PAGADALA NIKHIL KRISHNA

HU21CSEN0300328

Dataset taken from Kaggle

https://www.kaggle.com/datasets/clorichel/boat-types-recognition

Dataset created by taking three classes: Boats,Gondola and Ship

Acessing Google Drive

In [ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

DATASET LOADING AND SPLITTING

In [ ]:
```
import tensorflow as tf
import tensorflow_datasets as tfds
```

In [ ]:
```
dataset = tf.keras.preprocessing.image_dataset_from_directory("/content/drive/MyDrive/


ds_train = tf.keras.preprocessing.image_dataset_from_directory(
  directory = "/content/drive/MyDrive/SEM 6/Deep Learning/usecase1" ,
  validation_split=0.2,
  subset="training",
  seed=123
)
ds_test = tf.keras.preprocessing.image_dataset_from_directory(
  directory = "/content/drive/MyDrive/SEM 6/Deep Learning/usecase1",
  validation_split=0.2,
  subset="validation",
  seed=123)
```

```
Found 610 files belonging to 3 classes.
Found 610 files belonging to 3 classes.
Using 488 files for training.
Found 610 files belonging to 3 classes.
Using 122 files for validation.
```

In [ ]:
```
batch_size = 128
dataset_name = dataset
class_name = dataset.class_names
class_name
```

Out[ ]:
```
['Boat', 'Gondala', 'Ship']
```

IMAGE PREPROCESSING

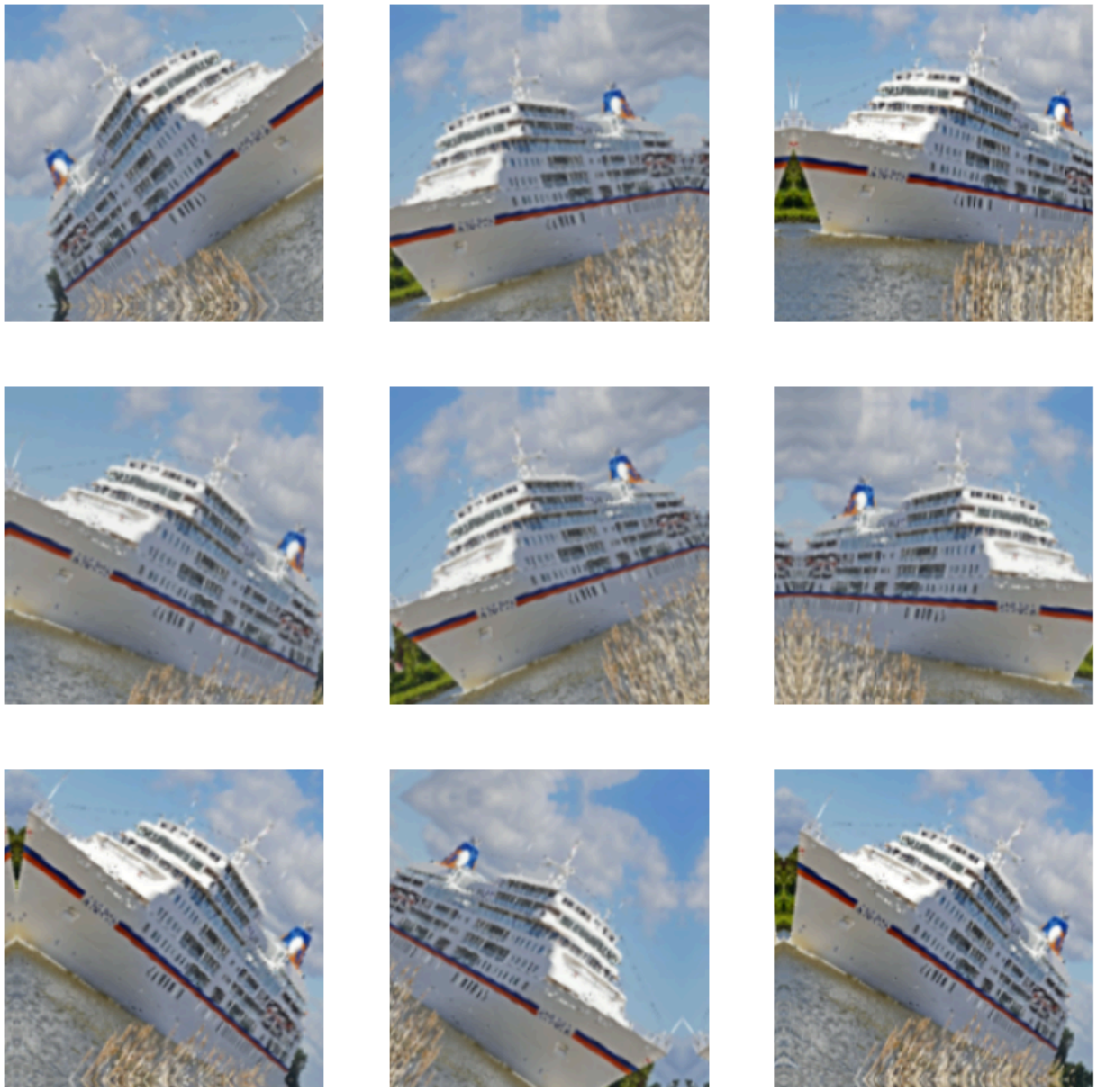
In [ ]:
```
size = (224,224)
ds_train = ds_train.map(lambda image , label : (tf.image.resize(image,size),label))
ds_test = ds_test.map(lambda image , label : (tf.image.resize(image,size),label))
```

In [ ]:
```
import matplotlib.pyplot as plt

plt.figure(figsize = (10,10))
for images, labels in ds_train.take(1):
  for i in range(9):
    ax = plt.subplot(3,3,i+1)
    plt.imshow(images[i].numpy().astype("uint8"))
    plt.title(class_name[labels[i]])
    plt.axis("off")
```

In [ ]:
```
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers

image = Sequential(
    [
        layers.RandomFlip("horizontal"),
        layers.RandomRotation(0.1),
        layers.RandomZoom(height_factor = (-0.2,-0.3),interpolation = "bilinear"),
        layers.RandomContrast(factor=0.1),
        layers.RandomTranslation(height_factor=0.1,width_factor = 0.1),
    ],
    name = "image"
)
```

In [ ]:
```
import numpy as np

for images,label in ds_train.take(1):
  plt.figure(figsize = (10,10))
  first_image= images[0]
```

```
  def f(x):
    return np.int(x)
  f2 = np.vectorize(f)
  for i in range(9):
    ax = plt.subplot(3,3,i+1)
    augmented_image = image(tf.expand_dims(first_image,0),training = True)
    plt.imshow(augmented_image[0].numpy().astype("int32"))
    plt.axis("off")
```

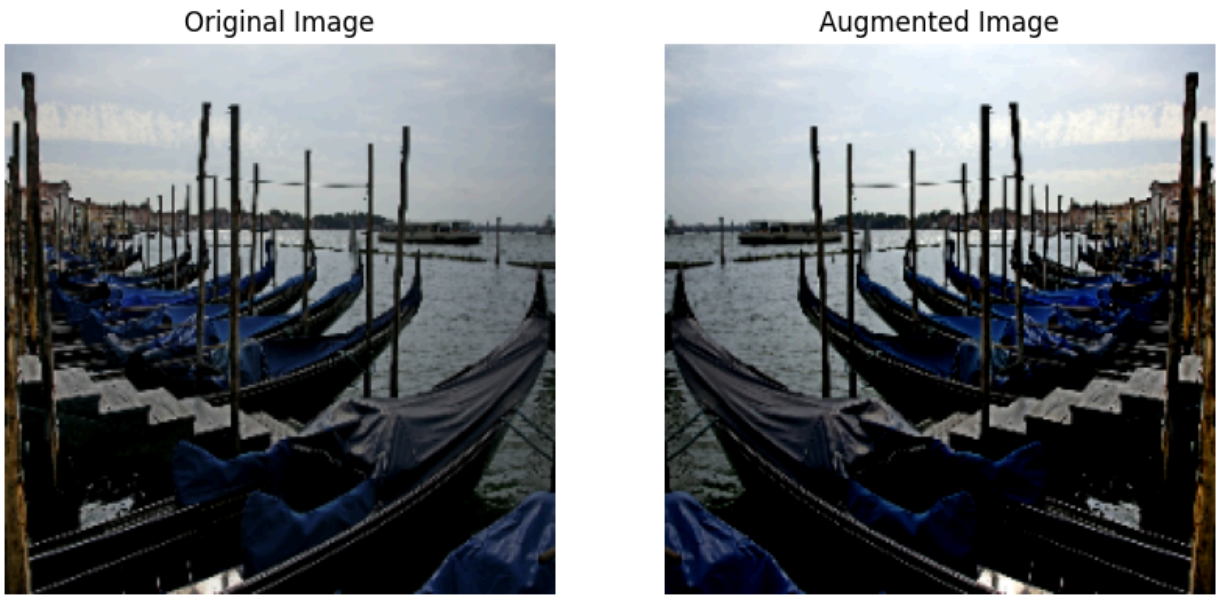


In [ ]:

```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf

for images, labels in ds_train.take(1):
    plt.figure(figsize=(10, 10))

    #Creating random index to choose any image from the folder
    random_index = np.random.randint(0, len(images))

    # Selecting a random image
    selected_image = images[random_index]
```

```
    # Function for image augmentation
    def image_augmentation(image):
        # Some augmentation operations as needed
        augmented_image = tf.image.random_flip_left_right(image)
        augmented_image = tf.image.random_flip_up_down(augmented_image)
        augmented_image = tf.image.random_brightness(augmented_image, max_delta=0.2)
        augmented_image = tf.image.random_contrast(augmented_image, lower=0.5, upper=1


        return augmented_image

    # Apply augmentation to the selected image
    augmented_image = image_augmentation(selected_image)

    # Displaying both original and augmented images for finding difference
    plt.subplot(1, 2, 1)
    plt.imshow(selected_image.numpy().astype("int32"))
    plt.title('Original Image')
    plt.axis('off')

    plt.subplot(1, 2, 2)
    plt.imshow(augmented_image.numpy().astype("int32"))
    plt.title('Augmented Image')
    plt.axis('off')

    plt.show()
```

```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB d
ata ([0..1] for floats or [0..255] for integers).
```

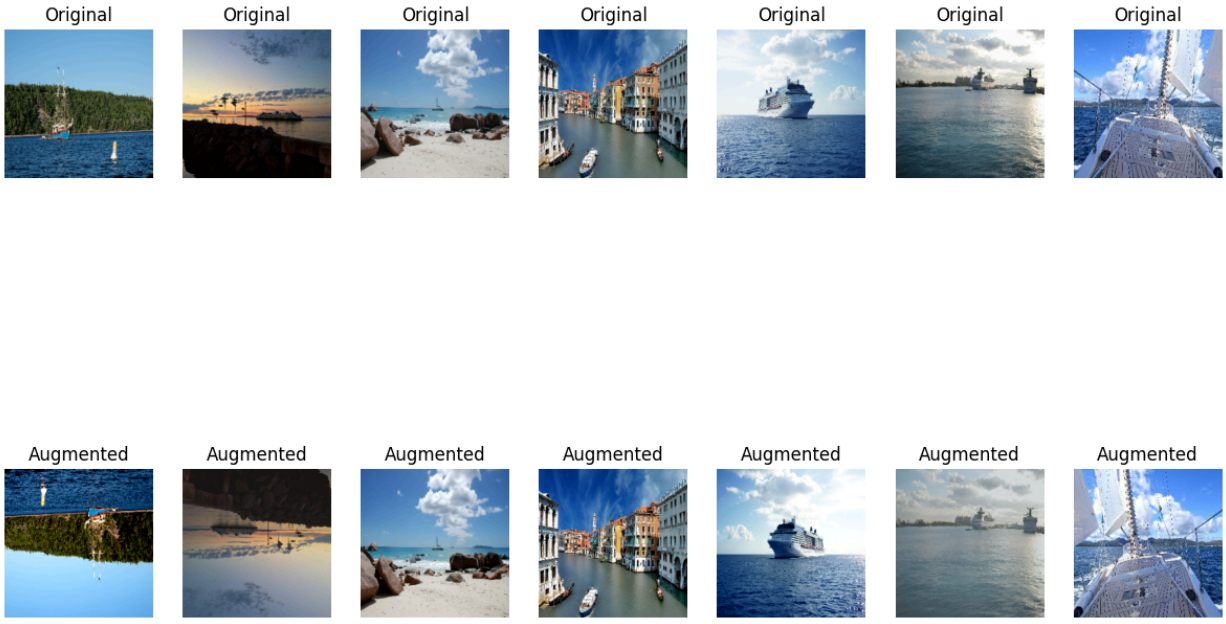


In [ ]:

```
for images, labels in ds_train.take(1):
    plt.figure(figsize=(15, 10))

    # Choosing 5-7 random indexs
    num_images = min(7, len(images))
    random_indices = np.random.choice(len(images), size=num_images, replace=False)

    # Function for image augmentation
    def image_augmentation(image):
        # Some augmentation operations
```

```
        augmented_image = tf.image.random_flip_left_right(image)
        augmented_image = tf.image.random_flip_up_down(augmented_image)
        augmented_image = tf.image.random_brightness(augmented_image, max_delta=0.2)
        augmented_image = tf.image.random_contrast(augmented_image, lower=0.5, upper=1


        return augmented_image

    # Displaying both original and augmented images for finding difference
    for i, random_index in enumerate(random_indices):
        plt.subplot(2, num_images, i + 1)
        selected_image = images[random_index]
        augmented_image = image_augmentation(selected_image)

        plt.imshow(selected_image.numpy().astype("int32"))
        plt.title(f'Original')
        plt.axis('off')

        plt.subplot(2, num_images, i + num_images + 1)
        plt.imshow(augmented_image.numpy().astype("int32"))
        plt.title(f'Augmented')
        plt.axis('off')

    plt.show()
```

```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB d
ata ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB d
ata ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB d
ata ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB d
ata ([0..1] for floats or [0..255] for integers).
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB d
ata ([0..1] for floats or [0..255] for integers).
```



MOBILENET MODEL

In [ ]:
```
!pip install split-folders
```

```
Collecting split-folders
  Downloading split_folders-0.5.1-py3-none-any.whl (8.4 kB)
Installing collected packages: split-folders
Successfully installed split-folders-0.5.1
```

Splitting the data into train,test and validation set using splitfolders

In [ ]:
```
import splitfolders

splitfolders.ratio("/content/drive/MyDrive/SEM 6/Deep Learning/usecase1", output="/cor
```

```
Copying files: 610 files [00:56, 10.79 files/s]
```

In [ ]:
```
from tensorflow.keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator, load_img
```

In [ ]:
```
datagen = ImageDataGenerator()

#TRAIN SET
train_generator = datagen.flow_from_directory(
    directory="/content/drive/MyDrive/SEM 6/Deep Learning/usecase1_split/train",
    classes = class_name,
    target_size=(200, 200),
    batch_size=32,
    class_mode="binary",
)

#VALIDATION SET
valid_generator = datagen.flow_from_directory(
    directory="/content/drive/MyDrive/SEM 6/Deep Learning/usecase1_split/val",
    classes = class_name,
    target_size=(224, 224),
    batch_size=32,
    class_mode="binary",
)

#TEST SET
test_generator = datagen.flow_from_directory(
    directory="/content/drive/MyDrive/SEM 6/Deep Learning/usecase1_split/test",
    classes = class_name,
    target_size=(224, 224),
    batch_size=32,
    class_mode="binary",
)
```

```
Found 427 images belonging to 3 classes.
Found 122 images belonging to 3 classes.
Found 61 images belonging to 3 classes.
```

MOBILENET Model

In [ ]:
```
import tensorflow as tf
from tensorflow.keras.applications import MobileNet
from tensorflow.keras import layers, models
```

In [ ]:
```
#MobileNet base model
base_model = MobileNet(input_shape=(224, 224, 3), include_top=False)

# Freeze the layers of the base model
for layer in base_model.layers:
    layer.trainable = False

# Creating layers for classification
x = layers.GlobalAveragePooling2D()(base_model.output)
x = layers.Dense(128, activation='relu')(x)
x = layers.Dropout(0.5)(x)
output = layers.Dense(len(class_name), activation='softmax')(x)

#Final model
model = models.Model(inputs=base_model.input, outputs=output)

#Compilation of  the model
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accu

# Display Model summary
model.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/mo
bilenet/mobilenet_1_0_224_tf_no_top.h5
17225924/17225924 [==============================] - 0s 0us/step
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 conv1 (Conv2D)              (None, 112, 112, 32)      864

 conv1_bn (BatchNormalizati  (None, 112, 112, 32)      128
 on)

 conv1_relu (ReLU)           (None, 112, 112, 32)      0

 conv_dw_1 (DepthwiseConv2D  (None, 112, 112, 32)      288
 )

 conv_dw_1_bn (BatchNormali  (None, 112, 112, 32)      128
 zation)

 conv_dw_1_relu (ReLU)       (None, 112, 112, 32)      0

 conv_pw_1 (Conv2D)          (None, 112, 112, 64)      2048

 conv_pw_1_bn (BatchNormali  (None, 112, 112, 64)      256
 zation)

 conv_pw_1_relu (ReLU)       (None, 112, 112, 64)      0

 conv_pad_2 (ZeroPadding2D)  (None, 113, 113, 64)      0

 conv_dw_2 (DepthwiseConv2D  (None, 56, 56, 64)        576
 )

 conv_dw_2_bn (BatchNormali  (None, 56, 56, 64)        256
 zation)

 conv_dw_2_relu (ReLU)       (None, 56, 56, 64)        0

 conv_pw_2 (Conv2D)          (None, 56, 56, 128)       8192

 conv_pw_2_bn (BatchNormali  (None, 56, 56, 128)       512
 zation)

 conv_pw_2_relu (ReLU)       (None, 56, 56, 128)       0

 conv_dw_3 (DepthwiseConv2D  (None, 56, 56, 128)       1152
 )

 conv_dw_3_bn (BatchNormali  (None, 56, 56, 128)       512
 zation)

 conv_dw_3_relu (ReLU)       (None, 56, 56, 128)       0

 conv_pw_3 (Conv2D)          (None, 56, 56, 128)       16384

 conv_pw_3_bn (BatchNormali  (None, 56, 56, 128)       512
 zation)
```

```
conv_pw_3_relu (ReLU)       (None, 56, 56, 128)       0

conv_pad_4 (ZeroPadding2D)  (None, 57, 57, 128)       0

conv_dw_4 (DepthwiseConv2D  (None, 28, 28, 128)       1152
)

conv_dw_4_bn (BatchNormali  (None, 28, 28, 128)       512
zation)

conv_dw_4_relu (ReLU)       (None, 28, 28, 128)       0

conv_pw_4 (Conv2D)          (None, 28, 28, 256)       32768

conv_pw_4_bn (BatchNormali  (None, 28, 28, 256)       1024
zation)

conv_pw_4_relu (ReLU)       (None, 28, 28, 256)       0

conv_dw_5 (DepthwiseConv2D  (None, 28, 28, 256)       2304
)

conv_dw_5_bn (BatchNormali  (None, 28, 28, 256)       1024
zation)

conv_dw_5_relu (ReLU)       (None, 28, 28, 256)       0

conv_pw_5 (Conv2D)          (None, 28, 28, 256)       65536

conv_pw_5_bn (BatchNormali  (None, 28, 28, 256)       1024
zation)

conv_pw_5_relu (ReLU)       (None, 28, 28, 256)       0

conv_pad_6 (ZeroPadding2D)  (None, 29, 29, 256)       0

conv_dw_6 (DepthwiseConv2D  (None, 14, 14, 256)       2304
)

conv_dw_6_bn (BatchNormali  (None, 14, 14, 256)       1024
zation)

conv_dw_6_relu (ReLU)       (None, 14, 14, 256)       0

conv_pw_6 (Conv2D)          (None, 14, 14, 512)       131072

conv_pw_6_bn (BatchNormali  (None, 14, 14, 512)       2048
zation)

conv_pw_6_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_7 (DepthwiseConv2D  (None, 14, 14, 512)       4608
)

conv_dw_7_bn (BatchNormali  (None, 14, 14, 512)       2048
zation)

conv_dw_7_relu (ReLU)       (None, 14, 14, 512)       0
```

```
conv_pw_7 (Conv2D)          (None, 14, 14, 512)       262144

conv_pw_7_bn (BatchNormali  (None, 14, 14, 512)       2048
zation)

conv_pw_7_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_8 (DepthwiseConv2D  (None, 14, 14, 512)       4608
)

conv_dw_8_bn (BatchNormali  (None, 14, 14, 512)       2048
zation)

conv_dw_8_relu (ReLU)       (None, 14, 14, 512)       0

conv_pw_8 (Conv2D)          (None, 14, 14, 512)       262144

conv_pw_8_bn (BatchNormali  (None, 14, 14, 512)       2048
zation)

conv_pw_8_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_9 (DepthwiseConv2D  (None, 14, 14, 512)       4608
)

conv_dw_9_bn (BatchNormali  (None, 14, 14, 512)       2048
zation)

conv_dw_9_relu (ReLU)       (None, 14, 14, 512)       0

conv_pw_9 (Conv2D)          (None, 14, 14, 512)       262144

conv_pw_9_bn (BatchNormali  (None, 14, 14, 512)       2048
zation)

conv_pw_9_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_10 (DepthwiseConv2  (None, 14, 14, 512)       4608
D)

conv_dw_10_bn (BatchNormal  (None, 14, 14, 512)       2048
ization)

conv_dw_10_relu (ReLU)      (None, 14, 14, 512)       0

conv_pw_10 (Conv2D)         (None, 14, 14, 512)       262144

conv_pw_10_bn (BatchNormal  (None, 14, 14, 512)       2048
ization)

conv_pw_10_relu (ReLU)      (None, 14, 14, 512)       0

conv_dw_11 (DepthwiseConv2  (None, 14, 14, 512)       4608
D)

conv_dw_11_bn (BatchNormal  (None, 14, 14, 512)       2048
ization)

conv_dw_11_relu (ReLU)      (None, 14, 14, 512)       0
```

```
conv_pw_11 (Conv2D)         (None, 14, 14, 512)       262144

conv_pw_11_bn (BatchNormal  (None, 14, 14, 512)       2048
ization)

conv_pw_11_relu (ReLU)      (None, 14, 14, 512)       0

conv_pad_12 (ZeroPadding2D  (None, 15, 15, 512)       0
)

conv_dw_12 (DepthwiseConv2  (None, 7, 7, 512)         4608
D)

conv_dw_12_bn (BatchNormal  (None, 7, 7, 512)         2048
ization)

conv_dw_12_relu (ReLU)      (None, 7, 7, 512)         0

conv_pw_12 (Conv2D)         (None, 7, 7, 1024)        524288

conv_pw_12_bn (BatchNormal  (None, 7, 7, 1024)        4096
ization)

conv_pw_12_relu (ReLU)      (None, 7, 7, 1024)        0

conv_dw_13 (DepthwiseConv2  (None, 7, 7, 1024)        9216
D)

conv_dw_13_bn (BatchNormal  (None, 7, 7, 1024)        4096
ization)

conv_dw_13_relu (ReLU)      (None, 7, 7, 1024)        0

conv_pw_13 (Conv2D)         (None, 7, 7, 1024)        1048576

conv_pw_13_bn (BatchNormal  (None, 7, 7, 1024)        4096
ization)

conv_pw_13_relu (ReLU)      (None, 7, 7, 1024)        0

global_average_pooling2d (  (None, 1024)              0
GlobalAveragePooling2D)

dense (Dense)               (None, 128)               131200

dropout (Dropout)           (None, 128)               0

dense_1 (Dense)             (None, 3)                 387

=================================================================
Total params: 3360451 (12.82 MB)
Trainable params: 131587 (514.01 KB)
Non-trainable params: 3228864 (12.32 MB)
_________________________________________________________________
```

In [ ]:
```
epochs = 10
history = model.fit(train_generator, epochs=epochs, validation_data=valid_generator,ba
```

```
Epoch 1/10
14/14 [==============================] - 15s 684ms/step - loss: 1.4756 - accuracy: 0.
4309 - val_loss: 0.6791 - val_accuracy: 0.7049
Epoch 2/10
14/14 [==============================] - 8s 557ms/step - loss: 0.7587 - accuracy: 0.6
768 - val_loss: 0.6235 - val_accuracy: 0.7541
Epoch 3/10
14/14 [==============================] - 7s 482ms/step - loss: 0.6431 - accuracy: 0.7
283 - val_loss: 0.5617 - val_accuracy: 0.8033
Epoch 4/10
14/14 [==============================] - 8s 573ms/step - loss: 0.5363 - accuracy: 0.7
822 - val_loss: 0.5315 - val_accuracy: 0.8033
Epoch 5/10
14/14 [==============================] - 7s 477ms/step - loss: 0.4918 - accuracy: 0.8
126 - val_loss: 0.5106 - val_accuracy: 0.8115
Epoch 6/10
14/14 [==============================] - 8s 590ms/step - loss: 0.4527 - accuracy: 0.8
478 - val_loss: 0.5080 - val_accuracy: 0.8033
Epoch 7/10
14/14 [==============================] - 7s 483ms/step - loss: 0.4341 - accuracy: 0.8
454 - val_loss: 0.4869 - val_accuracy: 0.8443
Epoch 8/10
14/14 [==============================] - 8s 580ms/step - loss: 0.3667 - accuracy: 0.8
689 - val_loss: 0.4762 - val_accuracy: 0.8525
Epoch 9/10
14/14 [==============================] - 8s 562ms/step - loss: 0.3451 - accuracy: 0.8
735 - val_loss: 0.4705 - val_accuracy: 0.8443
Epoch 10/10
14/14 [==============================] - 7s 492ms/step - loss: 0.3240 - accuracy: 0.8
829 - val_loss: 0.4593 - val_accuracy: 0.8361
```

In [ ]:
```
test_loss, test_acc = model.evaluate(test_generator)
print("The test loss is: ", test_loss)
print("The best accuracy is: ", test_acc*100)
```

```
2/2 [==============================] - 2s 1s/step - loss: 0.5064 - accuracy: 0.8033
The test loss is:  0.5064443349838257
The best accuracy is:  80.32786846160889
```

In [ ]:
```
Labels = ["Boat","Gondala","Ship"]
```

In [ ]:
```
def preprocess_image(image):
    img_array = tf.keras.preprocessing.image.img_to_array(image)
    img_array = tf.expand_dims(img_array, 0)
    img_array = tf.keras.applications.mobilenet.preprocess_input(img_array)
    return img_array

#Predictions for images in TEST DATA
for batch in ds_test.take(5):  # Take the first 5 batchs
    # Extract images and labels from the batch
    images, labels = batch

    # Preprocess the images for prediction
    preprocessed_images = tf.map_fn(preprocess_image, images)

    # Reshaping the input tensor to match the MobileNet model's input shape
    preprocessed_images = tf.reshape(preprocessed_images, (-1, 224, 224, 3))

    #Making predictions
    predictions = model.predict(preprocessed_images)
```

```
    class_labels = Labels #Category's names
    predicted_classes = np.argmax(predictions, axis=1)

    # Displaying each test image along with the original and predicted class
    for i in range(images.shape[0]):
        plt.imshow(tf.keras.preprocessing.image.array_to_img(images[i]))
        plt.title(f"Original Class: {class_labels[labels[i]]}\nPredicted Class: {class
        plt.axis("off")
        plt.show()
```

```
1/1 [==============================] - 0s 29ms/step
```

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Gondala

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Boat

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Boat

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Gondala

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Gondala

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Gondala
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Boat

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

```
1/1 [==============================] - 0s 53ms/step
```

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Gondala

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Boat

Original Class: Boat
Predicted Class: Boat

Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Gondala
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Ship

Original Class: Boat
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Boat
Predicted Class: Ship

Original Class: Boat
Predicted Class: Boat

```
1/1 [==============================] - 0s 40ms/step
```

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Gondala
Predicted Class: Ship

Original Class: Boat
Predicted Class: Ship

Original Class: Boat
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Gondala

Original Class: Ship
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Boat
Predicted Class: Ship

Original Class: Ship
Predicted Class: Ship

```
1/1 [==============================] - 0s 47ms/step
```

Original Class: Gondala
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Boat
Predicted Class: Boat

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Gondala
Predicted Class: Gondala

Original Class: Gondala
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Boat
Predicted Class: Boat

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Boat

Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Gondala
Predicted Class: Gondala

Original Class: Ship
Predicted Class: Ship

Original Class: Boat
Predicted Class: Boat

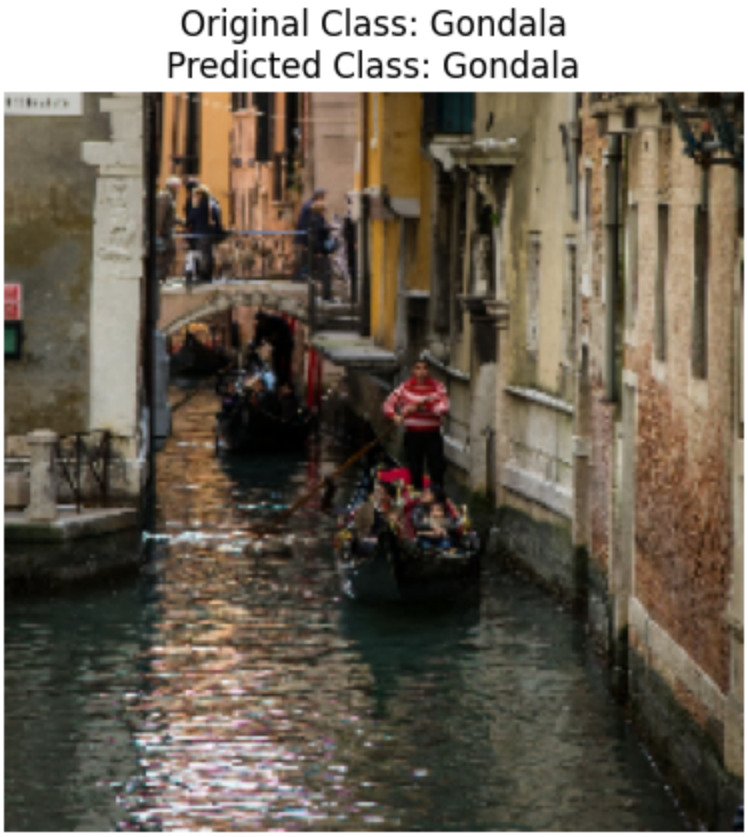
Original Class: Gondala
Predicted Class: Gondala

Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

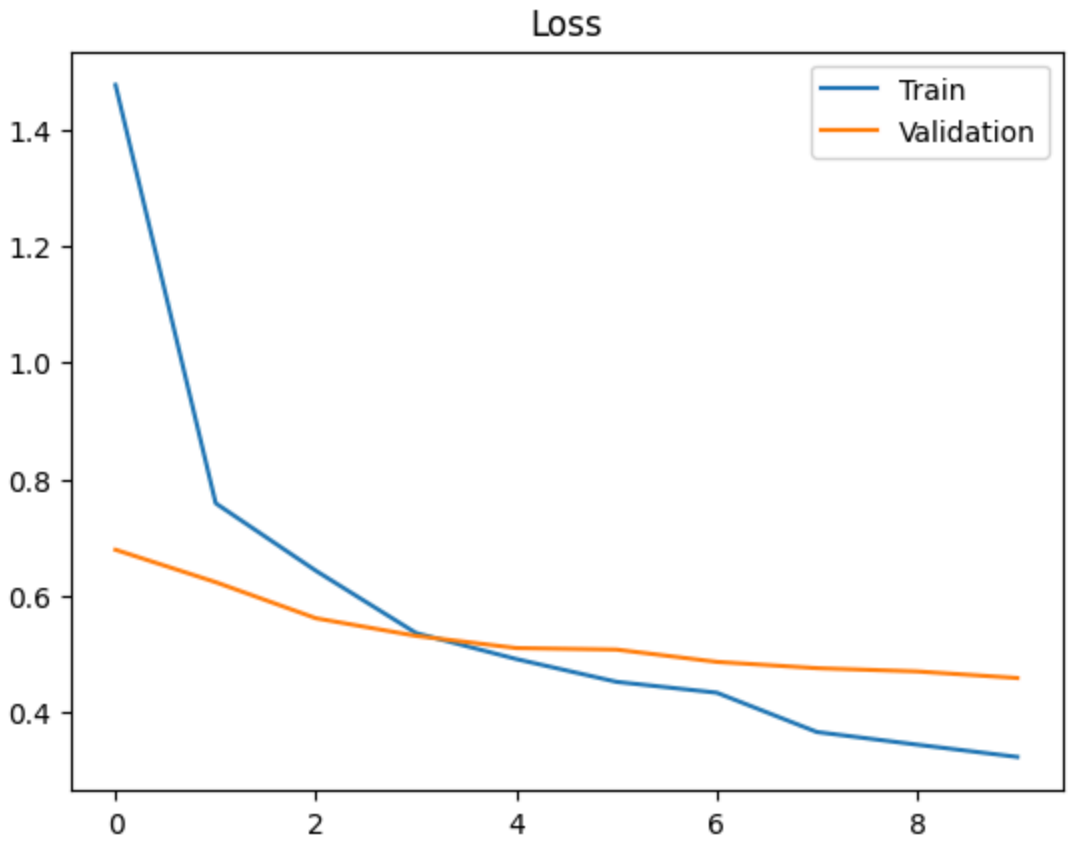
Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Ship

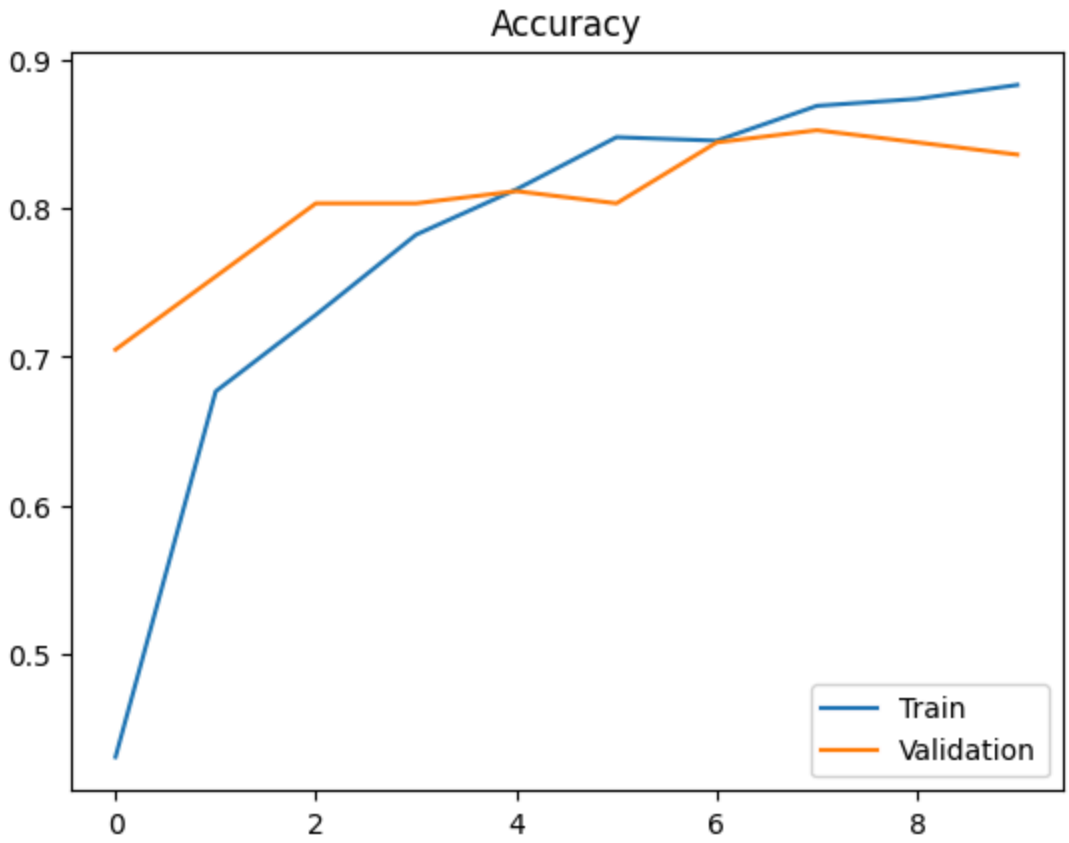


Performance Metrics

In [ ]:
```
loss_train_curve = history.history["loss"]
loss_val_curve = history.history["val_loss"]
plt.plot(loss_train_curve, label = "Train")
plt.plot(loss_val_curve, label = "Validation")
plt.legend(loc = 'upper right')
plt.title("Loss")
plt.show()
```



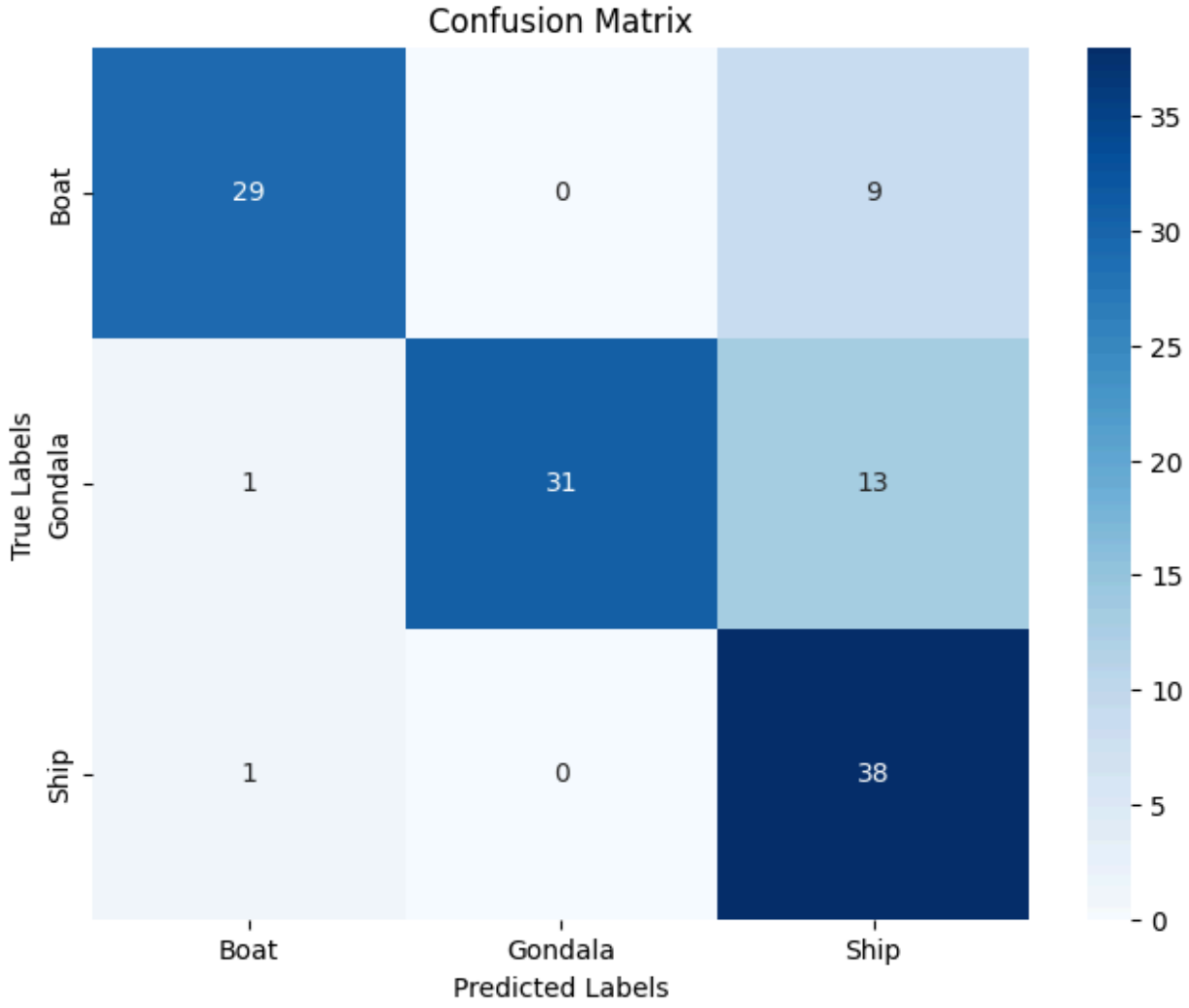
In [ ]:
```
acc_train_curve = history.history["accuracy"]
acc_val_curve = history.history["val_accuracy"]
plt.plot(acc_train_curve, label = "Train")
plt.plot(acc_val_curve, label = "Validation")
plt.legend(loc = 'lower right')
plt.title("Accuracy")
plt.show()
```



In [ ]:
```python
from sklearn.metrics import confusion_matrix, classification_report
import seaborn as sns

# Empty lists to store true labels and predicted labels
true_labels = []
predicted_labels = []

# Make predictions for images in TEST DATA
for batch in ds_test:
    # Extract images and labels from the batch
    images, labels = batch

    # Preprocess the images for prediction
    preprocessed_images = tf.map_fn(preprocess_image, images)

    # Reshape the input tensor to match the MobileNet model's input shape
    preprocessed_images = tf.reshape(preprocessed_images, (-1, 224, 224, 3))

    # Making predictions
    predictions = model.predict(preprocessed_images)


    predicted_classes = np.argmax(predictions, axis=1)

    # Append true and predicted labels to the lists
    true_labels.extend(labels.numpy())
    predicted_labels.extend(predicted_classes)

# Create a confusion matrix
conf_matrix = confusion_matrix(true_labels, predicted_labels)

# Displaying the confusion matrix using Seaborn
```

```
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=class_labels,
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix')
plt.show()
```

```
1/1 [==============================] - 0s 28ms/step
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 28ms/step
1/1 [==============================] - 0s 27ms/step
```



RESNET MODEL

In [ ]:
```
from tensorflow.keras import layers
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.applications.resnet50 import preprocess_input
```

In [ ]:
```
resnet_50 = ResNet50(include_top=False, weights='imagenet', input_shape=(224,224,3))
for layer in resnet_50.layers:
    layer.trainable = False
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/re
snet/resnet50_weights_tf_dim_ordering_tf_kernels_notop.h5
94765736/94765736 [==============================] - 1s 0us/step
```

In [ ]:
```
X = resnet_50.output
X = layers.GlobalAveragePooling2D()(X)
X = layers.Dense(512, activation='relu')(X)
X = layers.Dropout(0.5)(X)
X = layers.Dense(256, activation='relu')(X)
X = layers.Dropout(0.5)(X)
X = layers.Dense(128, activation='relu')(X)
X = layers.Dropout(0.5)(X)
X = layers.Dense(64, activation='relu')(X)
X = layers.Dropout(0.5)(X)
predictions = layers.Dense(3, activation='softmax')(X)
model = Model(inputs = resnet_50.input, outputs = predictions)
```

In [ ]:
```
model.compile(optimizer="adam", loss="sparse_categorical_crossentropy", metrics=["accu
model.summary()
```

```
Model: "model_1"
__________________________________________________________________________________________________
 Layer (type)                Output Shape                 Param #   Connected to
==================================================================================================
 input_2 (InputLayer)        [(None, 224, 224, 3)]        0         []

 conv1_pad (ZeroPadding2D)   (None, 230, 230, 3)          0         ['input_2[0][0]']

 conv1_conv (Conv2D)         (None, 112, 112, 64)         9472      ['conv1_pad[0]
[0]']

 conv1_bn (BatchNormalizati  (None, 112, 112, 64)         256       ['conv1_conv[0]
[0]']
 on)

 conv1_relu (Activation)     (None, 112, 112, 64)         0         ['conv1_bn[0]
[0]']

 pool1_pad (ZeroPadding2D)   (None, 114, 114, 64)         0         ['conv1_relu[0]
[0]']

 pool1_pool (MaxPooling2D)   (None, 56, 56, 64)           0         ['pool1_pad[0]
[0]']

 conv2_block1_1_conv (Conv2  (None, 56, 56, 64)           4160      ['pool1_pool[0]
[0]']
 D)

 conv2_block1_1_bn (BatchNo  (None, 56, 56, 64)           256       ['conv2_block1_1_
conv[0][0]']
 rmalization)

 conv2_block1_1_relu (Activ  (None, 56, 56, 64)           0         ['conv2_block1_1_
bn[0][0]']
 ation)

 conv2_block1_2_conv (Conv2  (None, 56, 56, 64)           36928     ['conv2_block1_1_
relu[0][0]']
 D)

 conv2_block1_2_bn (BatchNo  (None, 56, 56, 64)           256       ['conv2_block1_2_
conv[0][0]']
 rmalization)

 conv2_block1_2_relu (Activ  (None, 56, 56, 64)           0         ['conv2_block1_2_
bn[0][0]']
 ation)

 conv2_block1_0_conv (Conv2  (None, 56, 56, 256)          16640     ['pool1_pool[0]
[0]']
 D)

 conv2_block1_3_conv (Conv2  (None, 56, 56, 256)          16640     ['conv2_block1_2_
relu[0][0]']
 D)

 conv2_block1_0_bn (BatchNo  (None, 56, 56, 256)          1024      ['conv2_block1_0_
conv[0][0]']
```

```
 rmalization)

 conv2_block1_3_bn (BatchNo  (None, 56, 56, 256)          1024      ['conv2_block1_3_
conv[0][0]']
 rmalization)

 conv2_block1_add (Add)      (None, 56, 56, 256)          0         ['conv2_block1_0_
bn[0][0]',
                                                                     'conv2_block1_3_
bn[0][0]']

 conv2_block1_out (Activati  (None, 56, 56, 256)          0         ['conv2_block1_ad
d[0][0]']
 on)

 conv2_block2_1_conv (Conv2  (None, 56, 56, 64)           16448     ['conv2_block1_ou
t[0][0]']
 D)

 conv2_block2_1_bn (BatchNo  (None, 56, 56, 64)           256       ['conv2_block2_1_
conv[0][0]']
 rmalization)

 conv2_block2_1_relu (Activ  (None, 56, 56, 64)           0         ['conv2_block2_1_
bn[0][0]']
 ation)

 conv2_block2_2_conv (Conv2  (None, 56, 56, 64)           36928     ['conv2_block2_1_
relu[0][0]']
 D)

 conv2_block2_2_bn (BatchNo  (None, 56, 56, 64)           256       ['conv2_block2_2_
conv[0][0]']
 rmalization)

 conv2_block2_2_relu (Activ  (None, 56, 56, 64)           0         ['conv2_block2_2_
bn[0][0]']
 ation)

 conv2_block2_3_conv (Conv2  (None, 56, 56, 256)          16640     ['conv2_block2_2_
relu[0][0]']
 D)

 conv2_block2_3_bn (BatchNo  (None, 56, 56, 256)          1024      ['conv2_block2_3_
conv[0][0]']
 rmalization)

 conv2_block2_add (Add)      (None, 56, 56, 256)          0         ['conv2_block1_ou
t[0][0]',
                                                                     'conv2_block2_3_
bn[0][0]']

 conv2_block2_out (Activati  (None, 56, 56, 256)          0         ['conv2_block2_ad
d[0][0]']
 on)

 conv2_block3_1_conv (Conv2  (None, 56, 56, 64)           16448     ['conv2_block2_ou
t[0][0]']
 D)
```

```
 conv2_block3_1_bn (BatchNo  (None, 56, 56, 64)           256       ['conv2_block3_1_
conv[0][0]']
 rmalization)

 conv2_block3_1_relu (Activ  (None, 56, 56, 64)           0         ['conv2_block3_1_
bn[0][0]']
 ation)

 conv2_block3_2_conv (Conv2  (None, 56, 56, 64)           36928     ['conv2_block3_1_
relu[0][0]']
 D)

 conv2_block3_2_bn (BatchNo  (None, 56, 56, 64)           256       ['conv2_block3_2_
conv[0][0]']
 rmalization)

 conv2_block3_2_relu (Activ  (None, 56, 56, 64)           0         ['conv2_block3_2_
bn[0][0]']
 ation)

 conv2_block3_3_conv (Conv2  (None, 56, 56, 256)          16640     ['conv2_block3_2_
relu[0][0]']
 D)

 conv2_block3_3_bn (BatchNo  (None, 56, 56, 256)          1024      ['conv2_block3_3_
conv[0][0]']
 rmalization)

 conv2_block3_add (Add)      (None, 56, 56, 256)          0         ['conv2_block2_ou
t[0][0]',
                                                                     'conv2_block3_3_
bn[0][0]']

 conv2_block3_out (Activati  (None, 56, 56, 256)          0         ['conv2_block3_ad
d[0][0]']
 on)

 conv3_block1_1_conv (Conv2  (None, 28, 28, 128)          32896     ['conv2_block3_ou
t[0][0]']
 D)

 conv3_block1_1_bn (BatchNo  (None, 28, 28, 128)          512       ['conv3_block1_1_
conv[0][0]']
 rmalization)

 conv3_block1_1_relu (Activ  (None, 28, 28, 128)          0         ['conv3_block1_1_
bn[0][0]']
 ation)

 conv3_block1_2_conv (Conv2  (None, 28, 28, 128)          147584    ['conv3_block1_1_
relu[0][0]']
 D)

 conv3_block1_2_bn (BatchNo  (None, 28, 28, 128)          512       ['conv3_block1_2_
conv[0][0]']
 rmalization)

 conv3_block1_2_relu (Activ  (None, 28, 28, 128)          0         ['conv3_block1_2_
bn[0][0]']
 ation)
```

```
 conv3_block1_0_conv (Conv2  (None, 28, 28, 512)          131584    ['conv2_block3_ou
t[0][0]']
 D)

 conv3_block1_3_conv (Conv2  (None, 28, 28, 512)          66048     ['conv3_block1_2_
relu[0][0]']
 D)

 conv3_block1_0_bn (BatchNo  (None, 28, 28, 512)          2048      ['conv3_block1_0_
conv[0][0]']
 rmalization)

 conv3_block1_3_bn (BatchNo  (None, 28, 28, 512)          2048      ['conv3_block1_3_
conv[0][0]']
 rmalization)

 conv3_block1_add (Add)      (None, 28, 28, 512)          0         ['conv3_block1_0_
bn[0][0]',
                                                                     'conv3_block1_3_
bn[0][0]']

 conv3_block1_out (Activati  (None, 28, 28, 512)          0         ['conv3_block1_ad
d[0][0]']
 on)

 conv3_block2_1_conv (Conv2  (None, 28, 28, 128)          65664     ['conv3_block1_ou
t[0][0]']
 D)

 conv3_block2_1_bn (BatchNo  (None, 28, 28, 128)          512       ['conv3_block2_1_
conv[0][0]']
 rmalization)

 conv3_block2_1_relu (Activ  (None, 28, 28, 128)          0         ['conv3_block2_1_
bn[0][0]']
 ation)

 conv3_block2_2_conv (Conv2  (None, 28, 28, 128)          147584    ['conv3_block2_1_
relu[0][0]']
 D)

 conv3_block2_2_bn (BatchNo  (None, 28, 28, 128)          512       ['conv3_block2_2_
conv[0][0]']
 rmalization)

 conv3_block2_2_relu (Activ  (None, 28, 28, 128)          0         ['conv3_block2_2_
bn[0][0]']
 ation)

 conv3_block2_3_conv (Conv2  (None, 28, 28, 512)          66048     ['conv3_block2_2_
relu[0][0]']
 D)

 conv3_block2_3_bn (BatchNo  (None, 28, 28, 512)          2048      ['conv3_block2_3_
conv[0][0]']
 rmalization)

 conv3_block2_add (Add)      (None, 28, 28, 512)          0         ['conv3_block1_ou
t[0][0]',
```

```
                                                                  'conv3_block2_3_
bn[0][0]']

 conv3_block2_out (Activati  (None, 28, 28, 512)          0         ['conv3_block2_ad
d[0][0]']
 on)

 conv3_block3_1_conv (Conv2  (None, 28, 28, 128)          65664     ['conv3_block2_ou
t[0][0]']
 D)

 conv3_block3_1_bn (BatchNo  (None, 28, 28, 128)          512       ['conv3_block3_1_
conv[0][0]']
 rmalization)

 conv3_block3_1_relu (Activ  (None, 28, 28, 128)          0         ['conv3_block3_1_
bn[0][0]']
 ation)

 conv3_block3_2_conv (Conv2  (None, 28, 28, 128)          147584    ['conv3_block3_1_
relu[0][0]']
 D)

 conv3_block3_2_bn (BatchNo  (None, 28, 28, 128)          512       ['conv3_block3_2_
conv[0][0]']
 rmalization)

 conv3_block3_2_relu (Activ  (None, 28, 28, 128)          0         ['conv3_block3_2_
bn[0][0]']
 ation)

 conv3_block3_3_conv (Conv2  (None, 28, 28, 512)          66048     ['conv3_block3_2_
relu[0][0]']
 D)

 conv3_block3_3_bn (BatchNo  (None, 28, 28, 512)          2048      ['conv3_block3_3_
conv[0][0]']
 rmalization)

 conv3_block3_add (Add)      (None, 28, 28, 512)          0         ['conv3_block2_ou
t[0][0]',
                                                                   'conv3_block3_3_
bn[0][0]']

 conv3_block3_out (Activati  (None, 28, 28, 512)          0         ['conv3_block3_ad
d[0][0]']
 on)

 conv3_block4_1_conv (Conv2  (None, 28, 28, 128)          65664     ['conv3_block3_ou
t[0][0]']
 D)

 conv3_block4_1_bn (BatchNo  (None, 28, 28, 128)          512       ['conv3_block4_1_
conv[0][0]']
 rmalization)

 conv3_block4_1_relu (Activ  (None, 28, 28, 128)          0         ['conv3_block4_1_
bn[0][0]']
 ation)
```

```
 conv3_block4_2_conv (Conv2  (None, 28, 28, 128)          147584    ['conv3_block4_1_
relu[0][0]']
 D)

 conv3_block4_2_bn (BatchNo  (None, 28, 28, 128)          512       ['conv3_block4_2_
conv[0][0]']
 rmalization)

 conv3_block4_2_relu (Activ  (None, 28, 28, 128)          0         ['conv3_block4_2_
bn[0][0]']
 ation)

 conv3_block4_3_conv (Conv2  (None, 28, 28, 512)          66048     ['conv3_block4_2_
relu[0][0]']
 D)

 conv3_block4_3_bn (BatchNo  (None, 28, 28, 512)          2048      ['conv3_block4_3_
conv[0][0]']
 rmalization)

 conv3_block4_add (Add)      (None, 28, 28, 512)          0         ['conv3_block3_ou
t[0][0]',
                                                                     'conv3_block4_3_
bn[0][0]']

 conv3_block4_out (Activati  (None, 28, 28, 512)          0         ['conv3_block4_ad
d[0][0]']
 on)

 conv4_block1_1_conv (Conv2  (None, 14, 14, 256)          131328    ['conv3_block4_ou
t[0][0]']
 D)

 conv4_block1_1_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block1_1_
conv[0][0]']
 rmalization)

 conv4_block1_1_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block1_1_
bn[0][0]']
 ation)

 conv4_block1_2_conv (Conv2  (None, 14, 14, 256)          590080    ['conv4_block1_1_
relu[0][0]']
 D)

 conv4_block1_2_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block1_2_
conv[0][0]']
 rmalization)

 conv4_block1_2_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block1_2_
bn[0][0]']
 ation)

 conv4_block1_0_conv (Conv2  (None, 14, 14, 1024)         525312    ['conv3_block4_ou
t[0][0]']
 D)

 conv4_block1_3_conv (Conv2  (None, 14, 14, 1024)         263168    ['conv4_block1_2_
relu[0][0]']
 D)
```

```
 conv4_block1_0_bn (BatchNo  (None, 14, 14, 1024)         4096      ['conv4_block1_0_
conv[0][0]']
 rmalization)

 conv4_block1_3_bn (BatchNo  (None, 14, 14, 1024)         4096      ['conv4_block1_3_
conv[0][0]']
 rmalization)

 conv4_block1_add (Add)      (None, 14, 14, 1024)         0         ['conv4_block1_0_
bn[0][0]',
                                                                     'conv4_block1_3_
bn[0][0]']

 conv4_block1_out (Activati  (None, 14, 14, 1024)         0         ['conv4_block1_ad
d[0][0]']
 on)

 conv4_block2_1_conv (Conv2  (None, 14, 14, 256)          262400    ['conv4_block1_ou
t[0][0]']
 D)

 conv4_block2_1_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block2_1_
conv[0][0]']
 rmalization)

 conv4_block2_1_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block2_1_
bn[0][0]']
 ation)

 conv4_block2_2_conv (Conv2  (None, 14, 14, 256)          590080    ['conv4_block2_1_
relu[0][0]']
 D)

 conv4_block2_2_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block2_2_
conv[0][0]']
 rmalization)

 conv4_block2_2_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block2_2_
bn[0][0]']
 ation)

 conv4_block2_3_conv (Conv2  (None, 14, 14, 1024)         263168    ['conv4_block2_2_
relu[0][0]']
 D)

 conv4_block2_3_bn (BatchNo  (None, 14, 14, 1024)         4096      ['conv4_block2_3_
conv[0][0]']
 rmalization)

 conv4_block2_add (Add)      (None, 14, 14, 1024)         0         ['conv4_block1_ou
t[0][0]',
                                                                     'conv4_block2_3_
bn[0][0]']

 conv4_block2_out (Activati  (None, 14, 14, 1024)         0         ['conv4_block2_ad
d[0][0]']
 on)

 conv4_block3_1_conv (Conv2  (None, 14, 14, 256)          262400    ['conv4_block2_ou
```

```
t[0][0]']
 D)

 conv4_block3_1_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block3_1_
conv[0][0]']
 rmalization)

 conv4_block3_1_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block3_1_
bn[0][0]']
 ation)

 conv4_block3_2_conv (Conv2  (None, 14, 14, 256)          590080    ['conv4_block3_1_
relu[0][0]']
 D)

 conv4_block3_2_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block3_2_
conv[0][0]']
 rmalization)

 conv4_block3_2_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block3_2_
bn[0][0]']
 ation)

 conv4_block3_3_conv (Conv2  (None, 14, 14, 1024)         263168    ['conv4_block3_2_
relu[0][0]']
 D)

 conv4_block3_3_bn (BatchNo  (None, 14, 14, 1024)         4096      ['conv4_block3_3_
conv[0][0]']
 rmalization)

 conv4_block3_add (Add)      (None, 14, 14, 1024)         0         ['conv4_block2_ou
t[0][0]',
                                                                     'conv4_block3_3_
bn[0][0]']

 conv4_block3_out (Activati  (None, 14, 14, 1024)         0         ['conv4_block3_ad
d[0][0]']
 on)

 conv4_block4_1_conv (Conv2  (None, 14, 14, 256)          262400    ['conv4_block3_ou
t[0][0]']
 D)

 conv4_block4_1_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block4_1_
conv[0][0]']
 rmalization)

 conv4_block4_1_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block4_1_
bn[0][0]']
 ation)

 conv4_block4_2_conv (Conv2  (None, 14, 14, 256)          590080    ['conv4_block4_1_
relu[0][0]']
 D)

 conv4_block4_2_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block4_2_
conv[0][0]']
 rmalization)
```

```
 conv4_block4_2_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block4_2_
bn[0][0]']
 ation)

 conv4_block4_3_conv (Conv2  (None, 14, 14, 1024)         263168    ['conv4_block4_2_
relu[0][0]']
 D)

 conv4_block4_3_bn (BatchNo  (None, 14, 14, 1024)         4096      ['conv4_block4_3_
conv[0][0]']
 rmalization)

 conv4_block4_add (Add)      (None, 14, 14, 1024)         0         ['conv4_block3_ou
t[0][0]',
                                                                     'conv4_block4_3_
bn[0][0]']

 conv4_block4_out (Activati  (None, 14, 14, 1024)         0         ['conv4_block4_ad
d[0][0]']
 on)

 conv4_block5_1_conv (Conv2  (None, 14, 14, 256)          262400    ['conv4_block4_ou
t[0][0]']
 D)

 conv4_block5_1_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block5_1_
conv[0][0]']
 rmalization)

 conv4_block5_1_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block5_1_
bn[0][0]']
 ation)

 conv4_block5_2_conv (Conv2  (None, 14, 14, 256)          590080    ['conv4_block5_1_
relu[0][0]']
 D)

 conv4_block5_2_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block5_2_
conv[0][0]']
 rmalization)

 conv4_block5_2_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block5_2_
bn[0][0]']
 ation)

 conv4_block5_3_conv (Conv2  (None, 14, 14, 1024)         263168    ['conv4_block5_2_
relu[0][0]']
 D)

 conv4_block5_3_bn (BatchNo  (None, 14, 14, 1024)         4096      ['conv4_block5_3_
conv[0][0]']
 rmalization)

 conv4_block5_add (Add)      (None, 14, 14, 1024)         0         ['conv4_block4_ou
t[0][0]',
                                                                     'conv4_block5_3_
bn[0][0]']

 conv4_block5_out (Activati  (None, 14, 14, 1024)         0         ['conv4_block5_ad
d[0][0]']
```

```
 on)

 conv4_block6_1_conv (Conv2  (None, 14, 14, 256)          262400    ['conv4_block5_ou
t[0][0]']
 D)

 conv4_block6_1_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block6_1_
conv[0][0]']
 rmalization)

 conv4_block6_1_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block6_1_
bn[0][0]']
 ation)

 conv4_block6_2_conv (Conv2  (None, 14, 14, 256)          590080    ['conv4_block6_1_
relu[0][0]']
 D)

 conv4_block6_2_bn (BatchNo  (None, 14, 14, 256)          1024      ['conv4_block6_2_
conv[0][0]']
 rmalization)

 conv4_block6_2_relu (Activ  (None, 14, 14, 256)          0         ['conv4_block6_2_
bn[0][0]']
 ation)

 conv4_block6_3_conv (Conv2  (None, 14, 14, 1024)         263168    ['conv4_block6_2_
relu[0][0]']
 D)

 conv4_block6_3_bn (BatchNo  (None, 14, 14, 1024)         4096      ['conv4_block6_3_
conv[0][0]']
 rmalization)

 conv4_block6_add (Add)      (None, 14, 14, 1024)         0         ['conv4_block5_ou
t[0][0]',
                                                                     'conv4_block6_3_
bn[0][0]']

 conv4_block6_out (Activati  (None, 14, 14, 1024)         0         ['conv4_block6_ad
d[0][0]']
 on)

 conv5_block1_1_conv (Conv2  (None, 7, 7, 512)            524800    ['conv4_block6_ou
t[0][0]']
 D)

 conv5_block1_1_bn (BatchNo  (None, 7, 7, 512)            2048      ['conv5_block1_1_
conv[0][0]']
 rmalization)

 conv5_block1_1_relu (Activ  (None, 7, 7, 512)            0         ['conv5_block1_1_
bn[0][0]']
 ation)

 conv5_block1_2_conv (Conv2  (None, 7, 7, 512)            2359808   ['conv5_block1_1_
relu[0][0]']
 D)

 conv5_block1_2_bn (BatchNo  (None, 7, 7, 512)            2048      ['conv5_block1_2_
```

```
conv[0][0]']
 rmalization)

 conv5_block1_2_relu (Activ  (None, 7, 7, 512)           0         ['conv5_block1_2_
bn[0][0]']
 ation)

 conv5_block1_0_conv (Conv2  (None, 7, 7, 2048)          2099200   ['conv4_block6_ou
t[0][0]']
 D)

 conv5_block1_3_conv (Conv2  (None, 7, 7, 2048)          1050624   ['conv5_block1_2_
relu[0][0]']
 D)

 conv5_block1_0_bn (BatchNo  (None, 7, 7, 2048)          8192      ['conv5_block1_0_
conv[0][0]']
 rmalization)

 conv5_block1_3_bn (BatchNo  (None, 7, 7, 2048)          8192      ['conv5_block1_3_
conv[0][0]']
 rmalization)

 conv5_block1_add (Add)      (None, 7, 7, 2048)          0         ['conv5_block1_0_
bn[0][0]',
                                                                     'conv5_block1_3_
bn[0][0]']

 conv5_block1_out (Activati  (None, 7, 7, 2048)          0         ['conv5_block1_ad
d[0][0]']
 on)

 conv5_block2_1_conv (Conv2  (None, 7, 7, 512)           1049088   ['conv5_block1_ou
t[0][0]']
 D)

 conv5_block2_1_bn (BatchNo  (None, 7, 7, 512)           2048      ['conv5_block2_1_
conv[0][0]']
 rmalization)

 conv5_block2_1_relu (Activ  (None, 7, 7, 512)           0         ['conv5_block2_1_
bn[0][0]']
 ation)

 conv5_block2_2_conv (Conv2  (None, 7, 7, 512)           2359808   ['conv5_block2_1_
relu[0][0]']
 D)

 conv5_block2_2_bn (BatchNo  (None, 7, 7, 512)           2048      ['conv5_block2_2_
conv[0][0]']
 rmalization)

 conv5_block2_2_relu (Activ  (None, 7, 7, 512)           0         ['conv5_block2_2_
bn[0][0]']
 ation)

 conv5_block2_3_conv (Conv2  (None, 7, 7, 2048)          1050624   ['conv5_block2_2_
relu[0][0]']
 D)
```

```
 conv5_block2_3_bn (BatchNo  (None, 7, 7, 2048)           8192      ['conv5_block2_3_
conv[0][0]']
 rmalization)

 conv5_block2_add (Add)      (None, 7, 7, 2048)           0         ['conv5_block1_ou
t[0][0]',
                                                                     'conv5_block2_3_
bn[0][0]']

 conv5_block2_out (Activati  (None, 7, 7, 2048)           0         ['conv5_block2_ad
d[0][0]']
 on)

 conv5_block3_1_conv (Conv2  (None, 7, 7, 512)            1049088   ['conv5_block2_ou
t[0][0]']
 D)

 conv5_block3_1_bn (BatchNo  (None, 7, 7, 512)            2048      ['conv5_block3_1_
conv[0][0]']
 rmalization)

 conv5_block3_1_relu (Activ  (None, 7, 7, 512)            0         ['conv5_block3_1_
bn[0][0]']
 ation)

 conv5_block3_2_conv (Conv2  (None, 7, 7, 512)            2359808   ['conv5_block3_1_
relu[0][0]']
 D)

 conv5_block3_2_bn (BatchNo  (None, 7, 7, 512)            2048      ['conv5_block3_2_
conv[0][0]']
 rmalization)

 conv5_block3_2_relu (Activ  (None, 7, 7, 512)            0         ['conv5_block3_2_
bn[0][0]']
 ation)

 conv5_block3_3_conv (Conv2  (None, 7, 7, 2048)           1050624   ['conv5_block3_2_
relu[0][0]']
 D)

 conv5_block3_3_bn (BatchNo  (None, 7, 7, 2048)           8192      ['conv5_block3_3_
conv[0][0]']
 rmalization)

 conv5_block3_add (Add)      (None, 7, 7, 2048)           0         ['conv5_block2_ou
t[0][0]',
                                                                     'conv5_block3_3_
bn[0][0]']

 conv5_block3_out (Activati  (None, 7, 7, 2048)           0         ['conv5_block3_ad
d[0][0]']
 on)

 global_average_pooling2d_1  (None, 2048)                 0         ['conv5_block3_ou
t[0][0]']
  (GlobalAveragePooling2D)

 dense_2 (Dense)             (None, 512)                  1049088   ['global_average_
pooling2d_1[0
```

```
                                                                    ][0]']

 dropout_1 (Dropout)         (None, 512)               0         ['dense_2[0][0]']

 dense_3 (Dense)             (None, 256)               131328    ['dropout_1[0]
[0]']

 dropout_2 (Dropout)         (None, 256)               0         ['dense_3[0][0]']

 dense_4 (Dense)             (None, 128)               32896     ['dropout_2[0]
[0]']

 dropout_3 (Dropout)         (None, 128)               0         ['dense_4[0][0]']

 dense_5 (Dense)             (None, 64)                8256      ['dropout_3[0]
[0]']

 dropout_4 (Dropout)         (None, 64)                0         ['dense_5[0][0]']

 dense_6 (Dense)             (None, 3)                 195       ['dropout_4[0]
[0]']

==================================================================================
=============
Total params: 24809475 (94.64 MB)
Trainable params: 1221763 (4.66 MB)
Non-trainable params: 23587712 (89.98 MB)
__________________________________________________________________________________
_____________
```

In [ ]:
```
batch_size = 128
epochs = 15
Resnet_Model = model.fit(train_generator, validation_data=valid_generator, epochs=epoc
```

```
Epoch 1/15
14/14 [==============================] - 9s 650ms/step - loss: 0.4570 - accuracy: 0.8
407 - val_loss: 0.3061 - val_accuracy: 0.9098
Epoch 2/15
14/14 [==============================] - 8s 565ms/step - loss: 0.4829 - accuracy: 0.8
056 - val_loss: 0.2512 - val_accuracy: 0.9180
Epoch 3/15
14/14 [==============================] - 7s 513ms/step - loss: 0.4267 - accuracy: 0.8
548 - val_loss: 0.3693 - val_accuracy: 0.9016
Epoch 4/15
14/14 [==============================] - 8s 589ms/step - loss: 0.4173 - accuracy: 0.8
290 - val_loss: 0.2621 - val_accuracy: 0.9098
Epoch 5/15
14/14 [==============================] - 7s 501ms/step - loss: 0.3194 - accuracy: 0.8
759 - val_loss: 0.3481 - val_accuracy: 0.9098
Epoch 6/15
14/14 [==============================] - 8s 576ms/step - loss: 0.3437 - accuracy: 0.8
735 - val_loss: 0.2640 - val_accuracy: 0.9344
Epoch 7/15
14/14 [==============================] - 7s 491ms/step - loss: 0.2256 - accuracy: 0.9
133 - val_loss: 0.2727 - val_accuracy: 0.9016
Epoch 8/15
14/14 [==============================] - 7s 489ms/step - loss: 0.2465 - accuracy: 0.9
251 - val_loss: 0.3428 - val_accuracy: 0.9098
Epoch 9/15
14/14 [==============================] - 8s 564ms/step - loss: 0.1997 - accuracy: 0.9
297 - val_loss: 0.2522 - val_accuracy: 0.9262
Epoch 10/15
14/14 [==============================] - 7s 511ms/step - loss: 0.1745 - accuracy: 0.9
485 - val_loss: 0.3747 - val_accuracy: 0.9098
Epoch 11/15
14/14 [==============================] - 8s 588ms/step - loss: 0.1627 - accuracy: 0.9
321 - val_loss: 0.3608 - val_accuracy: 0.9262
Epoch 12/15
14/14 [==============================] - 7s 489ms/step - loss: 0.1299 - accuracy: 0.9
602 - val_loss: 0.3561 - val_accuracy: 0.9180
Epoch 13/15
14/14 [==============================] - 8s 594ms/step - loss: 0.1641 - accuracy: 0.9
555 - val_loss: 0.3297 - val_accuracy: 0.9180
Epoch 14/15
14/14 [==============================] - 7s 520ms/step - loss: 0.1212 - accuracy: 0.9
578 - val_loss: 0.3680 - val_accuracy: 0.9180
Epoch 15/15
14/14 [==============================] - 7s 519ms/step - loss: 0.0923 - accuracy: 0.9
766 - val_loss: 0.4453 - val_accuracy: 0.9262
```

In [ ]:
```
test_loss, test_acc = model.evaluate(test_generator)
print("The test loss is: ", test_loss)
print("The best accuracy is: ", test_acc*100)
```

```
2/2 [==============================] - 3s 2s/step - loss: 0.3804 - accuracy: 0.9344
The test loss is:  0.38037899136543274
The best accuracy is:  93.44262480735779
```

In [ ]:
```
def preprocess_image(image):
    image = tf.image.resize(image, (224, 224))
    img_array = tf.keras.preprocessing.image.img_to_array(image)
    img_array = tf.expand_dims(img_array, 0)
    img_array = tf.keras.applications.resnet.preprocess_input(img_array)
    return img_array
```

```
# Make predictions for images in Whole Dataset
for batch in dataset.take(1):  # Take the first 1 batche
    # Extract images and labels from the batch
    images, labels = batch

    # Preprocess the images for prediction
    preprocessed_images = tf.map_fn(preprocess_image, images)

    # Reshape the input tensor to match the ResNet model's input shape
    preprocessed_images = tf.reshape(preprocessed_images, (-1, 224, 224, 3))

    # Making predictions
    predictions = model.predict(preprocessed_images)


    class_labels = Labels  #Category's names
    predicted_classes = np.argmax(predictions, axis=1)

    # Displaying each test image along with the original and predicted class
    for i in range(images.shape[0]):
        plt.imshow(tf.keras.preprocessing.image.array_to_img(images[i]))
        plt.title(f"Original Class: {class_labels[labels[i]]}\nPredicted Class: {class
        plt.axis("off")
        plt.show()
```

```
1/1 [==============================] - 0s 34ms/step
```

Original Class: Ship
Predicted Class: Ship

Original Class: Gondala
Predicted Class: Gondala

Original Class: Boat
Predicted Class: Boat

Original Class: Ship
Predicted Class: Ship