Transfer Learning using Xception

Gayanthika Shankar

School of Computing and Data Science

gayanthika.s-26@scds.saiuniversity.edu.in

Dataset Information

- Total Number of Images in the dataset: 1661 (.jpg format)
- Number of class: 3 (Airplanes, Motorbikes, Schooners)
- Number of images under Airplanes: 800
- Number of images under Motorbikes: 798
- Number of images under Schooners: 63
- Number if images used for training: 1246
- Number if images used for training: 415

This model contains BatchNormalization and Dropout of 25% drop rate before the output layer

Importing the required libraries

```
!nvidia-smi
```

```
Mon Jan  6 17:59:18 2025
+---------------------------------------------------------------------------------------+
| NVIDIA-SMI 535.104.05             Driver Version: 535.104.05   CUDA Version:
|-----------------------------------------+----------------------+----------------------
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Un
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  C
|                                         |                      |
|=========================================+======================+======================
|   0  Tesla T4                       Off | 00000000:00:04.0 Off |
| N/A   73C    P8              12W /  70W |      0MiB / 15360MiB |      0%
|                                         |                      |
+-----------------------------------------+----------------------+----------------------

+---------------------------------------------------------------------------------------+
| Processes:
|  GPU   GI   CI        PID   Type   Process name                                 G
|        ID   ID                                                                  U
|=======================================================================================
|  No running processes found
+---------------------------------------------------------------------------------------+
```

```
import zipfile
from google.colab import drive


drive.mount("/content/gdrive")
```

```
Mounted at /content/gdrive
```

```
#Importing necessary libraries and setting random seed
import tensorflow as tf
print(tf.__version__)

from tensorflow import keras
tf.random.set_seed(42)

import numpy as np
np.random.seed(42)

import matplotlib.pyplot as plt
%matplotlib inline

import glob
import PIL
from PIL import Image
```

```
2.17.1
```

```
from tqdm import tqdm
zip_ref = zipfile.ZipFile("/content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_S
for file in tqdm(zip_ref.namelist()):
 zip_ref.extract(file, "/content/gdrive/MyDrive/DL_Data/")
zip_ref.close()
```

```
100%|██████████| 3332/3332 [38:40<00:00,  1.44it/s]
```

```
#Importing the images
imgFiles = glob.glob("/content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooner
for items in imgFiles[:8]:
  print(items)
```

```
/content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooners/airplanes/image
/content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooners/airplanes/image
/content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooners/airplanes/image
/content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooners/airplanes/image
/content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooners/airplanes/image
/content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooners/airplanes/image
/content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooners/airplanes/image
/content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooners/airplanes/image
```

Preparing the dataset (preprocessing and labelling)

```
from tensorflow.keras.applications.xception import preprocess_input
from tensorflow.keras.preprocessing.image import load_img, img_to_array
X = []
y = []
for fName in imgFiles:
 img = load_img(fName, target_size=(299, 299))
 img_array = img_to_array(img)
 img_preprocessed = preprocess_input(img_array)
 X.append(img_preprocessed)
 label = fName.split("/")[-2]
 y.append(label)
#Convert lists to numpy arrays
X = np.array(X)
y = np.array(y)
```

```
#Check the first few entries and their type
print(f"Type of imgFiles: {type(imgFiles)}")
print(f"Length of imgFiles: {len(imgFiles)}")
print("First few entries:")
for f in list(imgFiles)[:3]:
    print(f"- {f}, type: {type(f)}")
```

```
Type of imgFiles: <class 'list'>
Length of imgFiles: 1661
First few entries:
- /content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooners/airplanes/ima
- /content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooners/airplanes/ima
- /content/gdrive/MyDrive/DL_Data/Airplanes_Motorbikes_Schooners/airplanes/ima
```

```
print(y)
```

```
['airplanes' 'airplanes' 'airplanes' ... 'Motorbikes' 'Motorbikes'
 'Motorbikes']
```

```
class_counts = dict()

#Count images for each class
for file_path in imgFiles:
    class_name = file_path.split("/")[-2]
    if class_name not in class_counts:
        class_counts[class_name] = 1
    else:
        class_counts[class_name] += 1

for class_name, count in class_counts.items():
    print(f"Class: {class_name}, Count: {count}")
```

```
Class: airplanes, Count: 800
Class: schooner, Count: 63
Class: Motorbikes, Count: 798
```

```
from sklearn.preprocessing import LabelEncoder
lEncoder = LabelEncoder()
y = lEncoder.fit_transform(y)

print(set(y))
print(lEncoder.classes_)
```

```
{0, 1, 2}
['Motorbikes' 'airplanes' 'schooner']
```

```
X = np.array(X)
y = np.array(y)

print(X.shape)
print(y.shape)
```

```
(1661, 299, 299, 3)
(1661,)
```

Split the dataset

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
                                                    stratify=y, random_state=42)


print("X_train_shape: {}".format(X_train.shape))
print("X_test_shape: {}".format(X_test.shape))
```

```
X_train_shape: (1245, 299, 299, 3)
X_test_shape: (416, 299, 299, 3)
```

```
#Standardize
mu = X_train.mean()
std = X_train.std()

X_train_std = (X_train-mu)/std
X_test_std = (X_test-mu)/std


X_train_std.shape
```

```
(1245, 299, 299, 3)
```

```
y_train.shape
```

```
(1245,)
```

```
X_train
```

```
array([[[ 0.99215686,  1.        ,  1.        ],
        [ 0.99215686,  1.        ,  1.        ],
        [ 0.99215686,  1.        ,  1.        ],
        ...,
        [ 1.        ,  1.        ,  1.        ],
        [ 1.        ,  1.        ,  1.        ],
        [ 1.        ,  1.        ,  1.        ]],

       [[ 0.99215686,  1.        ,  1.        ],
        [ 0.99215686,  1.        ,  1.        ],
        [ 0.99215686,  1.        ,  1.        ],
        ...,
        [ 1.        ,  1.        ,  1.        ],
        [ 1.        ,  1.        ,  1.        ],
        [ 1.        ,  1.        ,  1.        ]],
```

```
[[ 0.99215686,  1.        ,  1.        ],
 [ 0.99215686,  1.        ,  1.        ],
 [ 0.99215686,  1.        ,  1.        ],
 ...,
 [ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ]],

...,

[[ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ],
 ...,
 [ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ]],

[[ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ],
 ...,
 [ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ]],

[[ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ],
 ...,
 [ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ],
 [ 1.        ,  1.        ,  1.        ]]],


[[[ 0.99215686,  1.        ,  1.        ],
  [-0.6392157 , -0.6313726 , -0.6       ],
  [-0.6156863 , -0.60784316, -0.5764706 ],
  ...,
  [-0.5372549 , -0.46666664, -0.49019605],
  [-0.54509807, -0.47450978, -0.4980392 ],
  [-0.56078434, -0.49019605, -0.5137255 ]],

 [[ 0.99215686,  1.        ,  1.        ],
```

```
#Load a single image and check its values before preprocessing
img = load_img(imgFiles[0], target_size=(299, 299))
img_array = img_to_array(img)
print("Original image values range:", img_array.min(), "to", img_array.max())
print("Sample of original image values:\n", img_array[0:2, 0:2])

#Check after preprocessing
img_preprocessed = preprocess_input(img_array)
print("\nPreprocessed image values range:", img_preprocessed.min(), "to", img_pre
print("Sample of preprocessed values:\n", img_preprocessed[0:2, 0:2])
```

```
Original image values range: 0.0 to 255.0
Sample of original image values:
 [[[255. 255. 255.]
  [255. 255. 255.]]

 [[255. 255. 255.]
  [255. 255. 255.]]]

Preprocessed image values range: -1.0 to 1.0
Sample of preprocessed values:
 [[[1. 1. 1.]
  [1. 1. 1.]]

 [[1. 1. 1.]
  [1. 1. 1.]]]
```

```
print("X_train shape:", X_train.shape)
print("Value range:", X_train.min(), "to", X_train.max())
```

```
X_train shape: (1245, 299, 299, 3)
Value range: -1.0 to 1.0
```

```
import matplotlib.pyplot as plt

def show_images(X, n=5):
    plt.figure(figsize=(15, 3))
    for i in range(n):
        plt.subplot(1, n, i+1)
        #Convert back from preprocessed form for visualization
        img = (X[i] + 1) / 2  #Scale  range
        plt.imshow(img)
        plt.axis('off')
    plt.show()

show_images(X_train)
```

Requirements of the Xception Model:

- Image size: 299 x 299
- Expects RGB images (3 channels)
- Values should be in float32 format
- Scale pixel values to [-1,1]

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
import numpy as np
import os
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt

tf.random.set_seed(42)
np.random.seed(42)
```

Building the model

We only require Xceptions' Feature Extractor to build our model for this use-case. We refrain from tweaking the Feature Extractor (the base model) and therefore we freezeits weights beforehand so that they are not updated during backpropogation

```
import tensorflow as tf
print(tf.__version__)

from tensorflow import keras
tf.random.set_seed(42)

#Xception model

Xception_model = keras.applications.Xception(
    include_top=True,
    weights="imagenet",
    input_tensor=None,
    input_shape=None,
    pooling=None,
    classes=1000,
    classifier_activation="softmax",
    name="xception",
)
```

```
2.17.1
Downloading data from https://storage.googleapis.com/tensorflow/keras-applicat
91884032/91884032 ━━━━━━━━━━━━━━━━━━━━ 5s 0us/step
```

```
Xception_model.summary()
```

Model: "xception"

Layer (type)	Output Shape	Param #	Connec
input_layer (InputLayer)	(None, 299, 299, 3)	0	–
block1_conv1 (Conv2D)	(None, 149, 149, 32)	864	input_
block1_conv1_bn (BatchNormalization)	(None, 149, 149, 32)	128	block1
block1_conv1_act (Activation)	(None, 149, 149, 32)	0	block1
block1_conv2 (Conv2D)	(None, 147, 147, 64)	18,432	block1
block1_conv2_bn (BatchNormalization)	(None, 147, 147, 64)	256	block1
block1_conv2_act (Activation)	(None, 147, 147, 64)	0	block1
block2_sepconv1 (SeparableConv2D)	(None, 147, 147, 128)	8,768	block1
block2_sepconv1_bn (BatchNormalization)	(None, 147, 147, 128)	512	block2
block2_sepconv2_act (Activation)	(None, 147, 147, 128)	0	block2
block2_sepconv2 (SeparableConv2D)	(None, 147, 147, 128)	17,536	block2
block2_sepconv2_bn (BatchNormalization)	(None, 147, 147, 128)	512	block2
conv2d (Conv2D)	(None, 74, 74, 128)	8,192	block1
block2_pool (MaxPooling2D)	(None, 74, 74, 128)	0	block2
batch_normalization (BatchNormalization)	(None, 74, 74, 128)	512	conv2d
add (Add)	(None, 74, 74, 128)	0	block2 batch_
block3_sepconv1_act (Activation)	(None, 74, 74, 128)	0	add[0]
block3_sepconv1	(None, 74, 74, 256)	33,920	block3

(SeparableConv2D)			
block3_sepconv1_bn (BatchNormalization)	(None, 74, 74, 256)	1,024	block3
block3_sepconv2_act (Activation)	(None, 74, 74, 256)	0	block3
block3_sepconv2 (SeparableConv2D)	(None, 74, 74, 256)	67,840	block3
block3_sepconv2_bn (BatchNormalization)	(None, 74, 74, 256)	1,024	block3
conv2d_1 (Conv2D)	(None, 37, 37, 256)	32,768	add[0]
block3_pool (MaxPooling2D)	(None, 37, 37, 256)	0	block3
batch_normalization_1 (BatchNormalization)	(None, 37, 37, 256)	1,024	conv2d
add_1 (Add)	(None, 37, 37, 256)	0	block3 batch_
block4_sepconv1_act (Activation)	(None, 37, 37, 256)	0	add_1
block4_sepconv1 (SeparableConv2D)	(None, 37, 37, 728)	188,672	block4
block4_sepconv1_bn (BatchNormalization)	(None, 37, 37, 728)	2,912	block4
block4_sepconv2_act (Activation)	(None, 37, 37, 728)	0	block4
block4_sepconv2 (SeparableConv2D)	(None, 37, 37, 728)	536,536	block4
block4_sepconv2_bn (BatchNormalization)	(None, 37, 37, 728)	2,912	block4
conv2d_2 (Conv2D)	(None, 19, 19, 728)	186,368	add_1
block4_pool (MaxPooling2D)	(None, 19, 19, 728)	0	block4
batch_normalization_2 (BatchNormalization)	(None, 19, 19, 728)	2,912	conv2d
add_2 (Add)	(None, 19, 19, 728)	0	block4 batch_

block5_sepconv1_act (Activation)	(None, 19, 19, 728)	0	add_2
block5_sepconv1 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block5
block5_sepconv1_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block5
block5_sepconv2_act (Activation)	(None, 19, 19, 728)	0	block5
block5_sepconv2 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block5
block5_sepconv2_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block5
block5_sepconv3_act (Activation)	(None, 19, 19, 728)	0	block5
block5_sepconv3 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block5
block5_sepconv3_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block5
add_3 (Add)	(None, 19, 19, 728)	0	block5 add_2
block6_sepconv1_act (Activation)	(None, 19, 19, 728)	0	add_3
block6_sepconv1 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block6
block6_sepconv1_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block6
block6_sepconv2_act (Activation)	(None, 19, 19, 728)	0	block6
block6_sepconv2 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block6
block6_sepconv2_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block6
block6_sepconv3_act (Activation)	(None, 19, 19, 728)	0	block6
block6_sepconv3 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block6

block6_sepconv3_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block6
add_4 (Add)	(None, 19, 19, 728)	0	block6 add_3
block7_sepconv1_act (Activation)	(None, 19, 19, 728)	0	add_4
block7_sepconv1 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block7
block7_sepconv1_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block7
block7_sepconv2_act (Activation)	(None, 19, 19, 728)	0	block7
block7_sepconv2 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block7
block7_sepconv2_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block7
block7_sepconv3_act (Activation)	(None, 19, 19, 728)	0	block7
block7_sepconv3 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block7
block7_sepconv3_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block7
add_5 (Add)	(None, 19, 19, 728)	0	block7 add_4
block8_sepconv1_act (Activation)	(None, 19, 19, 728)	0	add_5
block8_sepconv1 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block8
block8_sepconv1_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block8
block8_sepconv2_act (Activation)	(None, 19, 19, 728)	0	block8
block8_sepconv2 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block8
block8_sepconv2_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block8

block8_sepconv3_act (Activation)	(None, 19, 19, 728)	0	block8
block8_sepconv3 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block8
block8_sepconv3_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block8
add_6 (Add)	(None, 19, 19, 728)	0	block8 add_5
block9_sepconv1_act (Activation)	(None, 19, 19, 728)	0	add_6
block9_sepconv1 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block9
block9_sepconv1_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block9
block9_sepconv2_act (Activation)	(None, 19, 19, 728)	0	block9
block9_sepconv2 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block9
block9_sepconv2_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block9
block9_sepconv3_act (Activation)	(None, 19, 19, 728)	0	block9
block9_sepconv3 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block9
block9_sepconv3_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block9
add_7 (Add)	(None, 19, 19, 728)	0	block9 add_6
block10_sepconv1_act (Activation)	(None, 19, 19, 728)	0	add_7
block10_sepconv1 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block1
block10_sepconv1_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block1
block10_sepconv2_act (Activation)	(None, 19, 19, 728)	0	block1

(Activation)			
block10_sepconv2 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block1
block10_sepconv2_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block1
block10_sepconv3_act (Activation)	(None, 19, 19, 728)	0	block1
block10_sepconv3 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block1
block10_sepconv3_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block1
add_8 (Add)	(None, 19, 19, 728)	0	block1 add_7
block11_sepconv1_act (Activation)	(None, 19, 19, 728)	0	add_8
block11_sepconv1 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block1
block11_sepconv1_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block1
block11_sepconv2_act (Activation)	(None, 19, 19, 728)	0	block1
block11_sepconv2 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block1
block11_sepconv2_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block1
block11_sepconv3_act (Activation)	(None, 19, 19, 728)	0	block1
block11_sepconv3 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block1
block11_sepconv3_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block1
add_9 (Add)	(None, 19, 19, 728)	0	block1 add_8
block12_sepconv1_act (Activation)	(None, 19, 19, 728)	0	add_9
block12_sepconv1	(None, 19, 19, 728)	536,536	block1

(SeparableConv2D)			
block12_sepconv1_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block1
block12_sepconv2_act (Activation)	(None, 19, 19, 728)	0	block1
block12_sepconv2 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block1
block12_sepconv2_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block1
block12_sepconv3_act (Activation)	(None, 19, 19, 728)	0	block1
block12_sepconv3 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block1
block12_sepconv3_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block1
add_10 (Add)	(None, 19, 19, 728)	0	block1 add_9
block13_sepconv1_act (Activation)	(None, 19, 19, 728)	0	add_10
block13_sepconv1 (SeparableConv2D)	(None, 19, 19, 728)	536,536	block1
block13_sepconv1_bn (BatchNormalization)	(None, 19, 19, 728)	2,912	block1
block13_sepconv2_act (Activation)	(None, 19, 19, 728)	0	block1
block13_sepconv2 (SeparableConv2D)	(None, 19, 19, 1024)	752,024	block1
block13_sepconv2_bn (BatchNormalization)	(None, 19, 19, 1024)	4,096	block1
conv2d_3 (Conv2D)	(None, 10, 10, 1024)	745,472	add_10
block13_pool (MaxPooling2D)	(None, 10, 10, 1024)	0	block1
batch_normalization_3 (BatchNormalization)	(None, 10, 10, 1024)	4,096	conv2d
add_11 (Add)	(None, 10, 10, 1024)	0	block1 batch

block14_sepconv1 (SeparableConv2D)	(None, 10, 10, 1536)	1,582,080	add_1
block14_sepconv1_bn (BatchNormalization)	(None, 10, 10, 1536)	6,144	block1
block14_sepconv1_act (Activation)	(None, 10, 10, 1536)	0	block1
block14_sepconv2 (SeparableConv2D)	(None, 10, 10, 2048)	3,159,552	block1
block14_sepconv2_bn (BatchNormalization)	(None, 10, 10, 2048)	8,192	block1
block14_sepconv2_act (Activation)	(None, 10, 10, 2048)	0	block1
avg_pool (GlobalAveragePooling2D)	(None, 2048)	0	block1
predictions (Dense)	(None, 1000)	2,049,000	avg_p

Total params: 22,910,480 (87.40 MB)
Trainable params: 22,855,952 (87.19 MB)
Non-trainable params: 54,528 (213.00 KB)

```
base_model1 = keras.applications.xception.Xception(weights='imagenet', input_shap
                                                   include_top=False)


for layer in base_model1.layers:
  layer.trainabe = False


#classifier

from tensorflow.keras import layers


x = keras.layers.GlobalAveragePooling2D()(base_model1.output)
x = layers.BatchNormalization()(x)
x = keras.layers.Dropout(0.25)(x)
output_ = layers.Dense(3, activation='softmax')(x)

model1 = keras.models.Model(inputs=[base_model1.input], outputs=[output_])
```

Compile and train the model

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import numpy as np

model1.compile(loss='sparse_categorical_crossentropy',
               optimizer='adam',
               metrics=['accuracy'])

callbacks2 = [keras.callbacks.ModelCheckpoint("best_xception_TL.weights.h5",
                                              monitor='val_accuracy',
                                              save_weights_only=True,
                                              save_best_only=True)]

datagen = ImageDataGenerator(
    rotation_range=20,  #randomly rotate images by up to 20 degrees
    width_shift_range=0.2,  #randomly shift images horizontally by up to 20%
    height_shift_range=0.2,  #randomly shift images vertically by up to 20%
    horizontal_flip=True,  #randomly flip images horizontally
    fill_mode='nearest',  #strategy for filling in newly created pixels
    validation_split=0.1 #10% validation split defined here
)

#Create train generator
train_generator = datagen.flow(
    X_train_std,
    y_train,
    batch_size=32,
    subset='training'    #Specify this is for training
)

#Create validation generator
validation_generator = datagen.flow(
    X_train_std,
    y_train,
    batch_size=32,
    subset='validation'  #Specify this is for validation
)

#Train the model
history = model1.fit(
    train_generator,
```

```
    train_generator,
    validation_data=validation_generator,
    epochs=10,
    callbacks=callbacks2
)
```

```
Epoch 1/10
/usr/local/lib/python3.10/dist-packages/keras/src/trainers/data_adapters/py_da
  self._warn_if_super_not_called()
36/36 ━━━━━━━━━━━━━━━━━━━━ 149s 2s/step - accuracy: 0.8763 - loss: 0.3704 - va
Epoch 2/10
36/36 ━━━━━━━━━━━━━━━━━━━━ 39s 956ms/step - accuracy: 0.9710 - loss: 0.0933 -
Epoch 3/10
36/36 ━━━━━━━━━━━━━━━━━━━━ 39s 1s/step - accuracy: 0.9547 - loss: 0.0851 - val
Epoch 4/10
36/36 ━━━━━━━━━━━━━━━━━━━━ 40s 990ms/step - accuracy: 0.9979 - loss: 0.0098 -
Epoch 5/10
36/36 ━━━━━━━━━━━━━━━━━━━━ 38s 958ms/step - accuracy: 0.9911 - loss: 0.0310 -
Epoch 6/10
36/36 ━━━━━━━━━━━━━━━━━━━━ 39s 963ms/step - accuracy: 0.9967 - loss: 0.0134 -
Epoch 7/10
36/36 ━━━━━━━━━━━━━━━━━━━━ 39s 960ms/step - accuracy: 0.9964 - loss: 0.0102 -
Epoch 8/10
36/36 ━━━━━━━━━━━━━━━━━━━━ 39s 961ms/step - accuracy: 0.9952 - loss: 0.0122 -
Epoch 9/10
36/36 ━━━━━━━━━━━━━━━━━━━━ 39s 960ms/step - accuracy: 0.9952 - loss: 0.0164 -
Epoch 10/10
36/36 ━━━━━━━━━━━━━━━━━━━━ 39s 959ms/step - accuracy: 0.9895 - loss: 0.0306 -
```

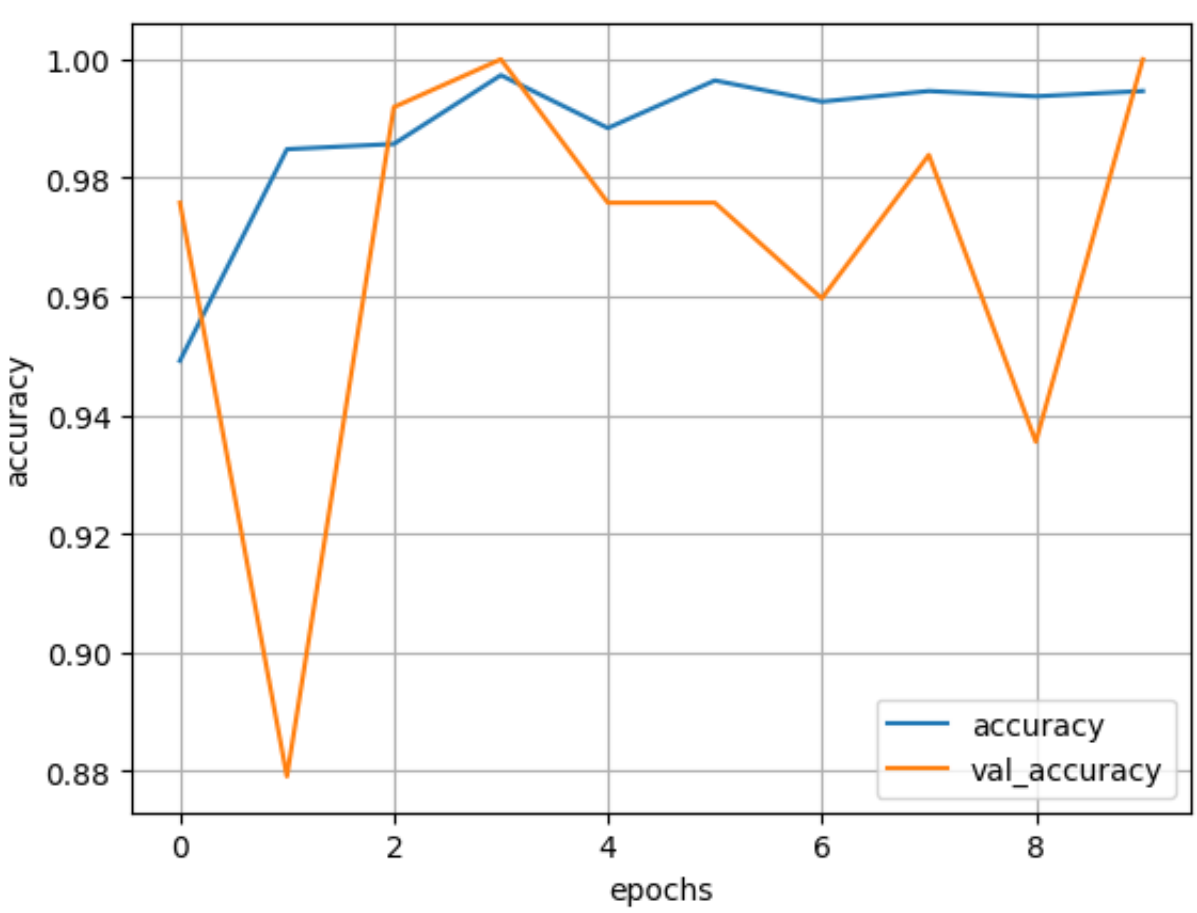
Accuracy Plot

```
keys = ['accuracy', 'val_accuracy']
progress = {k:v for k,v in history.history.items() if k in keys}

import pandas as pd
pd.DataFrame(progress).plot()

plt.xlabel("epochs")
plt.ylabel("accuracy")

plt.grid(True)
plt.show()
```



Evaluate

```
testLoss1, testAccuracy1 = model1.evaluate(x = X_test_std, y = y_test)
print("Test-loss: %f, Test-accuracy: %f" % (testLoss1, testAccuracy1))
```

```
13/13 ━━━━━━━━━━━━━━━━━━━━ 3s 253ms/step - accuracy: 0.9959 - loss: 0.0190
Test-loss: 0.031021, Test-accuracy: 0.992788
```

Update the model with the best weights

```
model1.load_weights("best_xception_TL.weights.h5")

testLoss1, testAccuracy1 = model1.evaluate(x = X_test_std, y = y_test)

print("Test-loss: %f, Test-accuracy: %f" % (testLoss1, testAccuracy1))
```

```
13/13 ━━━━━━━━━━━━━━━━━━━━ 3s 256ms/step - accuracy: 0.9971 - loss: 0.1240
Test-loss: 0.124476, Test-accuracy: 0.992788
```

Performance

```
y_prob = model1.predict(X_test_std)
y_predict = np.argmax(y_prob, axis=-1)
print(y_predict)
```

```
13/13 ━━━━━━━━━━━━━━━━━━━━ 5s 255ms/step
[1 1 0 0 0 0 0 1 0 0 0 1 0 0 1 1 1 1 0 1 1 1 1 1 1 0 0 0 1 1 0 1 0 0 0 0 1
 0 0 0 1 0 1 0 0 1 1 0 0 1 1 1 1 1 0 1 1 1 1 0 0 0 1 0 0 1 1 1 0 0 0 0 1 0
 0 1 1 0 1 0 1 0 2 1 0 1 0 0 0 0 1 1 1 0 0 2 0 0 1 0 0 0 0 0 1 1 1 1 1 0 0
 0 0 0 1 1 1 1 1 1 1 0 0 0 1 1 1 1 0 2 0 1 1 1 0 1 0 0 1 1 0 1 1 1 0 1 1 1
 1 0 1 0 1 1 1 1 0 2 1 0 0 1 0 1 1 0 0 0 1 0 0 1 1 1 0 1 0 0 0 0 0 0 0 1 1
 1 0 1 0 0 1 1 0 1 1 0 1 0 2 0 0 0 1 2 1 1 0 0 0 1 0 0 0 1 2 0 0 1 0 0 0 1
 0 1 0 0 0 0 1 0 1 1 0 1 0 1 1 1 1 1 1 1 0 0 0 0 0 0 0 1 0 1 0 1 1 1 0 0 1
 0 1 0 1 0 0 1 1 1 0 0 1 0 0 1 1 1 0 0 0 0 1 0 1 0 1 0 0 1 0 2 2 1 1 1 1 0
 0 2 1 0 0 1 0 1 1 0 0 1 0 1 1 1 0 0 1 1 0 2 1 2 0 0 0 0 1 0 1 0 0 0 1 1 2
 1 0 1 1 1 1 0 0 1 0 1 2 0 2 0 1 1 0 0 0 0 1 2 0 0 0 1 0 1 1 0 1 1 1 1 1 1
 0 0 1 1 0 0 1 0 1 2 1 0 0 0 0 0 1 1 0 1 1 0 2 1 1 1 1 1 1 1 0 1 0 1 1 0 1
 0 1 0 2 1 0 1 0 0]
```

```
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(y_true = y_test, y_pred = y_predict)
```

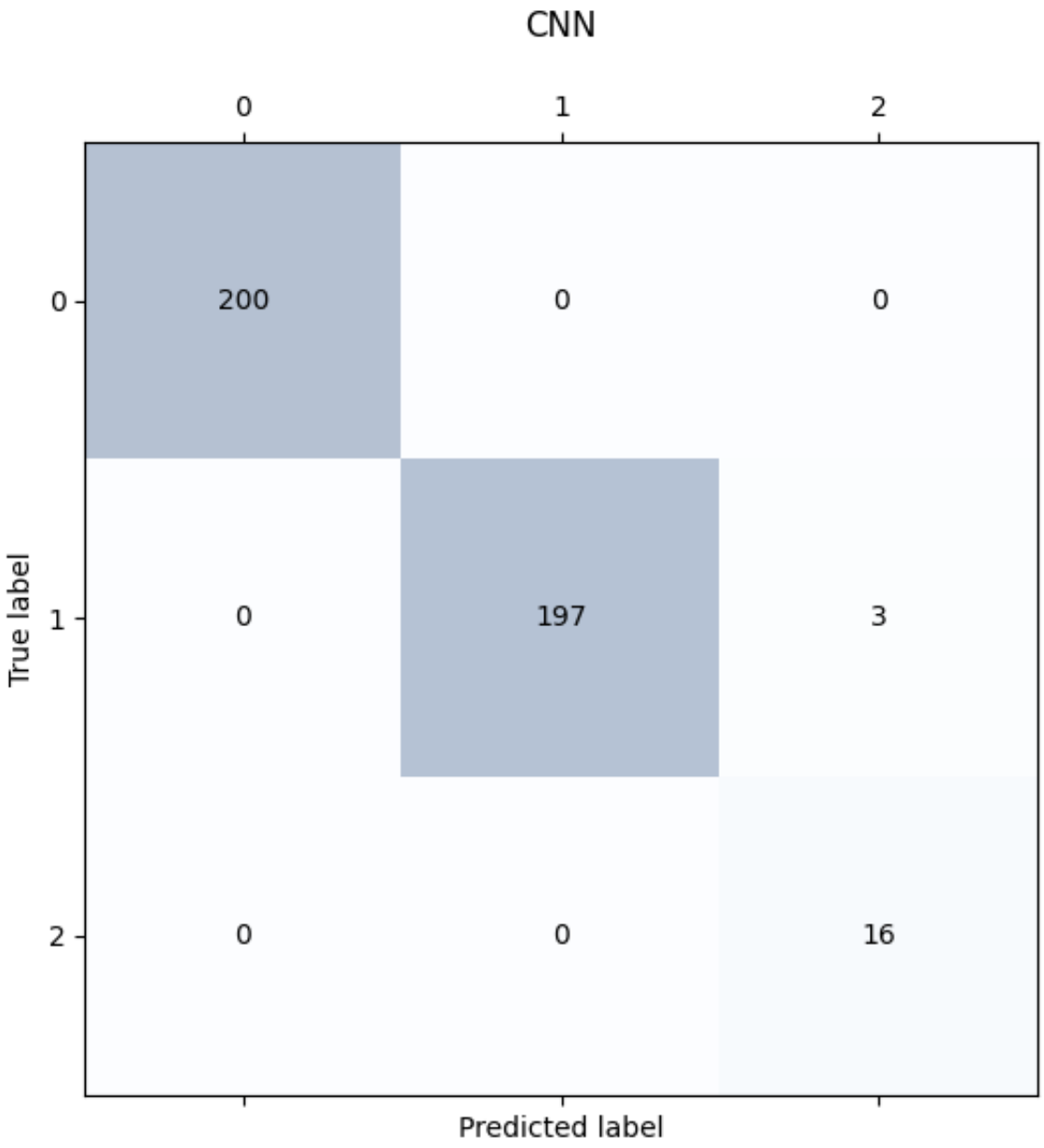
```
fig, ax = plt.subplots(figsize=(6, 6))
ax.matshow(cm, cmap=plt.cm.Blues, alpha=0.3)

for i in range(cm.shape[0]):
    for j in range(cm.shape[1]):
        ax.text(x=j, y=i, s=cm[i, j], va='center', ha='center')

ax.title.set_text('CNN\n')
plt.xlabel('Predicted label')
plt.ylabel('True label')

plt.tight_layout()
plt.savefig("ConfusionMatrixXception.png", dpi=300, format='png', pad_inches=0.3)
plt.show()

print(set(y))
print(lEncoder.classes_)
```




```
{0, 1, 2}
['Motorbikes' 'airplanes' 'schooner']
```

```
from sklearn.metrics import precision_score, recall_score, f1_score

pScore = precision_score(y_true= y_test, y_pred = y_predict, average = 'weighted'
print("Precision: ", pScore)

rScore = recall_score(y_true= y_test, y_pred = y_predict, average = 'weighted')
print("Recall: ", rScore)

fScore = f1_score(y_true= y_test, y_pred = y_predict, average = 'weighted')
print("F1-score: ", fScore)
```

```
Precision:  0.9939271255060729
Recall:  0.9927884615384616
F1-score:  0.9930702798460984
```

Save the model and datasets

```
model1.save('/content/gdrive/MyDrive/DL/Xception_Best_Model_TL.h5')
```

```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `
```

```
from numpy import save

save('/content/gdrive/MyDrive/DL/X_train_std.npy', X_train_std)
save('/content/gdrive/MyDrive/DL/X_test_std.npy', X_test_std)

save('/content/gdrive/MyDrive/DL/y_train.npy', y_train)
save('/content/gdrive/MyDrive/DL/y_test.npy', y_test)
```