
```
image_index = random.randint(0, len(test))
print("Original Output:", test['label'][image_index])
pred = model.predict(x_test[image_index].reshape(1, 48, 48, 1))
prediction_label = le.inverse_transform([pred.argmax()])[0]
print("Predicted Output:", prediction_label)
plt.imshow(x_test[image_index].reshape(48, 48), cmap='gray');
```

```
Original Output: sad
Predicted Output: sad
```



[44]:
```
image_index = random.randint(0, len(test))
print("Original Output:", test['label'][image_index])
pred = model.predict(x_test[image_index].reshape(1, 48, 48, 1))
prediction_label = le.inverse_transform([pred.argmax()])[0]
print("Predicted Output:", prediction_label)
plt.imshow(x_test[image_index].reshape(48, 48), cmap='gray');
```

```
Original Output: fear
Predicted Output: surprise
```


0
10
20
30
40
0
10
20
30
40