Start coding or generate with AI.

```
!ls /content/
```

```
     sample_data  Test.zip  Train.zip
```

```
!mkdir '/content/Train'
```

```
!mv '/content/Train.zip' '/content/Train/'
```

```
!mkdir '/content/Test'
```

```
!mkdir '/content/model/'
```

```
!mv '/content/Test.zip' '/content/Test/'
```

```
!unzip '/content/Train/Train.zip'
```

```
           inflating: Twentynote/17.jpg
           inflating: Twentynote/2.jpg
           inflating: Twentynote/20.jpg
           inflating: Twentynote/23.jpg
           inflating: Twentynote/26.jpg
           inflating: Twentynote/28.jpg
           inflating: Twentynote/29.jpg
           inflating: Twentynote/3.jpg
           inflating: Twentynote/32.jpg
           inflating: Twentynote/33.jpg
           inflating: Twentynote/4.jpg
           inflating: Twentynote/6.jpg
           inflating: Twentynote/7.jpg
           inflating: Twentynote/8.jpg
           inflating: Twentynote/9.jpg
```

```
!unzip '/content/Test/Test.zip'
```

```
     Archive:  /content/Test/Test.zip
        creating: Test/1Hundrednote/
       inflating: Test/1Hundrednote/1.jpg
       inflating: Test/1Hundrednote/14.jpg
       inflating: Test/1Hundrednote/15.jpg
       inflating: Test/1Hundrednote/16.jpg
       inflating: Test/1Hundrednote/2.jpg
       inflating: Test/1Hundrednote/3.jpg
        creating: Test/2Hundrednote/
       inflating: Test/2Hundrednote/1.jpg
       inflating: Test/2Hundrednote/2.jpg
       inflating: Test/2Hundrednote/3.jpg
       inflating: Test/2Hundrednote/31.jpg
       inflating: Test/2Hundrednote/32.jpg
       inflating: Test/2Hundrednote/33.jpg
        creating: Test/2Thousandnote/
       inflating: Test/2Thousandnote/1.jpg
       inflating: Test/2Thousandnote/2.jpg
       inflating: Test/2Thousandnote/3.jpg
       inflating: Test/2Thousandnote/31.jpg
       inflating: Test/2Thousandnote/32.jpg
       inflating: Test/2Thousandnote/33.jpg
        creating: Test/5Hundrednote/
       inflating: Test/5Hundrednote/1.jpg
       inflating: Test/5Hundrednote/2.jpg
       inflating: Test/5Hundrednote/3.jpg
       inflating: Test/5Hundrednote/31.jpg
       inflating: Test/5Hundrednote/32.jpg
       inflating: Test/5Hundrednote/33.jpg
        creating: Test/Fiftynote/
       inflating: Test/Fiftynote/1.jpg
       inflating: Test/Fiftynote/2.jpg
       inflating: Test/Fiftynote/27.jpg
       inflating: Test/Fiftynote/28.jpg
       inflating: Test/Fiftynote/29.jpg
       inflating: Test/Fiftynote/3.jpg
        creating: Test/Tennote/
       inflating: Test/Tennote/1.jpg
       inflating: Test/Tennote/2.jpg
       inflating: Test/Tennote/3.jpg
       inflating: Test/Tennote/31.jpg
       inflating: Test/Tennote/32.jpg
       inflating: Test/Tennote/33.jpg
        creating: Test/Twentynote/
       inflating: Test/Twentynote/1.jpg
       inflating: Test/Twentynote/18.jpg
       inflating: Test/Twentynote/2.jpg
       inflating: Test/Twentynote/24.jpg
       inflating: Test/Twentynote/3.jpg
       inflating: Test/Twentynote/30.jpg
```

```
import os
import numpy as np
import matplotlib.pyplot as plt
import random
import cv2
import PIL
import glob
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.preprocessing.image import ImageDataGenerator


ROOTPATH = '/content'
DATAPATH= ROOTPATH+'/Train'
TRAINPATH = ROOTPATH+'/Train'
TESTPATH =  ROOTPATH+'/Test'
MODELPATH = ROOTPATH+'/model/'


_1Hundrednote=glob.glob(DATAPATH+'/1Hundrednote/*')# [/content/Test/1Hundrednote/1.jpg,/content/Test/1Hundrednote/2
_2Hundrednote=glob.glob(DATAPATH+'/2Hundrednote/*')
_2Thousandnote=glob.glob(DATAPATH+'/2Thousandnote/*')
_5Hundrednote=glob.glob(DATAPATH+'/5Hundrednote/*')
_Fiftynote=glob.glob(DATAPATH+'/Fiftynote/*')
_Tennote=glob.glob(DATAPATH+'/Tennote/*')
_Twentynote=glob.glob(DATAPATH+'/Twentynote/*')


print(len(_1Hundrednote),_1Hundrednote)
print(len(_2Hundrednote),_2Hundrednote)
print(len(_2Thousandnote),_2Thousandnote)
print(len(_5Hundrednote),_5Hundrednote)
print(len(_Fiftynote),_Fiftynote)
print(len(_Tennote),_Tennote)
print(len(_Twentynote),_Twentynote)
```

```
22 ['/content/Train/1Hundrednote/20.jpg', '/content/Train/1Hundrednote/25.jpg', '/content/Train/1Hundrednote/2
22 ['/content/Train/2Hundrednote/20.jpg', '/content/Train/2Hundrednote/25.jpg', '/content/Train/2Hundrednote/2
21 ['/content/Train/2Thousandnote/20.jpg', '/content/Train/2Thousandnote/25.jpg', '/content/Train/2Thousandnot
22 ['/content/Train/5Hundrednote/20.jpg', '/content/Train/5Hundrednote/25.jpg', '/content/Train/5Hundrednote/2
22 ['/content/Train/Fiftynote/20.jpg', '/content/Train/Fiftynote/25.jpg', '/content/Train/Fiftynote/2.jpg', '/
22 ['/content/Train/Tennote/20.jpg', '/content/Train/Tennote/25.jpg', '/content/Train/Tennote/1.jpg', '/conter
22 ['/content/Train/Twentynote/20.jpg', '/content/Train/Twentynote/2.jpg', '/content/Train/Twentynote/8.jpg',
```

```
dataset_classes=[_1Hundrednote,_2Hundrednote,_2Thousandnote,_5Hundrednote,_Fiftynote,_Tennote,_Twentynote]
total_class=len(dataset_classes)
print('Total dataset class: ',total_class)
```

```
Total dataset class:  7
```

```
IMAGE_SIZE=224
BATCH_SIZE=64

#pre_processing_training
train_datagen = ImageDataGenerator(
        rescale=1./255,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True,
        fill_mode='nearest',
        validation_split=0.2)

training_set = train_datagen.flow_from_directory(
        TRAINPATH,
        shuffle=True,
        target_size=(IMAGE_SIZE,IMAGE_SIZE),
        batch_size=BATCH_SIZE,
        class_mode='categorical',
        subset='training')


validation_set = train_datagen.flow_from_directory(
        TRAINPATH,
         shuffle=True,
        target_size=(IMAGE_SIZE,IMAGE_SIZE),
        batch_size=BATCH_SIZE,
        class_mode='categorical',
        subset='validation')

     Found 125 images belonging to 8 classes.
     Found 28 images belonging to 8 classes.


test_datagen = ImageDataGenerator(rescale=1./255)
test_set = test_datagen.flow_from_directory(
        TESTPATH,
         shuffle=False,
        target_size=(IMAGE_SIZE,IMAGE_SIZE),
        batch_size=BATCH_SIZE,
        class_mode='categorical')

     Found 42 images belonging to 8 classes.


training_set.class_indices

     {'.ipynb_checkpoints': 0,
      '1Hundrednote': 1,
      '2Hundrednote': 2,
      '2Thousandnote': 3,
      '5Hundrednote': 4,
      'Fiftynote': 5,
      'Tennote': 6,
      'Twentynote': 7}


validation_set.class_indices

     {'.ipynb_checkpoints': 0,
      '1Hundrednote': 1,
      '2Hundrednote': 2,
      '2Thousandnote': 3,
      '5Hundrednote': 4,
      'Fiftynote': 5,
      'Tennote': 6,
      'Twentynote': 7}
```

```
test_set.class_indices
```

```
{'.ipynb_checkpoints': 0,
 '1Hundrednote': 1,
 '2Hundrednote': 2,
 '2Thousandnote': 3,
 '5Hundrednote': 4,
 'Fiftynote': 5,
 'Tennote': 6,
 'Twentynote': 7}
```

```
total_class=len(training_set.class_indices)
print('Number of classes in dataset: ',total_class)
```

```
Number of classes in dataset:  8
```

```
x,y=training_set.next()
fig=plt.figure(figsize=(15,15))
rows=5
cols=5
for i in range(rows*cols):
    fig.add_subplot(rows,cols,i+1)
    image=x[i]
    plt.imshow(image)
    plt.title(np.argmax(y[i]))
    plt.xticks([])
    plt.yticks([])

plt.show()
```


7
2
1
4
1
6
2
6
7
5
4
2
5
5
6
6
3
2
1
5
2
6
1
5
7

```
x,y=validation_set.next()
fig=plt.figure(figsize=(15,15))
rows=5
cols=5
for i in range(rows*cols):
    fig.add_subplot(rows,cols,i+1)
    image=x[i]
    plt.imshow(image)
    plt.title(np.argmax(y[i]))
    plt.xticks([])
    plt.yticks([])

plt.show()
```


4
2
2
5
2
4
4
7
1
NEW 100 RS NO
5
7
6

Start coding or generate with AI.

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-45-c0d2964e3798> in <cell line: 4>()
      2
      3 # Add an extra dimension to the labels
----> 4 training_set = Reshape(target_shape=(9,))(training_set_labels)
      5 validation_set = Reshape(target_shape=(9,))(validation_set_labels)
      6

NameError: name 'training_set_labels' is not defined
```

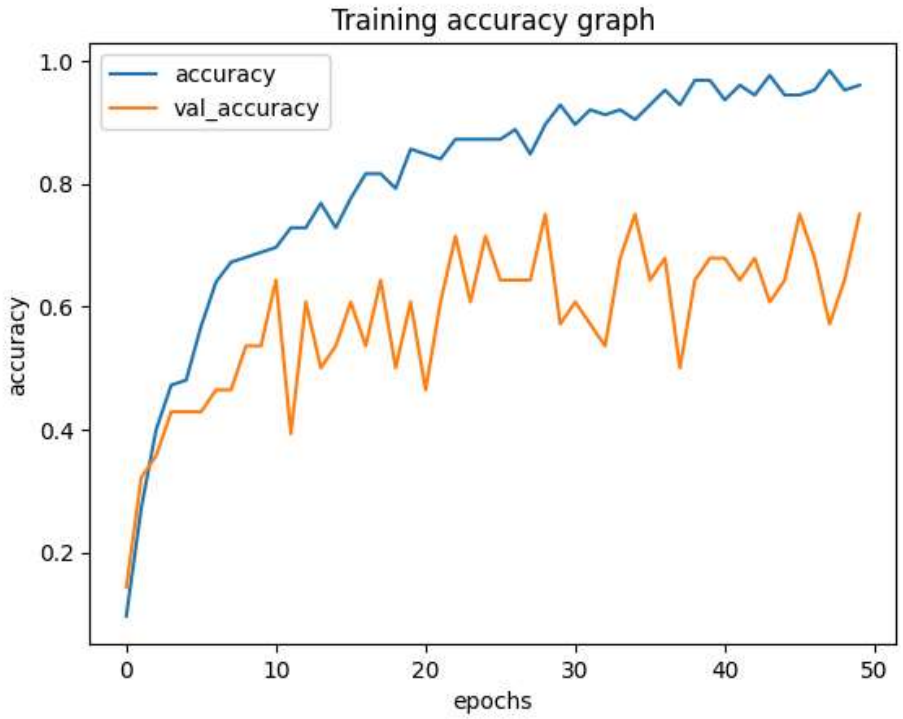
Next steps: Explain error

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.layers import Activation, Dense, Flatten


xception_model=tf.keras.applications.xception.Xception()


xception_model = Sequential()
pretrained_model=tf.keras.applications.xception.Xception(
    input_shape=(224,224,3),
    include_top=False, weights='imagenet', input_tensor=None, pooling='avg',
    classes=9
)
for layer in pretrained_model.layers:
        layer.trainable=False

xception_model.add(pretrained_model)


xception_model.add(Flatten())
xception_model.add(Dense(8, activation='softmax'))


from tensorflow.keras.utils import plot_model

# Plot the model and save it to an image file
plot_model(cnn, to_file='cnn_model.png', show_shapes=True, show_layer_names=True)


xception_model.summary()
```

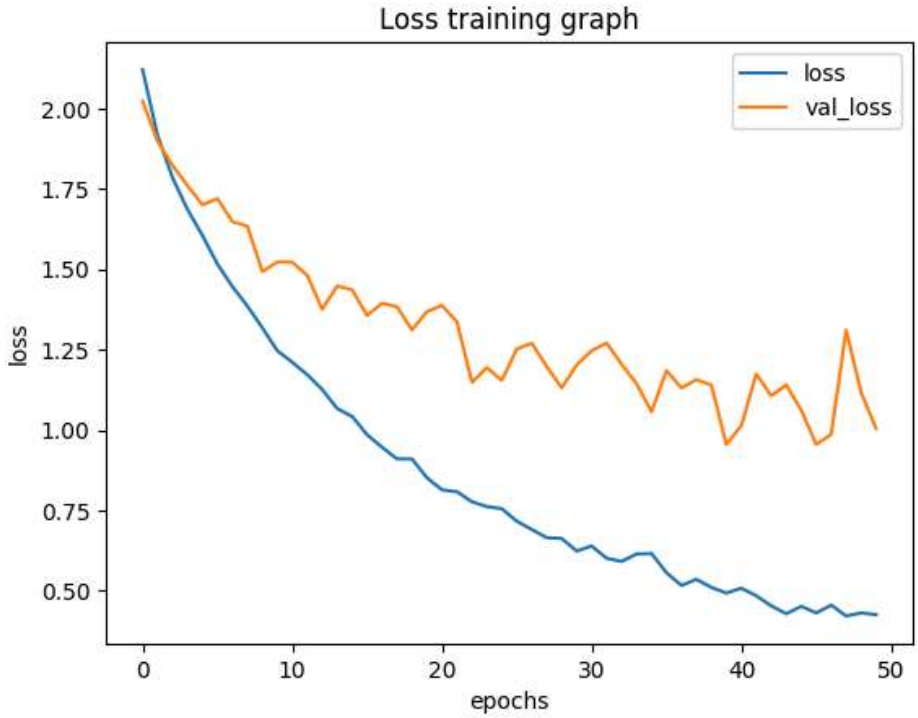
```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 xception (Functional)       (None, 2048)              20861480

 flatten_1 (Flatten)         (None, 2048)              0

 dense_1 (Dense)             (None, 8)                 16392

=================================================================
Total params: 20877872 (79.64 MB)
Trainable params: 16392 (64.03 KB)
Non-trainable params: 20861480 (79.58 MB)
_________________________________________________________________
```

```
xception_model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])


from tensorflow.keras.callbacks import ModelCheckpoint
checkpointer = ModelCheckpoint(filepath=MODELPATH+'Xception_Pretrained.model.best.hdf5', verbose=1 ,save_best_only


history=xception_model.fit(training_set,
                epochs=50,
                validation_data=validation_set,
                callbacks=[checkpointer])
```

```
plt.xlabel('epochs')
plt.ylabel('accuracy')
plt.title('Training accuracy graph')
plt.plot(history.history['accuracy'],label='accuracy')
plt.plot(history.history['val_accuracy'],label='val_accuracy')
plt.legend()
plt.show()
```




```
plt.xlabel('epochs')
plt.ylabel('loss')
plt.title('Loss training graph')
plt.plot(history.history['loss'],label='loss')
plt.plot(history.history['val_loss'],label='val_loss')
plt.legend()
plt.show()
```




```
test_loss,test_accuracy=xception_model.evaluate(test_set)
print('Test Loss: ',test_loss)
print('Test Accuracy: ',test_accuracy)
```

```
1/1 [==============================] - 11s 11s/step - loss: 1.0811 - accuracy: 0.5238
Test Loss:  1.0810688734054565
Test Accuracy:  0.523809552192688
```

```
print('Accuracy of the model is : ',test_accuracy*100)
```

```
Accuracy of the model is :  52.3809552192688
```

```
predicted_result=xception_model.predict(test_set)
predicted_result[:5]
```

```
1/1 [==============================] - 10s 10s/step
array([[8.7437184e-05, 8.1599891e-01, 2.6348304e-02, 3.3103991e-02,
        1.2296476e-02, 9.3727693e-02, 1.1403730e-02, 7.0334845e-03],
       [8.3704718e-04, 2.1739514e-01, 4.2387087e-02, 8.5137645e-03,
        1.7067313e-02, 3.5222021e-01, 2.5644672e-01, 1.0513276e-01],
       [4.2435099e-04, 6.6112593e-02, 4.7348522e-02, 1.2083506e-02,
        1.0549523e-02, 2.2320767e-01, 2.9068351e-01, 3.4959033e-01],
       [2.4712822e-04, 4.2886454e-01, 2.3970794e-02, 1.3554824e-02,
        1.5763137e-02, 5.0246185e-01, 7.7477610e-03, 7.3898565e-03],
       [2.6994621e-04, 8.2875651e-01, 8.6252922e-03, 8.4987283e-02,
        8.9824181e-03, 6.1165735e-02, 3.2335040e-03, 3.9791521e-03]],
      dtype=float32)
```

```
predicted_class=np.argmax(predicted_result,axis=-1)
predicted_class[:5]
```

```
array([1, 5, 7, 5, 1])
```

```
test_classes=test_set.classes
test_classes
```

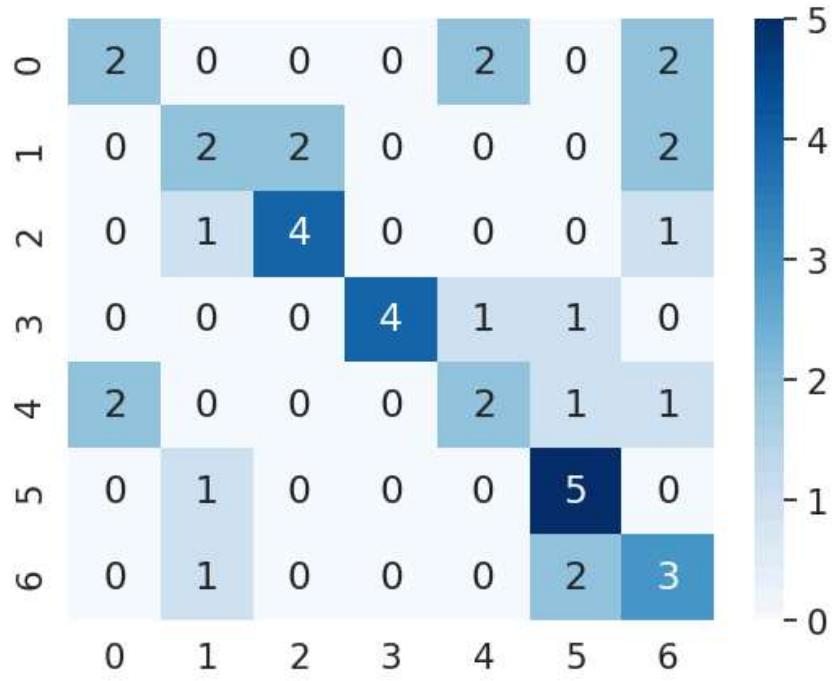
```
array([1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 4, 4, 4, 4,
       4, 4, 5, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 7],
      dtype=int32)
```

```
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(test_classes,predicted_class)
print(cm)
```

```
[[2 0 0 0 2 0 2]
 [0 2 2 0 0 0 2]
 [0 1 4 0 0 0 1]
 [0 0 0 4 1 1 0]
 [2 0 0 0 2 1 1]
 [0 1 0 0 0 5 0]
 [0 1 0 0 0 2 3]]
```

```
import seaborn as sns
sns.set(font_scale=1.4)
sns.heatmap(cm, annot=True,fmt='d',cmap="Blues")
```

```
<Axes: >
```




```
from sklearn.metrics import accuracy_score
print('Accuracy score: ',accuracy_score(test_classes,predicted_class))
```

```
Accuracy score:  0.5238095238095238
```

```
from sklearn.metrics import classification_report
print('Classification Report \n',classification_report(test_classes,predicted_class))
```

```
Classification Report
               precision    recall  f1-score   support

           1       0.50      0.33      0.40         6
           2       0.40      0.33      0.36         6
           3       0.67      0.67      0.67         6
           4       1.00      0.67      0.80         6
           5       0.40      0.33      0.36         6
           6       0.56      0.83      0.67         6
           7       0.33      0.50      0.40         6
```

```
          accuracy                           0.52        42
         macro avg       0.55      0.52      0.52        42
      weighted avg       0.55      0.52      0.52        42
```

```
import time
t = time.time()

export_path_keras = "/content/model/Model_5_xception_Pretrained{}_model_{}.h5".format(test_accuracy,int(t))
print(export_path_keras)
xception_model.save(export_path_keras)
```

```
/content/model/Model_5_xception_Pretrained0.523809552192688_model_1713519948.h5
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your mo
  saving_api.save_model(
```

```
from tensorflow.keras.models import load_model


model_path=export_path_keras
reload_model=load_model(model_path)
reload_model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 xception (Functional)       (None, 2048)              20861480

 flatten_1 (Flatten)         (None, 2048)              0

 dense_1 (Dense)             (None, 8)                 16392

=================================================================
Total params: 20877872 (79.64 MB)
Trainable params: 16392 (64.03 KB)
Non-trainable params: 20861480 (79.58 MB)
_________________________________________________________________
```

```
print(len(reload_model.weights))
print(reload_model.output_shape)
```

```
236
(None, 8)
```

```
reload_model.layers
```

```
[<keras.src.engine.functional.Functional at 0x7cd866de1750>,
 <keras.src.layers.reshaping.flatten.Flatten at 0x7cd847564f70>,
 <keras.src.layers.core.dense.Dense at 0x7cd847566da0>]
```

```
t = time.time()

export_path_sm = "/content/model/Model_5_xception_Pretrained {} Model {}".format(test_accuracy,int(t))
print(export_path_sm)

tf.saved_model.save(xception_model, export_path_sm)
```

```
/content/model/Model_5_xception_Pretrained 0.523809552192688 Model 1713519966
```

```
reload_tf_saved_model=tf.saved_model.load(export_path_sm)
```

```
reload_tf_saved_model.signatures['serving_default']
```

```
<ConcreteFunction (*, xception_input: TensorSpec(shape=(None, 224, 224, 3), dtype=tf.float32,
name='xception_input')) -> Dict[['dense_1', TensorSpec(shape=(None, 8), dtype=tf.float32, name='dense_1')]]
at 0x7CD846E23E80>
```

```
reload_tf_saved_model
```

```
<tensorflow.python.saved_model.load.Loader._recreate_base_user_object.<locals>._UserObject at 0x7cd84fa48670>
```

```
model=reload_model
```

```
!pip install pyttsx3
!pip install playsound
!pip install pyttsx3
!sudo apt-get install espeak
!ldconfig -p | grep libespeak.so.1
```

```
Reading state information... Done
The following additional packages will be installed:
  espeak-data libespeak1 libportaudio2 libsonic0
The following NEW packages will be installed:
  espeak espeak-data libespeak1 libportaudio2 libsonic0
0 upgraded, 5 newly installed, 0 to remove and 45 not upgraded.
Need to get 1,382 kB of archives.
After this operation, 3,178 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libportaudio2 amd64 19.6.0-1.1 [65.3 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/main amd64 libsonic0 amd64 0.2.0-11build1 [10.3 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/universe amd64 espeak-data amd64 1.48.15+dfsg-3 [1,085 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libespeak1 amd64 1.48.15+dfsg-3 [156 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy/universe amd64 espeak amd64 1.48.15+dfsg-3 [64.2 kB]
Fetched 1,382 kB in 26s (53.1 kB/s)
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend cannot be used. at /usr/
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (This frontend requires a controlling tty.)
debconf: falling back to frontend: Teletype
dpkg-preconfigure: unable to re-open stdin:
Selecting previously unselected package libportaudio2:amd64.
(Reading database ... 121752 files and directories currently installed.)
Preparing to unpack .../libportaudio2_19.6.0-1.1_amd64.deb ...
Unpacking libportaudio2:amd64 (19.6.0-1.1) ...
Selecting previously unselected package libsonic0:amd64.
Preparing to unpack .../libsonic0_0.2.0-11build1_amd64.deb ...
Unpacking libsonic0:amd64 (0.2.0-11build1) ...
```

```
/sbin/ldconfig.real: /usr/local/lib/libtbbbind.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbb.so.12 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_5.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_0.so.3 is not a symbolic link

        libespeak.so.1 (libc6,x86-64) => /lib/x86_64-linux-gnu/libespeak.so.1
```

```
import os
import pyttsx3
import numpy as np
import matplotlib.pyplot as plt
import random
import cv2
import PIL
import glob
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
```

```
checkpointer = ModelCheckpoint(filepath=MODELPATH+'5.Model_Xception_Pretrained.h5', verbose=1, save_best_only=True
```

```
export_path_keras = "/content/model/5.Model_Xception_Pretrained.h5"
```

```
print(export_path_keras)
tf.saved_model.save(xception_model, export_path_keras)
```

```
/content/model/5.Model_Xception_Pretrained.h5
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-96-b5083e8b0ea5> in <cell line: 2>()
      1 print(export_path_keras)
----> 2 tf.keras.models.save(xception_model, export_path_keras)

AttributeError: module 'keras.api._v2.keras.models' has no attribute 'save'
```

Next steps: Explain error

```
# MODELPATH=export_path_keras
# reload_model=tf.saved_model.load(MODELPATH)


# # Load the SavedModel
# #reload_model = tf.keras.models.load_model(MODELPATH)

# # Print the model summary
# reload_model.summary()


# MODELPATH = export_path_keras
# reload_model = tf.keras.models.load_model(MODELPATH)

# # Print the model summary
# reload_model.summary()
```

```
WARNING:tensorflow:SavedModel saved prior to TF 2.5 detected when loading Keras model.
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-103-47ef0a3afd98> in <cell line: 13>()
     11
     12 MODELPATH = export_path_keras
---> 13 reload_model = tf.keras.models.load_model(MODELPATH)
     14
     15 # Print the model summary

                                 2 frames
/usr/local/lib/python3.10/dist-packages/keras/src/saving/legacy/saved_model/load.py in
_read_legacy_metadata(object_graph_def, metadata, path)
    222         ):
    223             if not proto.user_object.metadata:
--> 224                 raise ValueError(
    225                     "Unable to create a Keras model from SavedModel at "
    226                     f"{path}. This SavedModel was exported with "

ValueError: Unable to create a Keras model from SavedModel at
/content/model/5.Model_Xception_Pretrained.h5. This SavedModel was exported with
`tf.saved_model.save`, and lacks the Keras metadata file. Please save your Keras model
by calling `model.save` or `tf.keras.models.save_model`. Note that you can still load
this SavedModel with `tf.saved_model.load`.
```

Next steps: Explain error

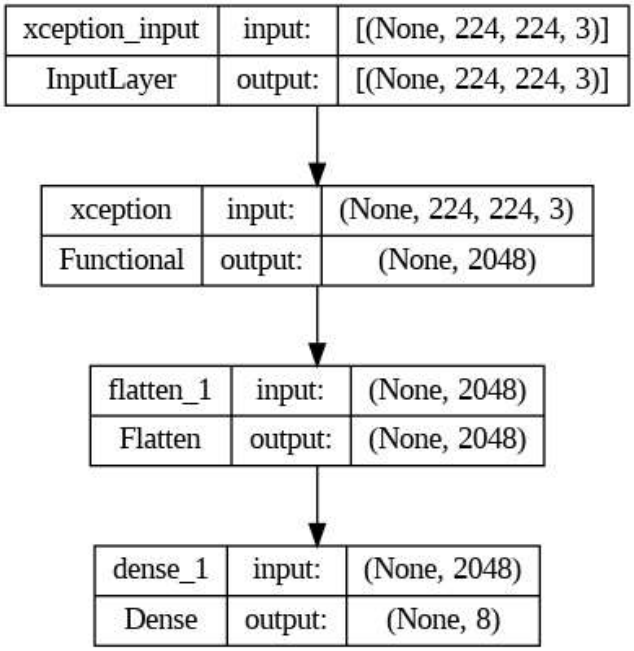
```
export_path_keras = "/content/model/5.Model_Xception_Pretrained_updated.h5"
print(export_path_keras)

xception_model.save(export_path_keras)
```

```
/content/model/5.Model_Xception_Pretrained_updated.h5
```

```
MODELPATH = export_path_keras
reload_model = tf.keras.models.load_model(MODELPATH)

# Print the model summary
reload_model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 xception (Functional)       (None, 2048)              20861480

 flatten_1 (Flatten)         (None, 2048)              0

 dense_1 (Dense)             (None, 8)                 16392

=================================================================
Total params: 20877872 (79.64 MB)
Trainable params: 16392 (64.03 KB)
Non-trainable params: 20861480 (79.58 MB)
```

```
from tensorflow.keras.utils import plot_model

# Plot the model and save it to an image file
plot_model(xception_model, to_file='cnn_model_Xception.png', show_shapes=True, show_layer_names=True)
```




```
def noteclass(cls):
    txt=pyttsx3.init()
    # if cls==0:
    #     ans="Two Taka"
    #     print(ans)
    #     txt.say(ans)
```