SHIP CLASSIFICATION USING IMAGE DATASET

TEAM DETAILS:

G.JYOTHI-160617733135

V.ANUSHA-160617733172

K.MOUNIKA-160617733148

INTRODUCTION

- The increased presence of autonomous systems requires reliable classification algorithms to understand their surrounding environment. These autonomous systems have the potential to find widespread use in sea and ocean waters, necessitating a reliable classification of their surrounding. Since ships are the most popular means of transportation and warfare in seas and oceans, they need to be classified by autonomous systems. This can be used to detect unusual entry of ships which can be harmful.

PROPOSED SYSTEM

A System is proposed to automate the detection of presence of ships in the given image using Machine Learning and Deep Learning Algorithms.We are proposing ship detection ,a ship classification based on the type and category of ships.

EXPERIMENTAL INVESTIGATION

- Data Collection

Data preparation is the most important step in training macine learning models.It all starts with data that is clean,sufficient and representative of the problem statement.A raw dataset was collected with type of ships. We were collected 10 types of ship Images are downloaded from google and saved in different folders. Our dataset consisting 2000 images(aircraft carrier,bulkers,cruise,drilling rigs,fire fighting vessels, fishing vessels,inland dry cargo vessels,motor yachts,restaurent ships,submarines). So we divide dataset into two parts:train set and test set.The total number of images used in train set was 1400 and test set has 600 images.

Data Collection

•Data Preprocessing:

Data preprocessing is a data mining technique which is used to transform the raw data in a useful and efficient format.

->Import The ImageDataGenerator Library

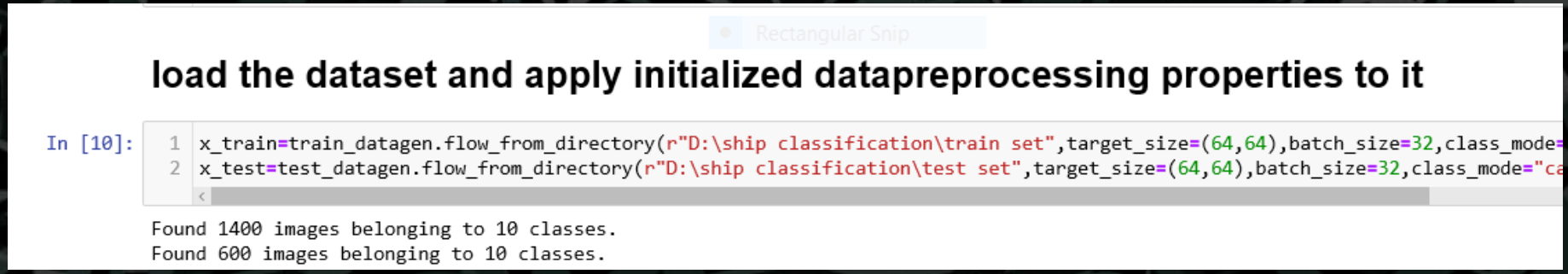
In our project we created a dataset of various types of ships and for preprocessing we first imported liabraries as below

In [ ]:
```
# importing  the necessary libraries
```

In [4]:
```
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten

```

->*Load The Trainset And Testset*



load the dataset and apply initialized datapreprocessing properties to it

```
In [10]:
1 x_train=train_datagen.flow_from_directory(r"D:\ship classification\train set",target_size=(64,64),batch_size=32,class_mode=
2 x_test=test_datagen.flow_from_directory(r"D:\ship classification\test set",target_size=(64,64),batch_size=32,class_mode="ca
```

```
Found 1400 images belonging to 10 classes.
Found 600 images belonging to 10 classes.
```

Model Building:

- Here we have to initialize the model and add convolution layer.pooling layer,flattening layer,hidden layer and output layer. After that compile the model and save the model.

initializing the model

In [12]:
```
model=Sequential()
```

add convolution layer

In [13]:
```
model.add(Convolution2D(32,(3,3),input_shape=(64,64,3),activation="relu"))
```

```
WARNING:tensorflow:From C:\Users\kkolipaka\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:517: The name tf.placeholder is deprecated. Please use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From C:\Users\kkolipaka\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:4138: The name tf.random_uniform is deprecated. Please use tf.random.uniform instead.
```

add pooling layer

In [14]:
```
model.add(MaxPooling2D(pool_size=(2,2)))
```

```
WARNING:tensorflow:From C:\Users\kkolipaka\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:3976: The name tf.nn.max_pool is deprecated. Please use tf.nn.max_pool2d instead.
```

add flattening layer

In [15]:
```
model.add(Flatten())
```

add hidden layer

In [12]:
```
model.add(Dense(units=128,init="uniform",activation="relu"))
```

```
C:\Users\kkolipaka\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Dense` call to the Keras 2 API: `Dense(units=128, activation="relu", kernel_initializer="uniform")`
  """Entry point for launching an IPython kernel.
```

fit the model with datagenerator class

In [18]:
```
model.fit_generator(x_train,steps_per_epoch=44,epochs=10,validation_data=x_test,validation_steps=18.75)
```

```
WARNING:tensorflow:From C:\Users\kkolipaka\Anaconda3\lib\site-packages\tensorflow\python\ops\math_grad.py:1250: add_dispatch_su
pport.<locals>.wrapper (from tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
WARNING:tensorflow:From C:\Users\kkolipaka\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:986: The name tf.ass
ign_add is deprecated. Please use tf.compat.v1.assign_add instead.

Epoch 1/10
44/44 [==============================] - 17s 383ms/step - loss: 2.4698 - acc: 0.1517 - val_loss: 2.1401 - val_acc: 0.2067
Epoch 2/10
44/44 [==============================] - 11s 243ms/step - loss: 2.0948 - acc: 0.2439 - val_loss: 2.0074 - val_acc: 0.2833
Epoch 3/10
44/44 [==============================] - 16s 362ms/step - loss: 1.9752 - acc: 0.3040 - val_loss: 1.9716 - val_acc: 0.2733
Epoch 4/10
44/44 [==============================] - 12s 270ms/step - loss: 1.8909 - acc: 0.3599 - val_loss: 1.9371 - val_acc: 0.3300
Epoch 5/10
44/44 [==============================] - 17s 381ms/step - loss: 1.8008 - acc: 0.3866 - val_loss: 1.9254 - val_acc: 0.3383
Epoch 6/10
44/44 [==============================] - 12s 266ms/step - loss: 1.7241 - acc: 0.4229 - val_loss: 1.9243 - val_acc: 0.3467
Epoch 7/10
44/44 [==============================] - 11s 260ms/step - loss: 1.6776 - acc: 0.4122 - val_loss: 1.9514 - val_acc: 0.3233
Epoch 8/10
44/44 [==============================] - 14s 311ms/step - loss: 1.6586 - acc: 0.4515 - val_loss: 1.9893 - val_acc: 0.3483
Epoch 9/10
44/44 [==============================] - 14s 311ms/step - loss: 1.6017 - acc: 0.4567 - val_loss: 1.8458 - val_acc: 0.3567
Epoch 10/10
44/44 [==============================] - 13s 296ms/step - loss: 1.5135 - acc: 0.5040 - val_loss: 1.8924 - val_acc: 0.3750
```

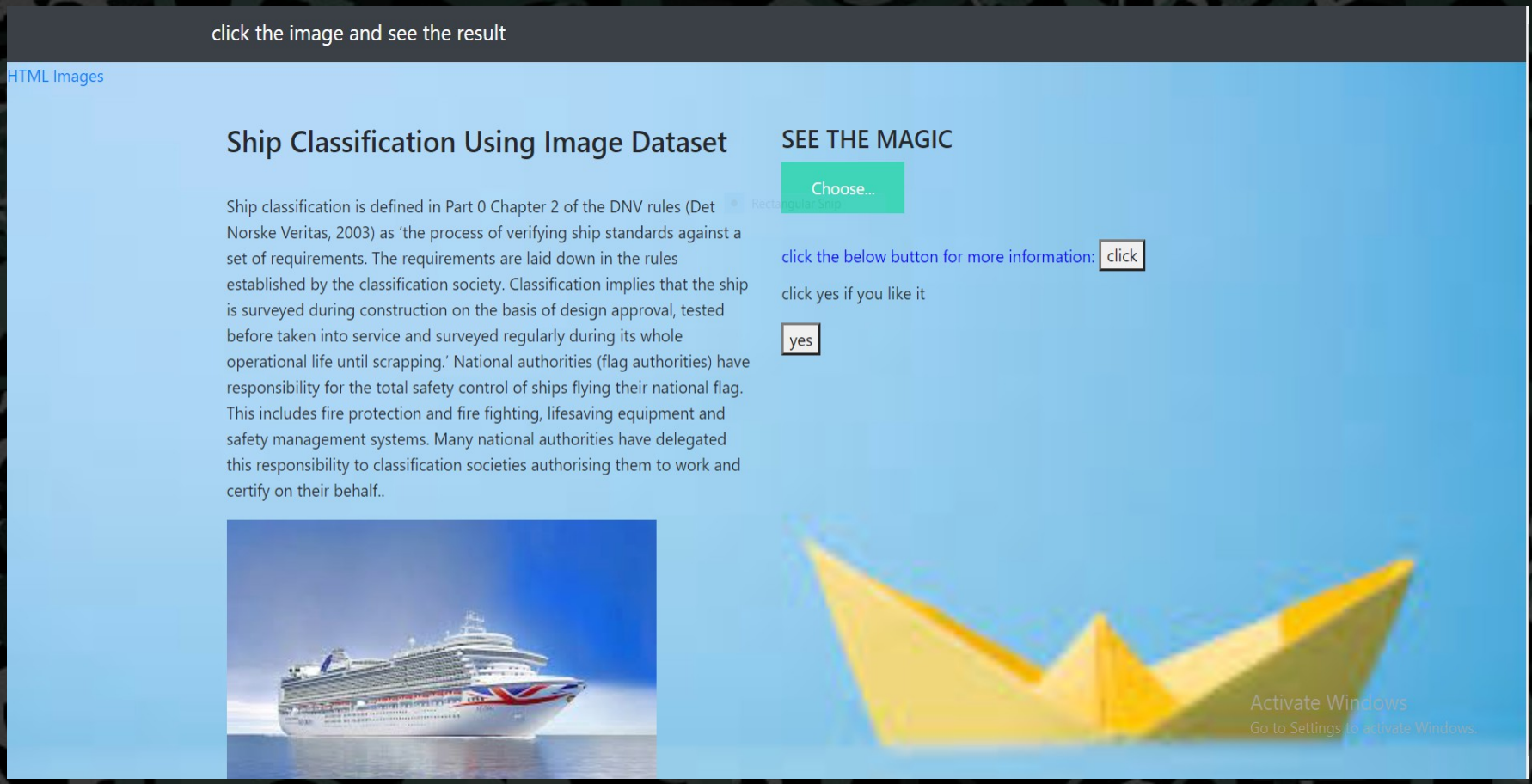
Out[18]:
```
<keras.callbacks.History at 0x1dc5a07c4c8>
```

save the model

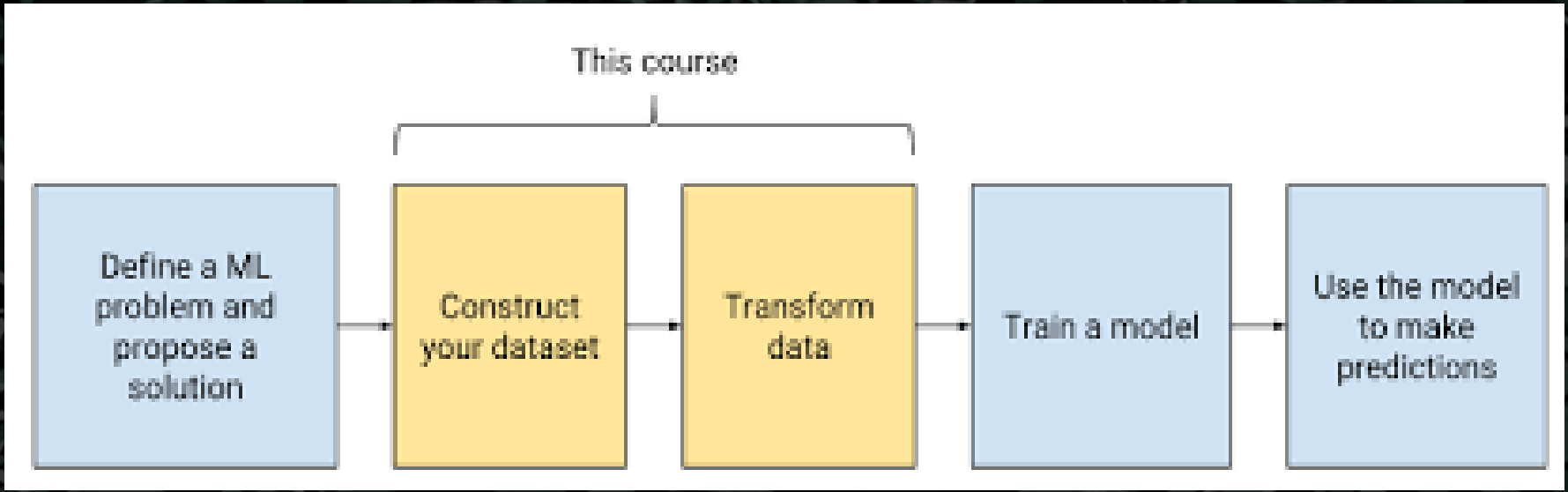
In [19]:
```
model.save("ship.h5")
```

Application Building

- Here we are using flask app for application building.In flask we load the ship model and segregate the model .Finally we got output like below



Flow Chart:



Result:

- We have to type the code in spyder for flask app and html code and execute that in anaconda prompt to get this as a output.It detects type of ship and gives the name of the ship.
-

Output:

click the image and see the result

HTML Images

Ship Classification Using Image Dataset

Ship classification is defined in Part 0 Chapter 2 of the DNV rules (Det Norske Veritas, 2003) as 'the process of verifying ship standards against a set of requirements. The requirements are laid down in the rules established by the classification society. Classification implies that the ship is surveyed during construction on the basis of design approval, tested before taken into service and surveyed regularly during its whole operational life until scrapping.' National authorities (flag authorities) have responsibility for the total safety control of ships flying their national flag. This includes fire protection and fire fighting, lifesaving equipment and safety management systems. Many national authorities have delegated this responsibility to classification societies authorising them to work and certify on their behalf..

SEE THE MAGIC

Choose...

click the button for more information: click

Result: the predicted ship is : drilling rigs

click yes if you like it

yes

Conclusion

- As our project is used to identify and classify the type of the ship.Human health issue will be reduced in the ship classification method.All types of ship can be identified and classified in our ship classification method and therefore it can be used as a dataset for ship classification.


thank you
Fungsitacan.com