SYMBIOSIS UNIVERSITY OF APPLIED SCIENCES

INDORE

PROJECT REPORT

ON

"IMAGE CLASSIFICATION USING KERAS"

Submitted to "Symbiosis University of Applied Sciences, Indore
As an Internship report for the partial fulfillment of the award of degree of

BACHELOR OF TECHNOLOGY

IN

SCHOOL OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

Submitted To:
Mr. Deepak Yadav
Professor

Submitted By:
Aayank Singhai (2018BTCS001)
Akshat Porwal (2018BTCS007)
Ashutosh Tripathi (2018BTCS014)

TABLE OF CONTENTS

Title: Image Classification using Keras

Objective:

In this project, we will build a Convolution Neural Network (CNN) using Keras which is a deep learning Application Programming Interface (API) which runs on top of TensorFlow platform. This is a classification problem which is a type of supervised learning algorithm. The classification problem is to categorize all the pixels of a digital image into one of the defined classes.

The problem uses CIFAR-10 dataset is already available in the datasets module of Keras. This dataset consists of 60000 32x32 color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.

Scope:

Image classification is the most critical use case in digital image analysis. Image classification is an application of both supervised classification and unsupervised classification. It is used in automated image organization, visual search for improved product discoverability, image and face recognition on social networks, etc.

Technologies Used:

- Keras
- Python 3.8.6
- CIFAR-10 dataset
- TensorFlow 2.3.1
- Tkinter
- Pillow
- NumPy

- Matplotlib
- Jupyter Notebook

Proposed Methodology:

The project uses a sequential model for classifying images into 10 different classes which are:

a) Airplane
b) Car
c) Bird
d) Cat
e) Deer
f) Dog
g) Frog
h) Horse
i) Ship
j) Truck

We train our neural network on the target class samples given in the CIFAR-10 dataset and then classify new samples. The image pixels are fed to the neural network as float values. The model is then trained with the images for 10 epochs and the accuracy is calculated.

The Graphical User Interface (GUI) of the project is created using Tkinter which is a GUI toolkit.

Results:

The sequential model was created for the CIFAR-10 dataset with an accuracy of 67.54%.

IMAGE CLASSIFICATION USING KERAS

Import the required layers and modules to create our convolution neural net architecture and load the dataset from keras datasets module

In [11]:
```python
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.constraints import maxnorm
from keras.optimizers import SGD
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from keras.utils import np_utils
from keras.datasets import cifar10
import matplotlib.pyplot as plt
```

Convert the pixel values of the dataset to float type and then normalize the dataset

In [12]:
```python
(train_X,train_Y),(test_X,test_Y)=cifar10.load_data()
train_x=train_X.astype('float32')
test_X=test_X.astype('float32')

train_X=train_X/255.0
test_X=test_X/255.0
```

Train the model

In [18]:
```python
model.fit(train_X,train_Y,
    validation_data=(test_X,test_Y),
    epochs=10,batch_size=32)
```

```
Epoch 1/10
1563/1563 [==============================] - 208s 133ms/step - loss: 1.7689 - accuracy: 0.3596 - val_loss: 1.4998 - val_accu
racy: 0.4647
Epoch 2/10
1563/1563 [==============================] - 201s 129ms/step - loss: 1.3599 - accuracy: 0.5119 - val_loss: 1.2608 - val_accu
racy: 0.5513
Epoch 3/10
1563/1563 [==============================] - 202s 129ms/step - loss: 1.1954 - accuracy: 0.5729 - val_loss: 1.1169 - val_accu
racy: 0.6015
Epoch 4/10
1563/1563 [==============================] - 203s 130ms/step - loss: 1.0816 - accuracy: 0.6170 - val_loss: 1.0534 - val_accu
racy: 0.6314
Epoch 5/10
1563/1563 [==============================] - 200s 128ms/step - loss: 0.9903 - accuracy: 0.6490 - val_loss: 1.0259 - val_accu
racy: 0.6399
Epoch 6/10
1563/1563 [==============================] - 200s 128ms/step - loss: 0.9080 - accuracy: 0.6795 - val_loss: 0.9910 - val_accu
racy: 0.6512
Epoch 7/10
1563/1563 [==============================] - 202s 129ms/step - loss: 0.8396 - accuracy: 0.7024 - val_loss: 0.9745 - val_accu
racy: 0.6632
Epoch 8/10
1563/1563 [==============================] - 202s 129ms/step - loss: 0.7783 - accuracy: 0.7251 - val_loss: 0.9592 - val_accu
racy: 0.6662
Epoch 9/10
1563/1563 [==============================] - 203s 130ms/step - loss: 0.7226 - accuracy: 0.7434 - val_loss: 0.9489 - val_accu
racy: 0.6734
Epoch 10/10
1563/1563 [==============================] - 209s 134ms/step - loss: 0.6654 - accuracy: 0.7656 - val_loss: 0.9487 - val_accu
racy: 0.6754
```

Out[18]:
```
<tensorflow.python.keras.callbacks.History at 0x217b122c910>
```

View the model summary for better understanding of model architecture

In [17]:
```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_2 (Conv2D)            (None, 32, 32, 32)        896
_________________________________________________________________
dropout_2 (Dropout)          (None, 32, 32, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 32, 32, 32)        9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 16, 16, 32)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 8192)              0
_________________________________________________________________
dense_2 (Dense)              (None, 512)               4194816
_________________________________________________________________
dropout_3 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_3 (Dense)              (None, 10)                5130
=================================================================
Total params: 4,210,090
Trainable params: 4,210,090
Non-trainable params: 0
_________________________________________________________________
```

Perform the one-hot encoding for target classes

In [13]:
```
train_Y=np_utils.to_categorical(train_Y)
test_Y=np_utils.to_categorical(test_Y)

num_classes=test_Y.shape[1]
```

Create the sequential model and add the layers

In [14]:
```
model=Sequential()
model.add(Conv2D(32,(3,3),input_shape=(32,32,3),
    padding='same',activation='relu',
    kernel_constraint=maxnorm(3)))
model.add(Dropout(0.2))
model.add(Conv2D(32,(3,3),activation='relu',padding='same',kernel_constraint=maxnorm(3)))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(512,activation='relu',kernel_constraint=maxnorm(3)))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
```

Configure the optimizer and compile the model

In [16]:
```
sgd=SGD(lr=0.01,momentum=0.9,decay=(0.01/25),nesterov=False)

model.compile(loss='categorical_crossentropy',
  optimizer=sgd,
  metrics=['accuracy'])
```

Calculate its accuracy on testing data

In [19]:
```
_,acc=model.evaluate(test_X,test_Y)
print(acc*100)
```

```
313/313 [==============================] - 5s 17ms/step - loss: 0.9487 - accuracy: 0.6754
67.54000186920166
```

Save the model

In [20]:
```
model.save("model1_cifar_10epoch.h5")
```

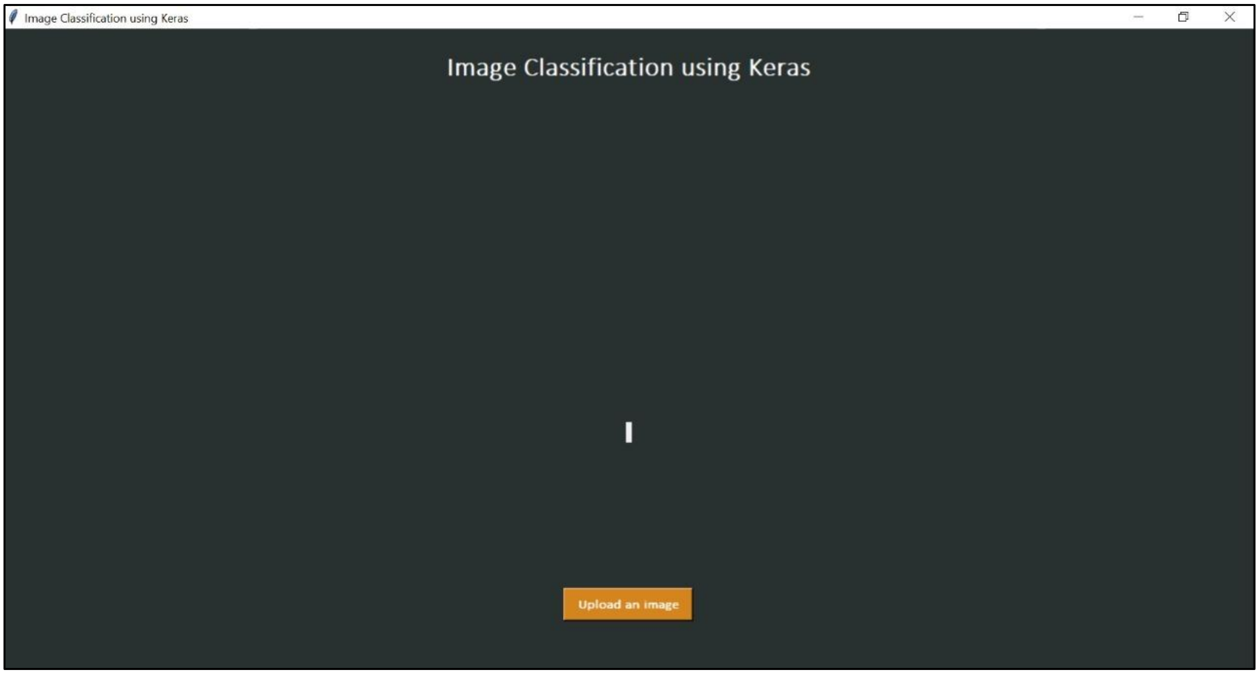


Image Classification using Keras
Image Classification using Keras
Upload an image

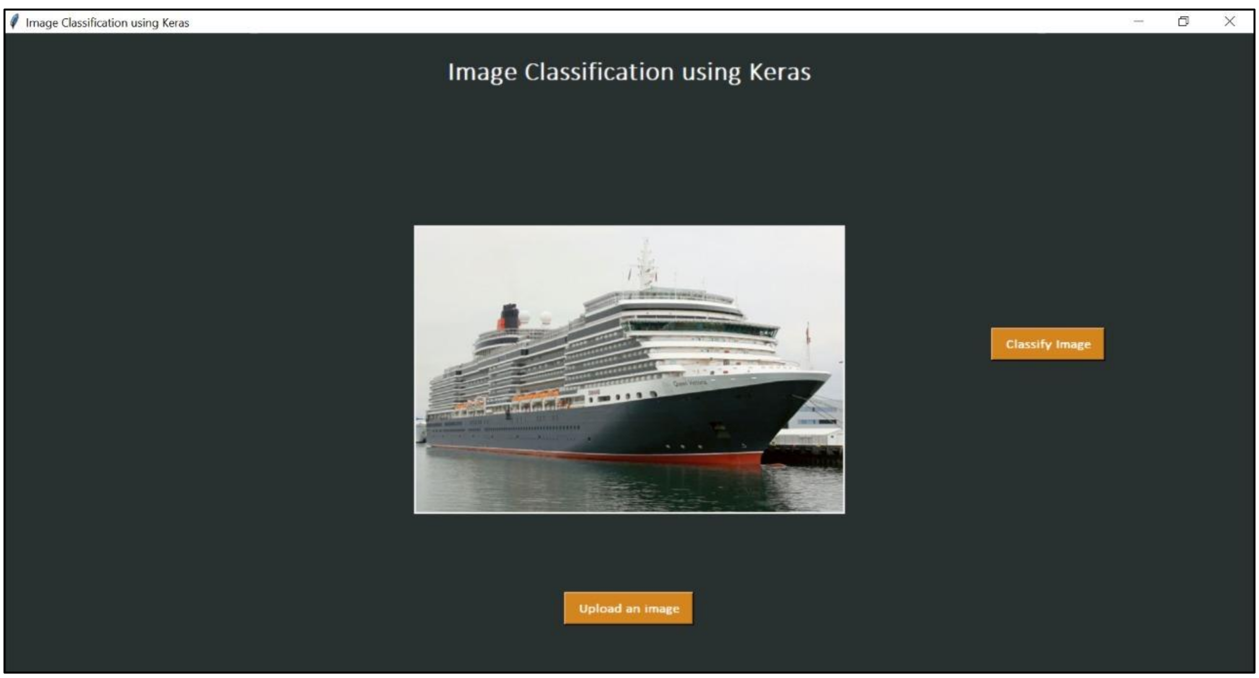


Image Classification using Keras
Image Classification using Keras
Classify Image
Upload an image

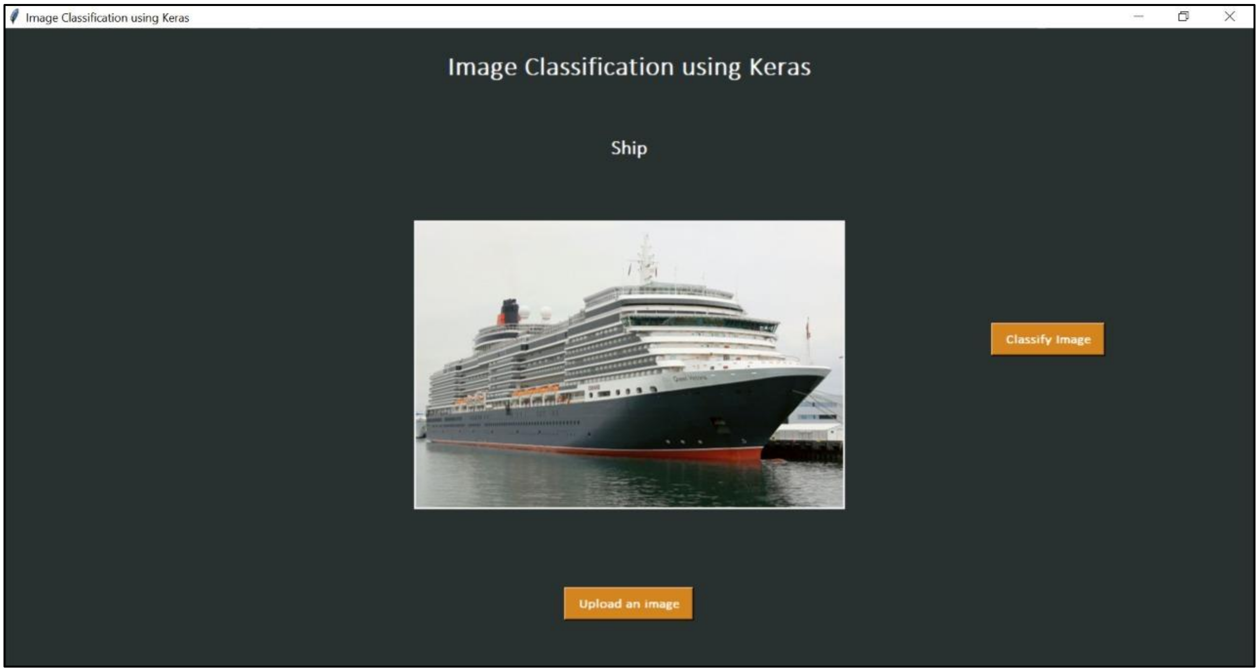


Image Classification using Keras
Image Classification using Keras
Ship
Classify Image
Upload an image

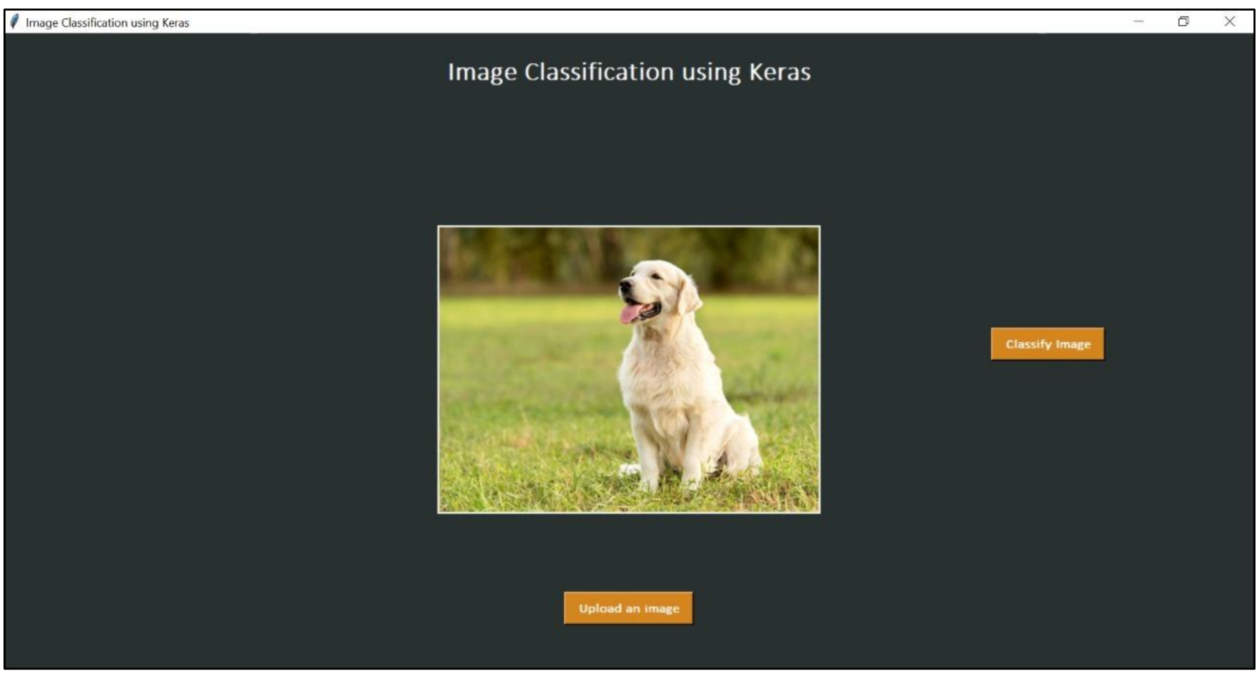


Image Classification using Keras
Image Classification using Keras
Classify Image
Upload an image

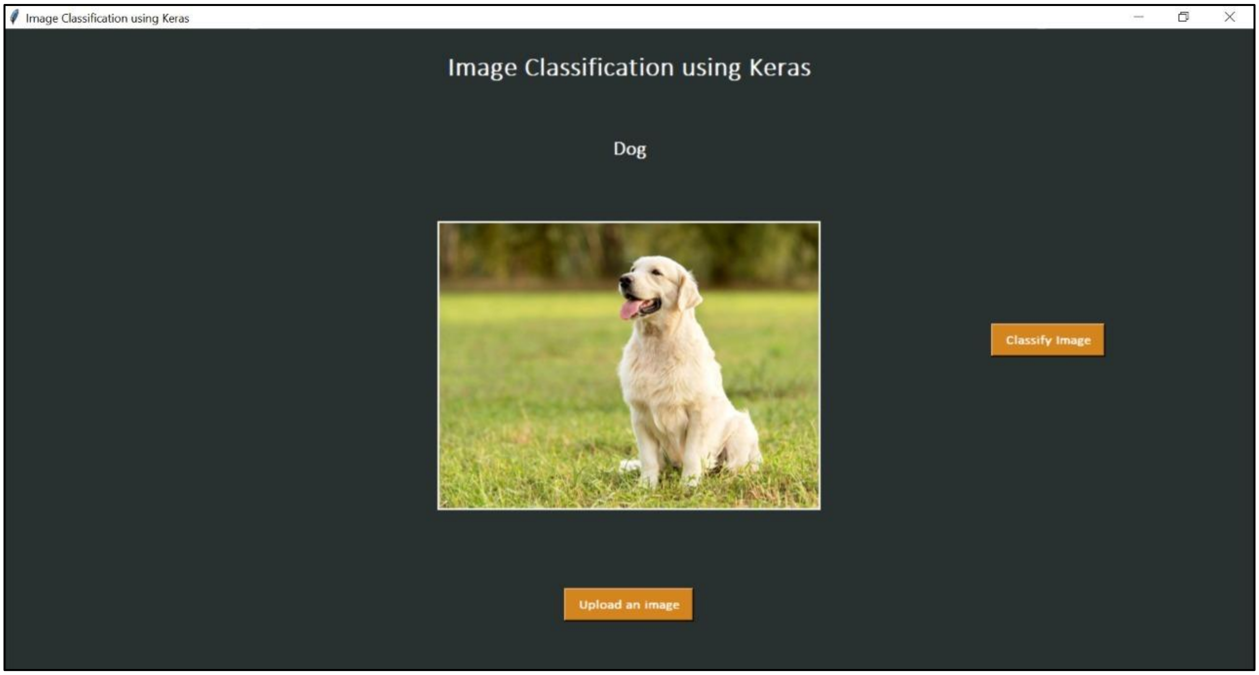


Image Classification using Keras
Image Classification using Keras
Dog
Classify Image
Upload an image

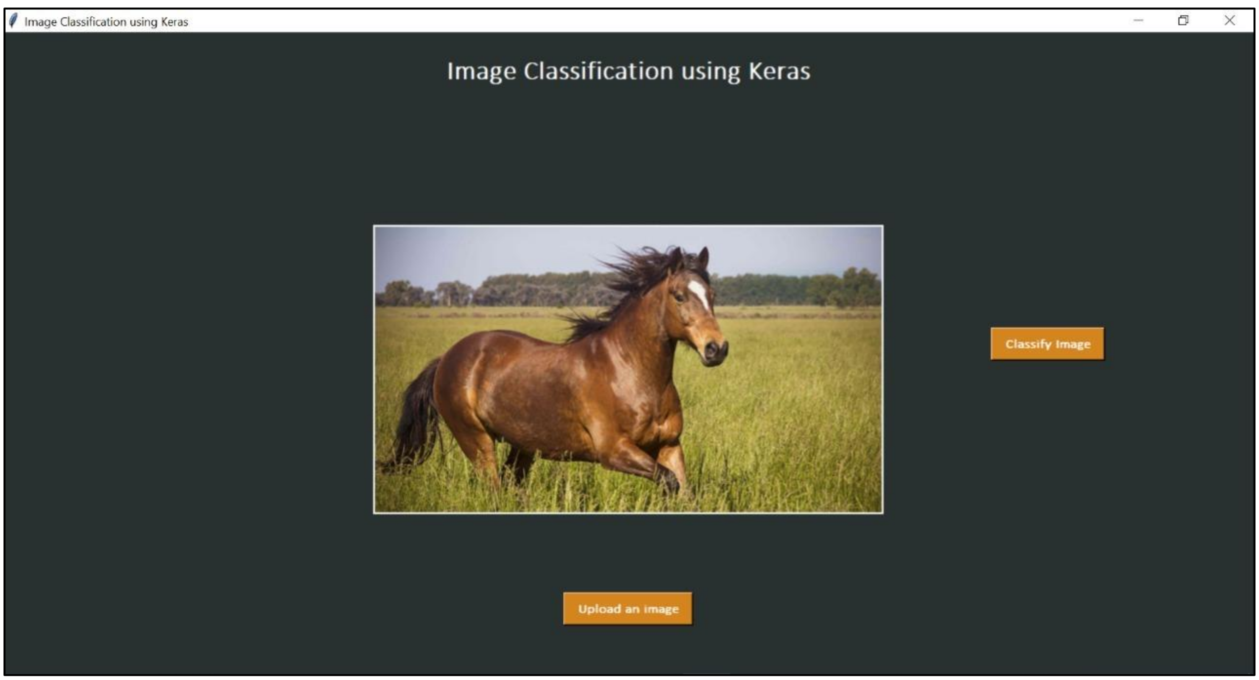


Image Classification using Keras
Image Classification using Keras
Classify Image
Upload an image

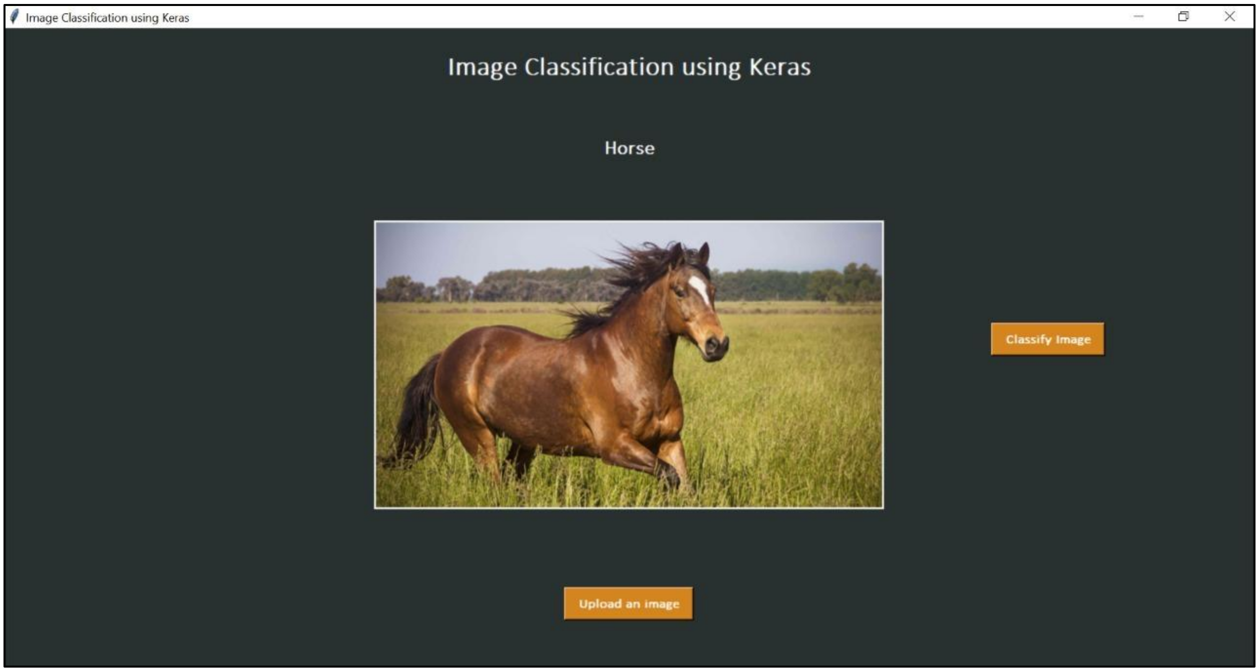


Image Classification using Keras
Image Classification using Keras
Horse
Classify Image
Upload an image

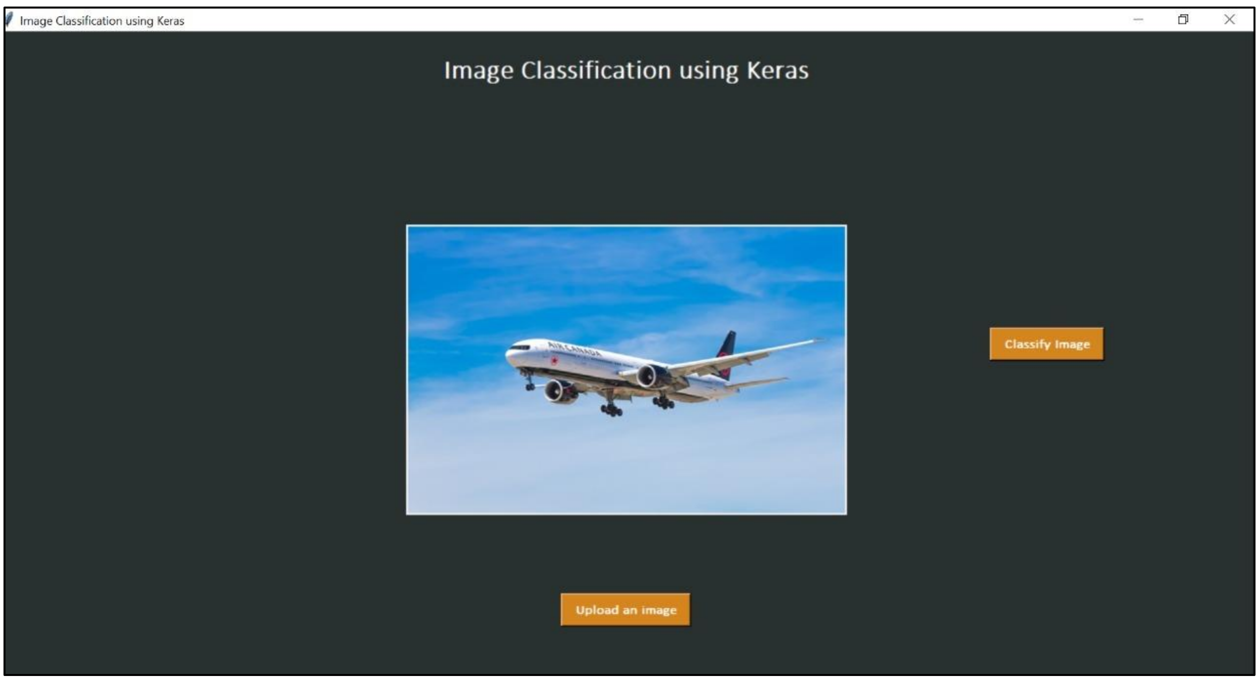
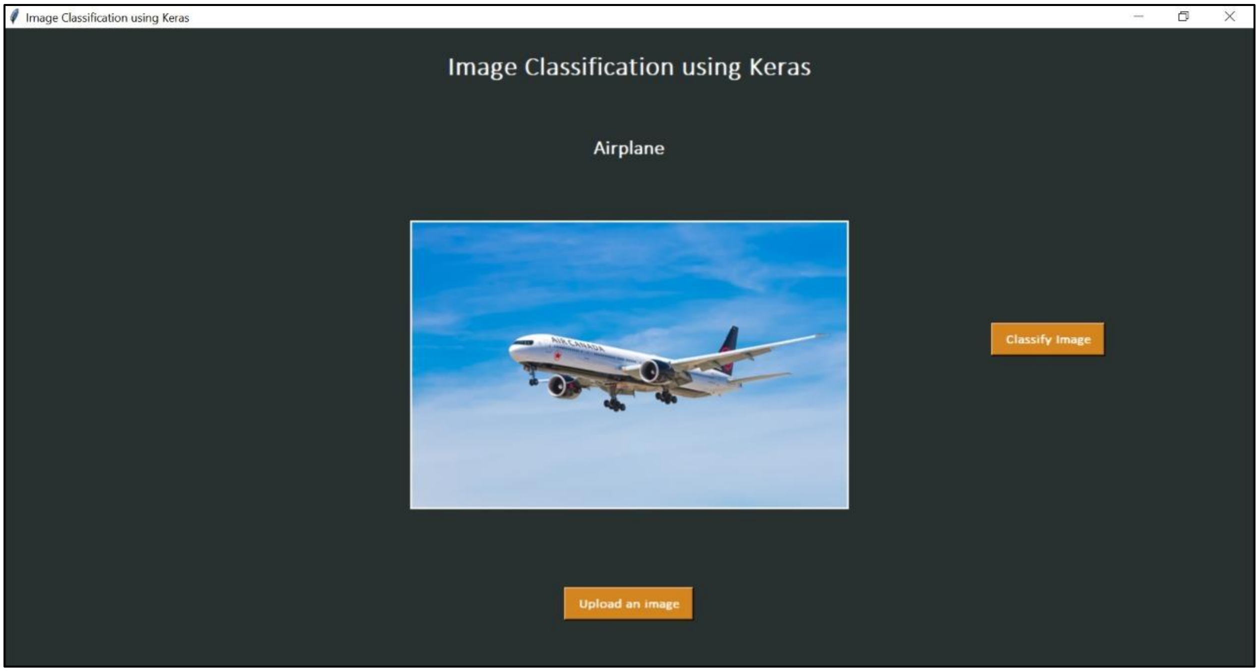


Image Classification using Keras
Image Classification using Keras
Classify Image
Upload an image



GitHub Repository: https://github.com/Ashhhutosh/Image_Classification.git

Bibliography & Reference:

- https://www.cs.toronto.edu/~kriz/cifar.html
- https://medium.com/mlait/introduction-to-keras-deep-learning-library-2844b39f0496
- https://www.geeksforgeeks.org/python-gui-tkinter/
- https://medium.com/deep-learning-demystified/tensorflow-2-model-building-with-tf-keras-1a9b050373ed
- https://medium.com/@gkadusumilli/gui-programming-using-python-tkinter-package-8e7f67fa4637