Importing necessary libraries

In [1]:

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import os


from keras.preprocessing.image import load_img, img_to_array
from keras.preprocessing.image import ImageDataGenerator
from keras.layers import Dense,Input,Dropout,GlobalAveragePooling2D,Flatten,Conv2D,
from keras.models import Model,Sequential
from keras.optimizers import Adam,SGD,RMSprop
```

```
2024-02-25 04:20:22.226513: E external/local_xla/xla/stream_executor/cuda/cuda_dnn.c
c:9261] Unable to register cuDNN factory: Attempting to register factory for plugin
cuDNN when one has already been registered
2024-02-25 04:20:22.226607: E external/local_xla/xla/stream_executor/cuda/cuda_fft.c
c:607] Unable to register cuFFT factory: Attempting to register factory for plugin c
uFFT when one has already been registered
2024-02-25 04:20:22.361731: E external/local_xla/xla/stream_executor/cuda/cuda_blas.
cc:1515] Unable to register cuBLAS factory: Attempting to register factory for plugi
n cuBLAS when one has already been registered
```

In [2]:

```
picture_size = 48
folder_path = "../input/face-expression-recognition-dataset/images/"
```

In [3]:

```
expression = 'angry'

plt.figure(figsize= (12,12))
for i in range(1, 10, 1):
    plt.subplot(3,3,i)
    img = load_img(folder_path+"train/"+expression+"/"+
                  os.listdir(folder_path + "train/" + expression)[i], target_size=(
    plt.imshow(img)
plt.show()
```

In [4]:

```
batch_size  = 128

datagen_train  = ImageDataGenerator()
datagen_val = ImageDataGenerator()

train_set = datagen_train.flow_from_directory(folder_path+"train",
                                              target_size = (picture_size,picture_s
                                              color_mode = "grayscale",
                                              batch_size=batch_size,
                                              class_mode='categorical',
                                              shuffle=True)


test_set = datagen_val.flow_from_directory(folder_path+"validation",
                                              target_size = (picture_size,picture_s
                                              color_mode = "grayscale",
                                              batch_size=batch_size,
                                              class_mode='categorical',
                                              shuffle=False)
```

```
Found 28821 images belonging to 7 classes.
Found 7066 images belonging to 7 classes.
```

CNN Layers

In [5]:

```
from keras.optimizers import Adam,SGD,RMSprop


no_of_classes = 7

model = Sequential()

#1st CNN layer
model.add(Conv2D(64,(3,3),padding = 'same',input_shape = (48,48,1)))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Dropout(0.25))

#2nd CNN layer
model.add(Conv2D(128,(5,5),padding = 'same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Dropout (0.25))

#3rd CNN layer
model.add(Conv2D(512,(3,3),padding = 'same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size = (2,2)))
model.add(Dropout (0.25))

#4th CNN layer
model.add(Conv2D(512,(3,3), padding='same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Flatten())

#Fully connected 1st layer
model.add(Dense(256))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.25))


# Fully connected layer 2nd layer
model.add(Dense(512))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.25))
model.add(Dense(no_of_classes, activation='softmax'))



opt = Adam(lr = 0.0001)
model.compile(optimizer=opt,loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 48, 48, 64)        640

 batch_normalization (Batch  (None, 48, 48, 64)        256
 Normalization)

 activation (Activation)     (None, 48, 48, 64)        0

 max_pooling2d (MaxPooling2  (None, 24, 24, 64)        0
 D)

 dropout (Dropout)           (None, 24, 24, 64)        0

 conv2d_1 (Conv2D)           (None, 24, 24, 128)       204928

 batch_normalization_1 (Bat  (None, 24, 24, 128)       512
 chNormalization)

 activation_1 (Activation)   (None, 24, 24, 128)       0

 max_pooling2d_1 (MaxPoolin  (None, 12, 12, 128)       0
 g2D)

 dropout_1 (Dropout)         (None, 12, 12, 128)       0

 conv2d_2 (Conv2D)           (None, 12, 12, 512)       590336

 batch_normalization_2 (Bat  (None, 12, 12, 512)       2048
 chNormalization)

 activation_2 (Activation)   (None, 12, 12, 512)       0

 max_pooling2d_2 (MaxPoolin  (None, 6, 6, 512)         0
 g2D)

 dropout_2 (Dropout)         (None, 6, 6, 512)         0

 conv2d_3 (Conv2D)           (None, 6, 6, 512)         2359808

 batch_normalization_3 (Bat  (None, 6, 6, 512)         2048
 chNormalization)

 activation_3 (Activation)   (None, 6, 6, 512)         0

 max_pooling2d_3 (MaxPoolin  (None, 3, 3, 512)         0
 g2D)

 dropout_3 (Dropout)         (None, 3, 3, 512)         0

 flatten (Flatten)           (None, 4608)              0

 dense (Dense)               (None, 256)               1179904
 batch_normalization_4 (Bat  (None, 256)              1024      
 chNormalization)                                                

 activation_4 (Activation)   (None, 256)              0         

 dropout_4 (Dropout)         (None, 256)              0         

 dense_1 (Dense)             (None, 512)              131584    

 batch_normalization_5 (Bat  (None, 512)              2048      
 chNormalization)                                                

 activation_5 (Activation)   (None, 512)              0         

 dropout_5 (Dropout)         (None, 512)              0         

 dense_2 (Dense)             (None, 7)                3591      

=================================================================
Total params: 4478727 (17.08 MB)
Trainable params: 4474759 (17.07 MB)
Non-trainable params: 3968 (15.50 KB)
_________________________________________________________________
```

In [6]:

```
from keras.optimizers import RMSprop,SGD,Adam
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau

checkpoint = ModelCheckpoint("./model.h5", monitor='val_acc', verbose=1, save_best_

early_stopping = EarlyStopping(monitor='val_loss',
                          min_delta=0,
                          patience=3,
                          verbose=1,
                          restore_best_weights=True
                          )

reduce_learningrate = ReduceLROnPlateau(monitor='val_loss',
                              factor=0.2,
                              patience=3,
                              verbose=1,
                              min_delta=0.0001)

callbacks_list = [early_stopping,checkpoint,reduce_learningrate]

epochs = 48

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

In [7]:

```
history = model.fit_generator(generator=train_set,
                                steps_per_epoch=train_set.n//train_set.batch_size,
                                epochs=epochs,
                                validation_data = test_set,
                                validation_steps = test_set.n//test_set.batch_size,
                                callbacks=callbacks_list
                                )
```

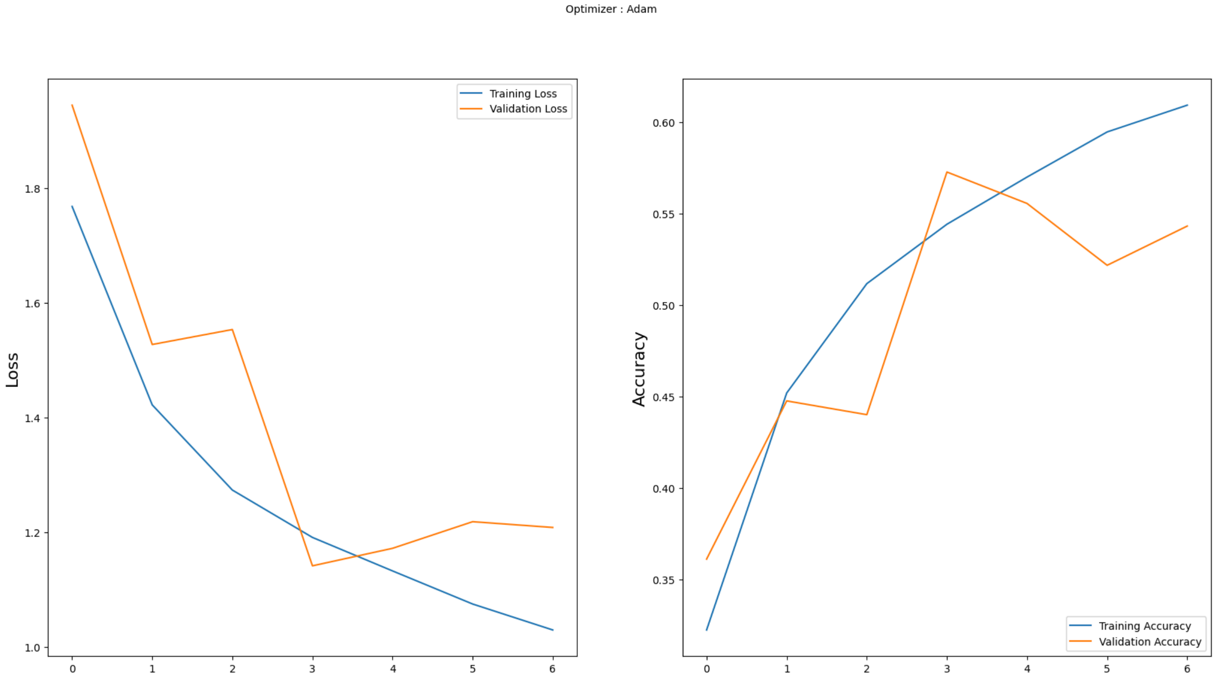
```
/tmp/ipykernel_27/3853974908.py:1: UserWarning: `Model.fit_generator` is deprecated
and will be removed in a future version. Please use `Model.fit`, which supports gene
rators.
  history = model.fit_generator(generator=train_set,
Epoch 1/48
2024-02-25 04:21:08.099823: E tensorflow/core/grappler/optimizers/meta_optimizer.cc:
961] layout failed: INVALID_ARGUMENT: Size of values 0 does not match size of permut
ation 4 @ fanin shape insequential/dropout/dropout/SelectV2-2-TransposeNHWCToNCHW-La
youtOptimizer
WARNING: All log messages before absl::InitializeLog() is called are written to STDE
RR
I0000 00:00:1708834872.928062      85 device_compiler.h:186] Compiled cluster using
XLA!  This line is logged at most once for the lifetime of the process.
225/225 [==============================] - 233s 995ms/step - loss: 1.7677 - accurac
y: 0.3225 - val_loss: 1.9443 - val_accuracy: 0.3612 - lr: 0.0010
Epoch 2/48
225/225 [==============================] - 29s 129ms/step - loss: 1.4219 - accuracy:
0.4521 - val_loss: 1.5272 - val_accuracy: 0.4477 - lr: 0.0010
Epoch 3/48
225/225 [==============================] - 30s 134ms/step - loss: 1.2736 - accuracy:
0.5118 - val_loss: 1.5533 - val_accuracy: 0.4402 - lr: 0.0010
Epoch 4/48
225/225 [==============================] - 29s 129ms/step - loss: 1.1910 - accuracy:
0.5443 - val_loss: 1.1415 - val_accuracy: 0.5729 - lr: 0.0010
Epoch 5/48
225/225 [==============================] - 30s 132ms/step - loss: 1.1326 - accuracy:
0.5701 - val_loss: 1.1720 - val_accuracy: 0.5557 - lr: 0.0010
Epoch 6/48
225/225 [==============================] - 29s 128ms/step - loss: 1.0749 - accuracy:
0.5948 - val_loss: 1.2184 - val_accuracy: 0.5219 - lr: 0.0010
Epoch 7/48
225/225 [==============================] - ETA: 0s - loss: 1.0297 - accuracy: 0.6094
Restoring model weights from the end of the best epoch: 4.

Epoch 7: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
225/225 [==============================] - 33s 147ms/step - loss: 1.0297 - accuracy:
0.6094 - val_loss: 1.2083 - val_accuracy: 0.5433 - lr: 0.0010
Epoch 7: early stopping
```

In [8]:

```
plt.figure(figsize=(20,10))
plt.subplot(1, 2, 1)
plt.suptitle('Optimizer : Adam', fontsize=10)
plt.ylabel('Loss', fontsize=16)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.legend(loc='upper right')

plt.subplot(1, 2, 2)
plt.ylabel('Accuracy', fontsize=16)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend(loc='lower right')
plt.show()
```


Optimizer : Adam
Training Loss
Validation Loss
Loss
Accuracy
Training Accuracy
Validation Accuracy