

```
import os
import shutil
os.environ['KAGGLE_CONFIG_DIR']='/content'
```

link_to_dataset [link](#)

▼ Downloading the dataset through kaggle API

```
!kaggle datasets download -d jonathanheix/face-expression-recognition-dataset
```

```
Warning: Your Kaggle API key is readable by other users on this system! To fix this, you
Downloading face-expression-recognition-dataset.zip to /content
 94% 113M/121M [00:01<00:00, 94.4MB/s]
100% 121M/121M [00:01<00:00, 113MB/s]
```

▼ Unzipping the downloaded zip file

```
!unzip "/content/face-expression-recognition-dataset.zip" -d "/content"
```

```
Streaming output truncated to the last 5000 lines.
  inflating: /content/images/validation/fear/8797.jpg
  inflating: /content/images/validation/fear/8818.jpg
  inflating: /content/images/validation/fear/886.jpg
  inflating: /content/images/validation/fear/9037.jpg
  inflating: /content/images/validation/fear/9040.jpg
  inflating: /content/images/validation/fear/9101.jpg
  inflating: /content/images/validation/fear/911.jpg
  inflating: /content/images/validation/fear/9179.jpg
  inflating: /content/images/validation/fear/9205.jpg
Saving...      inflating: /content/images/validation/fear/9232.jpg
  inflating: /content/images/validation/fear/9251.jpg
  inflating: /content/images/validation/fear/9261.jpg
  inflating: /content/images/validation/fear/9281.jpg
  inflating: /content/images/validation/fear/9302.jpg
  inflating: /content/images/validation/fear/9333.jpg
  inflating: /content/images/validation/fear/9369.jpg
  inflating: /content/images/validation/fear/9370.jpg
  inflating: /content/images/validation/fear/9474.jpg
  inflating: /content/images/validation/fear/949.jpg
  inflating: /content/images/validation/fear/9602.jpg
  inflating: /content/images/validation/fear/9606.jpg
  inflating: /content/images/validation/fear/9842.jpg
  inflating: /content/images/validation/fear/9898.jpg
  inflating: /content/images/validation/happy/10019.jpg
  inflating: /content/images/validation/happy/10023.jpg
```

inflating: /content/images/validation/happy/10074.jpg
inflating: /content/images/validation/happy/10096.jpg
inflating: /content/images/validation/happy/10106.jpg
inflating: /content/images/validation/happy/10126.jpg
inflating: /content/images/validation/happy/10138.jpg
inflating: /content/images/validation/happy/10141.jpg
inflating: /content/images/validation/happy/1020.jpg
inflating: /content/images/validation/happy/10218.jpg
inflating: /content/images/validation/happy/10237.jpg
inflating: /content/images/validation/happy/10248.jpg
inflating: /content/images/validation/happy/10257.jpg
inflating: /content/images/validation/happy/1027.jpg
inflating: /content/images/validation/happy/10273.jpg
inflating: /content/images/validation/happy/10276.jpg
inflating: /content/images/validation/happy/10312.jpg
inflating: /content/images/validation/happy/10317.jpg
inflating: /content/images/validation/happy/10344.jpg
inflating: /content/images/validation/happy/10362.jpg
inflating: /content/images/validation/happy/10367.jpg
inflating: /content/images/validation/happy/10370.jpg
inflating: /content/images/validation/happy/10432.jpg
inflating: /content/images/validation/happy/10456.jpg
inflating: /content/images/validation/happy/10467.jpg
inflating: /content/images/validation/happy/10468.jpg
inflating: /content/images/validation/happy/10480.jpg
inflating: /content/images/validation/happy/10528.jpg
inflating: /content/images/validation/happy/10540.jpg
inflating: /content/images/validation/happy/10552.jpg
inflating: /content/images/validation/happy/1056.jpg
inflating: /content/images/validation/happy/10571.jpg
inflating: /content/images/validation/happy/1058.jpg
inflating: /content/images/validation/happy/10622.jpg
inflating: /content/images/validation/happy/10638.jpg

We got 7 different classes of emotion dataset. As part of our case study we require only 4 classes

Saving...

Here we are coping the required classes of images into another folder

```
classes = ["angry", "happy", "neutral", "sad"]

for emotion in classes:
    src1 = "/content/images/train/" + str(emotion)
    dest1 = "/content/dataset/train/" + str(emotion)
    src2 = "/content/images/validation/" + str(emotion)
    dest2 = "/content/dataset/test/" + str(emotion)
    shutil.rmtree(dest1)
    shutil.copytree(src1, dest1)
```

```
shutil.copytree(src2, dest2)
print("{} emotion images copied".format(emotion))

angry emotion images copied
happy emotion images copied
neutral emotion images copied
sad emotion images copied
```

▼ Importing the required packages

```
from tensorflow.keras.applications import VGG16
from tensorflow.keras.layers import Conv2D, MaxPool2D
from tensorflow.keras.layers import Activation, Dropout, Flatten, Dense, Input
from tensorflow.keras.models import Model
from keras.preprocessing.image import load_img, img_to_array
from keras.preprocessing.image import ImageDataGenerator
#from tensorflow import keras
import matplotlib.pyplot as plt
import tensorflow as tf
import os
import numpy as np
import datetime
%load_ext tensorboard

tf.keras.backend.clear_session()
```

Our dataset is having image size 48X48. Hence we picked image size



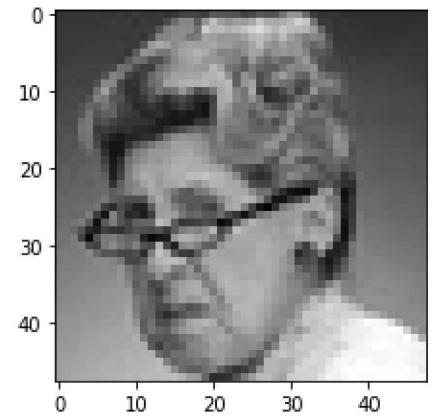
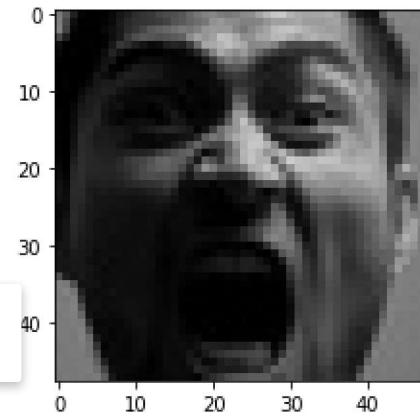
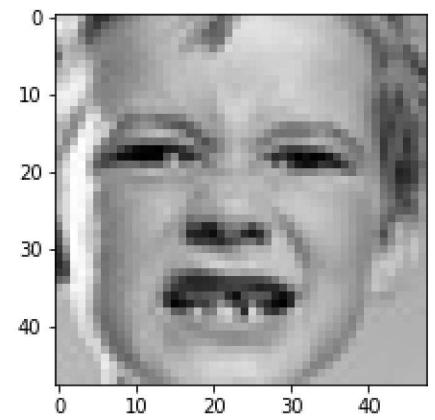
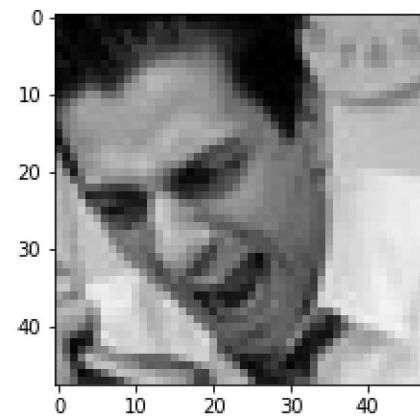
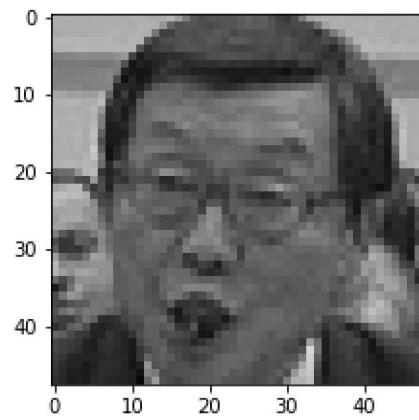
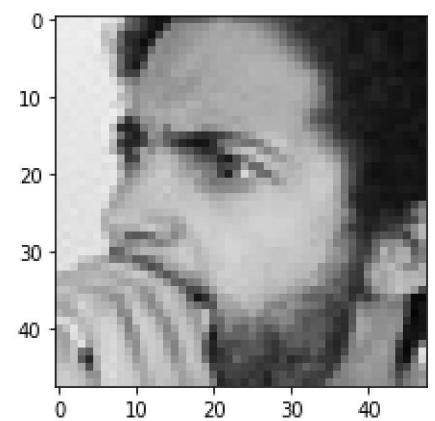
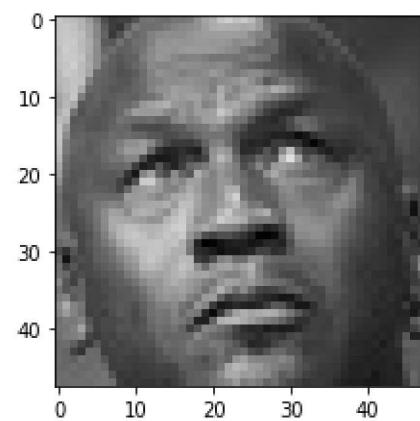
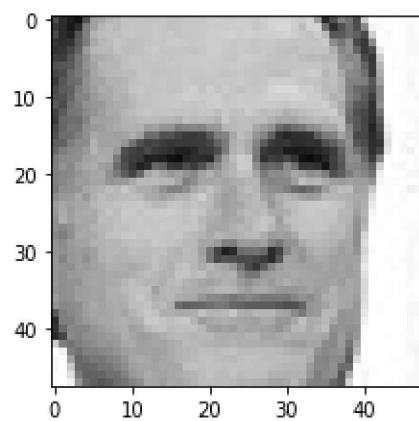
```
targetsize = 48
folder_path = "/content/dataset/"
```

▼ Plotting some of the images

```
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=(targetsize  
plt.imshow(img)  
plt.show()
```



Saving...

Here we are reading the data from the directory by using Image dat generator and flow from directory

```
batch_size = 64
```

```

datagen_train = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)

datagen_val = ImageDataGenerator(rescale = 1./255)

training_set = datagen_train.flow_from_directory(folder_path+"train",
                                                 target_size = (targetsize,targetsize),
                                                 batch_size=batch_size,
                                                 class_mode='categorical',
                                                 shuffle=True)

test_set = datagen_val.flow_from_directory(folder_path+"test",
                                           target_size = (targetsize,targetsize),
                                           batch_size=batch_size,
                                           class_mode='categorical',
                                           shuffle=False)

```

Found 21077 images belonging to 4 classes.
 Found 5140 images belonging to 4 classes.

▼ Model 1

```

vgg = VGG16(include_top=False, weights='imagenet')
vgg.trainable = False

```

Downloading data from <https://storage.googleapis.com/tensorflow/keras-applications/vgg1f>
 58892288/58889256 [=====] - 1s 0us/step
 58900480/58889256 [=====] - 1s 0us/step

Saving...

```

input_layer = Input(shape=(targetsize,targetsize,3,))
vgg_layer = vgg(input_layer)
layer1 = Conv2D(filters=256,kernel_size=(3,3),strides=(1,1),padding="same",activation='relu',
                kernel_initializer=tf.keras.initializers.he_normal(seed=20))(vgg_layer)
layer2 = MaxPool2D(pool_size=(2, 2), strides=(1,1), padding="same")(layer1)
flatten_layer = Flatten()(layer2)
layer3 = Dense(128,activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=1))
layer4 = Dense(64,activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=2))
output = Dense(4,activation='softmax',kernel_initializer=tf.keras.initializers.GlorotNormal(seed=3))
model_1 = Model(inputs=input_layer,outputs=output)
model_1.summary()

```

Model: "model"

Layer (type)	Output Shape	Param #
<hr/>		
input_2 (InputLayer)	[(None, 48, 48, 3)]	0
vgg16 (Functional)	(None, None, None, 512)	14714688
conv2d (Conv2D)	(None, 1, 1, 256)	1179904
max_pooling2d (MaxPooling2D)	(None, 1, 1, 256)	0
flatten (Flatten)	(None, 256)	0
dense (Dense)	(None, 128)	32896
dense_1 (Dense)	(None, 64)	8256
dense_2 (Dense)	(None, 4)	260
<hr/>		
Total params: 15,936,004		
Trainable params: 1,221,316		
Non-trainable params: 14,714,688		

```
# Clear any logs from previous runs
!rm -rf ./logs1/
```

```
model_1.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001), loss='categorical_crossentropy')

log_dir=os.path.join("logs1",datetime.datetime.now().strftime("%Y%m%d-%H%M%S"))
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir,histogram_freq=1, write_graph=True)

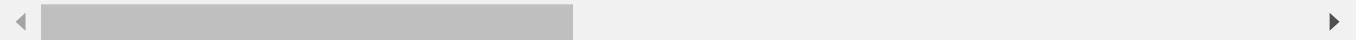
model_1.fit_generator(training_set, validation_data=test_set, epochs=25, steps_per_epoch=len(training_set))

WARNING:tensorflow:`write_grads` will be ignored in TensorFlow 2.0 for the `TensorBoard` /usr/local/lib/python3.7/dist-packages/keras/engine/training.py:1972: UserWarning: `Model` warnings.warn(``Model.fit_generator`` is deprecated and '
```

Saving... X

```
[=====] - 67s 117ms/step - loss: 1.2775 - accuracy: 0.4
330/330 [=====] - 37s 111ms/step - loss: 1.2320 - accuracy: 0.4
Epoch 3/25
330/330 [=====] - 36s 109ms/step - loss: 1.2125 - accuracy: 0.4
Epoch 4/25
330/330 [=====] - 36s 108ms/step - loss: 1.1998 - accuracy: 0.4
Epoch 5/25
330/330 [=====] - 35s 107ms/step - loss: 1.1863 - accuracy: 0.4
Epoch 6/25
330/330 [=====] - 37s 112ms/step - loss: 1.1759 - accuracy: 0.4
Epoch 7/25
330/330 [=====] - 37s 111ms/step - loss: 1.1647 - accuracy: 0.4
Epoch 8/25
330/330 [=====] - 36s 108ms/step - loss: 1.1543 - accuracy: 0.4
Epoch 9/25
330/330 [=====] - 35s 105ms/step - loss: 1.1437 - accuracy: 0.4
```

```
Epoch 10/25
330/330 [=====] - 36s 111ms/step - loss: 1.1361 - accuracy: 0.!
Epoch 11/25
330/330 [=====] - 36s 109ms/step - loss: 1.1312 - accuracy: 0.!
Epoch 12/25
330/330 [=====] - 36s 108ms/step - loss: 1.1191 - accuracy: 0.!
Epoch 13/25
330/330 [=====] - 37s 112ms/step - loss: 1.1105 - accuracy: 0.!
Epoch 14/25
330/330 [=====] - 37s 111ms/step - loss: 1.1048 - accuracy: 0.!
Epoch 15/25
330/330 [=====] - 36s 110ms/step - loss: 1.0974 - accuracy: 0.!
Epoch 16/25
330/330 [=====] - 36s 109ms/step - loss: 1.0916 - accuracy: 0.!
Epoch 17/25
330/330 [=====] - 35s 107ms/step - loss: 1.0858 - accuracy: 0.!
Epoch 18/25
330/330 [=====] - 37s 112ms/step - loss: 1.0758 - accuracy: 0.!
Epoch 19/25
330/330 [=====] - 37s 111ms/step - loss: 1.0669 - accuracy: 0.!
Epoch 20/25
330/330 [=====] - 36s 109ms/step - loss: 1.0599 - accuracy: 0.!
Epoch 21/25
330/330 [=====] - 36s 108ms/step - loss: 1.0584 - accuracy: 0.!
Epoch 22/25
330/330 [=====] - 35s 106ms/step - loss: 1.0505 - accuracy: 0.!
Epoch 23/25
330/330 [=====] - 35s 105ms/step - loss: 1.0452 - accuracy: 0.!
Epoch 24/25
330/330 [=====] - 36s 110ms/step - loss: 1.0329 - accuracy: 0.!
Epoch 25/25
330/330 [=====] - 36s 108ms/step - loss: 1.0230 - accuracy: 0.!
<keras.callbacks.History at 0x7f98e031b3d0>
```



```
%tensorboard --logdir logs1/
```



TensorBoard

SCALARS

GRAPHS

INACTIVE

- Show data download links
- Ignore outliers in chart scaling

Tooltip sorting method: default ▾

Smoothing



0.6

Horizontal Axis

STEP RELATIVE

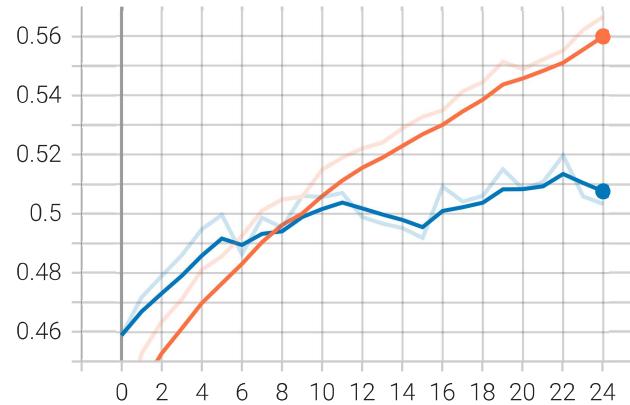
WALL

Runs

Filter tags (regular expressions supported)

epoch_accuracy

epoch_accuracy
tag: epoch_accuracy



20210917-034/20/train

Model 2

Saving...



batch_size = 64

```
datagen_train = ImageDataGenerator(rescale = 1./255,
                                    shear_range = 0.2,
                                    zoom_range = 0.2,
                                    horizontal_flip = True)
```

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

```
training_set = datagen_train.flow_from_directory(folder_path+"train",
                                                 target_size = (96,96),
                                                 batch_size=batch_size,
```

```
        class_mode='categorical',
        shuffle=True)
```

```
test_set = datagen_val.flow_from_directory(folder_path+"test",
                                            target_size = (96,96),
                                            batch_size=batch_size,
                                            class_mode='categorical',
                                            shuffle=False)
```

Found 21077 images belonging to 4 classes.
 Found 5140 images belonging to 4 classes.

```
classes = ["angry", "happy", "neutral", "sad"]
```

```
from tensorflow.keras.applications.inception_v3 import InceptionV3

inception = InceptionV3(input_shape=(96,96,3), weights='imagenet', include_top=False)
# don't train existing weights
for layer in inception.layers:
    layer.trainable = False

flatten = Flatten(data_format='channels_last', name='Flatten')(inception.output)

Out = Dense(units=len(classes), activation='softmax', kernel_initializer=tf.keras.initializers.

model2 = Model(inputs=inception.input, outputs=Out)
model2.summary()
```

batch_normalization_359 (BatchN (None, 1, 1, 384))	1152	conv2d_359[0][0]
batch_normalization_363 (BatchN (None, 1, 1, 384))	1152	conv2d_363[0][0]
activation_359 (Activation) (None, 1, 1, 384)	0	batch_normalization_
Saving... (None, 1, 1, 384)	0	batch_normalization_
conv2d_360 (Conv2D) (None, 1, 1, 384)	442368	activation_359[0][0]
conv2d_361 (Conv2D) (None, 1, 1, 384)	442368	activation_359[0][0]
conv2d_364 (Conv2D) (None, 1, 1, 384)	442368	activation_363[0][0]
conv2d_365 (Conv2D) (None, 1, 1, 384)	442368	activation_363[0][0]
average_pooling2d_34 (AveragePo (None, 1, 1, 1280))	0	mixed8[0][0]
conv2d_358 (Conv2D) (None, 1, 1, 320)	409600	mixed8[0][0]
batch_normalization_360 (BatchN (None, 1, 1, 384))	1152	conv2d_360[0][0]
batch_normalization_361 (BatchN (None, 1, 1, 384))	1152	conv2d_361[0][0]

30/09/2021, 23:38

Emotion_classification_updated.ipynb - Colaboratory

batch_normalization_364 (BatchN (None, 1, 1, 384)	1152	conv2d_364[0][0]
conv2d_366 (Conv2D) (None, 1, 1, 192)	245760	average_pooling2d_34
batch_normalization_358 (BatchN (None, 1, 1, 320)	960	conv2d_358[0][0]
activation_360 (Activation) (None, 1, 1, 384)	0	batch_normalization_360
activation_361 (Activation) (None, 1, 1, 384)	0	batch_normalization_361
activation_364 (Activation) (None, 1, 1, 384)	0	batch_normalization_364
activation_365 (Activation) (None, 1, 1, 384)	0	batch_normalization_365
batch_normalization_366 (BatchN (None, 1, 1, 192)	576	conv2d_366[0][0]
activation_358 (Activation) (None, 1, 1, 320)	0	batch_normalization_358
mixed9_0 (Concatenate) (None, 1, 1, 768)	0	activation_360[0][0] activation_361[0][0]
concatenate_6 (Concatenate) (None, 1, 1, 768)	0	activation_364[0][0] activation_365[0][0]
activation_366 (Activation) (None, 1, 1, 192)	0	batch_normalization_366
mixed9 (Concatenate) (None, 1, 1, 2048)	0	activation_358[0][0] mixed9_0[0][0] concatenate_6[0][0] activation_366[0][0]
conv2d_371 (Conv2D) (None, 1, 1, 448)	917504	mixed9[0][0]

```
# Clear any logs from previous runs
!rm -rf ./logs2/
```

Saving...

timizers.Adam(lr=0.001), loss='categorical_crossentropy', m

```
log_dir=os.path.join("logs2",datetime.datetime.now().strftime("%Y%m%d-%H%M%S"))
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir,histogram_freq=1, write
```

```
result = model2.fit_generator(training_set,validation_data=test_set,epochs=20,steps_per_epoch
```

```
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/optimizer_v2.py:356: UserWarning:
  "The `lr` argument is deprecated, use `learning_rate` instead.")
WARNING:tensorflow: `write_grads` will be ignored in TensorFlow 2.0 for the `TensorBoard`
```

/usr/local/lib/python3.7/dist-packages/keras/engine/training.py:1972: UserWarning: `Model`

warnings.warn(`Model.fit_generator` is deprecated and '

Epoch 1/20

330/330 [=====] - 83s 231ms/step - loss: 1.3134 - accuracy: 0.4

```
Epoch 2/20
330/330 [=====] - 78s 237ms/step - loss: 1.2416 - accuracy: 0.4
Epoch 3/20
330/330 [=====] - 75s 226ms/step - loss: 1.2212 - accuracy: 0.4
Epoch 4/20
330/330 [=====] - 74s 224ms/step - loss: 1.2004 - accuracy: 0.4
Epoch 5/20
330/330 [=====] - 73s 221ms/step - loss: 1.1934 - accuracy: 0.4
Epoch 6/20
330/330 [=====] - 73s 221ms/step - loss: 1.1990 - accuracy: 0.4
Epoch 7/20
330/330 [=====] - 74s 224ms/step - loss: 1.2044 - accuracy: 0.4
Epoch 8/20
330/330 [=====] - 74s 224ms/step - loss: 1.1904 - accuracy: 0.4
Epoch 9/20
330/330 [=====] - 72s 219ms/step - loss: 1.1810 - accuracy: 0.4
Epoch 10/20
330/330 [=====] - 73s 222ms/step - loss: 1.1825 - accuracy: 0.4
Epoch 11/20
330/330 [=====] - 72s 218ms/step - loss: 1.1755 - accuracy: 0.4
Epoch 12/20
330/330 [=====] - 73s 221ms/step - loss: 1.1834 - accuracy: 0.4
Epoch 13/20
330/330 [=====] - 72s 218ms/step - loss: 1.1833 - accuracy: 0.4
Epoch 14/20
330/330 [=====] - 73s 222ms/step - loss: 1.1750 - accuracy: 0.4
Epoch 15/20
330/330 [=====] - 73s 222ms/step - loss: 1.1760 - accuracy: 0.4
Epoch 16/20
330/330 [=====] - 72s 219ms/step - loss: 1.1734 - accuracy: 0.4
Epoch 17/20
330/330 [=====] - 73s 220ms/step - loss: 1.1772 - accuracy: 0.4
Epoch 18/20
330/330 [=====] - 72s 218ms/step - loss: 1.1740 - accuracy: 0.4
Epoch 19/20
330/330 [=====] - 72s 218ms/step - loss: 1.1812 - accuracy: 0.4
Epoch 20/20
330/330 [=====] - 72s 220ms/step - loss: 1.1735 - accuracy: 0.4
```



TensorBoard

SCALARS

GRAPHS

INACTIVE

- Show data download links
- Ignore outliers in chart scaling

Tooltip sorting method: default ▾

Smoothing



0.6

Horizontal Axis

STEP RELATIVE

WALL

Runs

Write a regex to filter runs

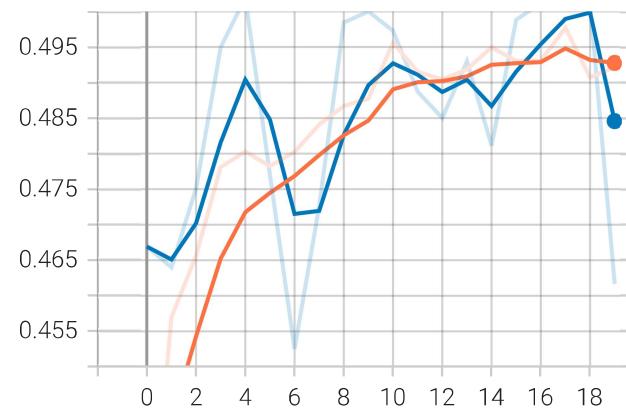
 20210917-043302/train 20210917-043302/validation

Saving...

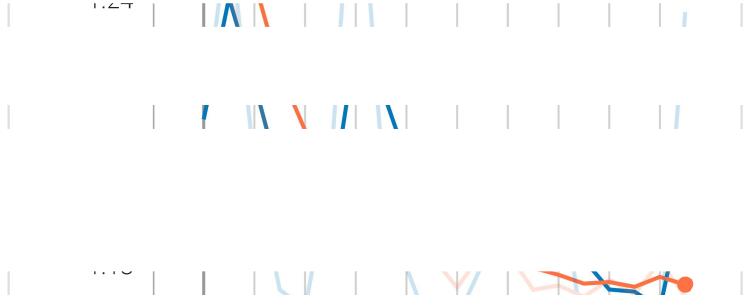


Filter tags (regular expressions supported)

epoch_accuracy

epoch_accuracy
tag: epoch_accuracy

epoch_loss

epoch_loss
tag: epoch_loss

▼ Model 3

batch_size = 64

```
datagen_train = ImageDataGenerator(rescale = 1./255,
                                    shear_range = 0.2,
                                    zoom_range = 0.2,
                                    horizontal_flip = True)
```

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

```
training_set = datagen_train.flow_from_directory(folder_path+"train",
                                                target_size = (targetsize,targetsize),
                                                batch_size=batch_size,
                                                class_mode='categorical',
                                                shuffle=True)
```

```
test_set = datagen_val.flow_from_directory(folder_path+"test",
                                            target_size = (targetsize,targetsize),
                                            batch_size=batch_size,
                                            class_mode='categorical',
                                            shuffle=False)
```

Found 21077 images belonging to 4 classes.

Found 5140 images belonging to 4 classes.

```
vgg = VGG16(include_top=False, weights='imagenet',input_shape=(48,48,3))
```

```
for layer in vgg.layers[:-6]:
```

```
    layer.trainable = False
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg1f/58892288/58889256 [=====] - 1s 0us/step
58900480/58889256 [=====] - 1s 0us/step
```

```
input_layer = Input(shape=(48,48,3,))
```

```
vgg_layer = vgg(input_layer)
```

```
layer1 = Conv2D(filters=1024,kernel_size=(7,7),strides=(1,1),padding="same",activation='relu'
                 kernel_initializer='he_normal')(vgg_layer)
```

```
layer2 = Conv2D(filters=1024,kernel_size=(1,1),strides=(1,1),padding="same",activation='relu'
                 kernel_initializer='he_normal')(layer1)
```

```
flatten_layer = Flatten()(layer2)
```

```
layer3 = Dense(128,activation='relu',kernel_initializer=tf.keras.initializers.he_normal(seed=1))
layer4 = Dense(64 activation='relu' kernel_initializer=tf.keras.initializers.he_normal(seed=2)
Saving...                                         ',kernel_initializer=tf.keras.initializers.GlorotNormal(s
outputs=output)
```

```
model_3.summary()
```

Model: "model"

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	[None, 48, 48, 3]	0
vgg16 (Functional)	(None, 1, 1, 512)	14714688
conv2d (Conv2D)	(None, 1, 1, 1024)	25691136
conv2d_1 (Conv2D)	(None, 1, 1, 1024)	1049600
flatten (Flatten)	(None, 1024)	0

dense (Dense)	(None, 128)	131200
dense_1 (Dense)	(None, 64)	8256
dense_2 (Dense)	(None, 4)	260
<hr/>		
Total params:	41,595,140	
Trainable params:	36,319,684	
Non-trainable params:	5,275,456	

```
!rm -rf ./logs3/
```

```
from keras.callbacks import ModelCheckpoint, EarlyStopping

filepath="/content/model_save/weights-{epoch:02d}-{val_accuracy:.4f}.hdf5"

checkpoint = ModelCheckpoint(filepath=filepath, monitor='val_loss', verbose=1, save_best_only=True)

model_3.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001), loss='categorical_crossentropy')

log_dir=os.path.join("logs3",datetime.datetime.now().strftime("%Y%m%d-%H%M%S"))
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir,histogram_freq=1, write_graph=False)

model_3.fit_generator(training_set,validation_data=test_set,epochs=25,steps_per_epoch=len(training_set))

Epoch 00001: saving model to /content/model_save/weights-00-0.6422.hdf5
Epoch 12/25
330/330 [=====] - 45s 137ms/step - loss: 0.8476 - accuracy: 0.8051

Epoch 00012: saving model to /content/model_save/weights-12-0.6422.hdf5
Epoch 13/25
330/330 [=====] - 46s 139ms/step - loss: 0.8205 - accuracy: 0.8051

Saving... × /content/model_save/weights-13-0.6607.hdf5
Epoch 00014: saving model to /content/model_save/weights-14-0.6488.hdf5
Epoch 15/25
330/330 [=====] - 45s 137ms/step - loss: 0.7944 - accuracy: 0.7916

Epoch 00015: saving model to /content/model_save/weights-15-0.6512.hdf5
Epoch 16/25
330/330 [=====] - 45s 137ms/step - loss: 0.7916 - accuracy: 0.7916

Epoch 00016: saving model to /content/model_save/weights-16-0.6535.hdf5
Epoch 17/25
330/330 [=====] - 46s 138ms/step - loss: 0.7736 - accuracy: 0.7736

Epoch 00017: saving model to /content/model_save/weights-17-0.6296.hdf5
Epoch 18/25
330/330 [=====] - 46s 138ms/step - loss: 0.7607 - accuracy: 0.7607
```

```
[=====] - 46s 138ms/step - loss: 0.7597 - accuracy:  
Epoch 00018: saving model to /content/model_save/weights-18-0.6508.hdf5  
Epoch 19/25  
330/330 [=====] - 46s 138ms/step - loss: 0.7587 - accuracy:  
  
Epoch 00019: saving model to /content/model_save/weights-19-0.6551.hdf5  
Epoch 20/25  
330/330 [=====] - 45s 136ms/step - loss: 0.7473 - accuracy:  
  
Epoch 00020: saving model to /content/model_save/weights-20-0.6685.hdf5  
Epoch 21/25  
330/330 [=====] - 45s 136ms/step - loss: 0.7425 - accuracy:  
  
Epoch 00021: saving model to /content/model_save/weights-21-0.6508.hdf5  
Epoch 22/25  
330/330 [=====] - 45s 136ms/step - loss: 0.7287 - accuracy:  
  
Epoch 00022: saving model to /content/model_save/weights-22-0.6551.hdf5  
Epoch 23/25  
330/330 [=====] - 45s 136ms/step - loss: 0.7260 - accuracy:  
  
Epoch 00023: saving model to /content/model_save/weights-23-0.6574.hdf5  
Epoch 24/25  
330/330 [=====] - 45s 136ms/step - loss: 0.7217 - accuracy:  
  
Epoch 00024: saving model to /content/model_save/weights-24-0.6564.hdf5  
Epoch 25/25  
330/330 [=====] - 45s 136ms/step - loss: 0.7059 - accuracy:  
  
Epoch 00025: saving model to /content/model_save/weights-25-0.6518.hdf5  
<keras.callbacks.History at 0x7f91b3eead50>
```

```
%tensorboard --logdir logs3/
```

Saving... X

TensorBoard

SCALARS

GRAPHS

INACTIVE

- Show data download links
- Ignore outliers in chart scaling

Tooltip sorting method: default ▾

Smoothing

0.6

Horizontal Axis

STEP RELATIVE

WALL

Runs

```
model_3.load_weights('/content/model_save/weights-20-0.6685.hdf5')
```

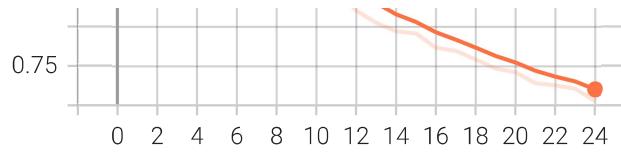
```
model_3.save('/content/saved_model/face_emotion.h5')
```

10

tag: epoch loss

Saving...

C6.U





Saving...

