day1-036

June 28, 2024

```
[ ]: import tensorflow as tf
     from tensorflow import keras
     from tensorflow.keras import layers
     from tensorflow.keras.preprocessing.image import ImageDataGenerator

     # DEfine image size and batch size
     IMG_SIZE=325
     BATCH_SIZE=32
```

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

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

```
[ ]: train_datagen = ImageDataGenerator(rescale=1./255,validation_split=0.2) #␣
      ↪Correct the typo here

     train_generator = train_datagen.flow_from_directory(
        '/content/drive/MyDrive/Pistachio_Image_Dataset',
         target_size=(IMG_SIZE,IMG_SIZE),
         batch_size=BATCH_SIZE,
         class_mode='binary',
         subset='training'
     )
     validation_generator = train_datagen.flow_from_directory(
         '/content/drive/MyDrive/Pistachio_Image_Dataset',
         target_size=(IMG_SIZE,IMG_SIZE),
         batch_size=BATCH_SIZE,
         class_mode='binary',
         subset='validation'
     )
```

```
Found 1465 images belonging to 3 classes.
Found 366 images belonging to 3 classes.
```

[ ]:
```
# Define the model
model = keras.Sequential([
    layers.Conv2D(32,␣
 ↪(3,3),activation='relu',input_shape=(IMG_SIZE,IMG_SIZE,3)),
    layers.MaxPooling2D(2,2),
    layers.Conv2D(64,(3,3),activation='relu'),
    layers.MaxPooling2D(2,2),
    layers.Conv2D(128,(3,3),activation='relu'),
    layers.MaxPooling2D(2,2),
    layers.Flatten(),
    layers.Dense(128,activation='relu'),
    layers.Dense(1,activation='sigmoid') #output layer
])
```

[ ]:
```
model.compile(optimizer='adam', loss='binary_crossentropy',␣
 ↪metrics=['accuracy'])
```

[ ]:
```
model.fit(train_generator, validation_data=validation_generator, epochs=5) #␣
 ↪Use 'validation_generator'
```

```
Epoch 1/5
46/46 [==============================] - 382s 8s/step - loss: -323628288.0000 -
accuracy: 0.0000e+00 - val_loss: -1179421312.0000 - val_accuracy: 0.0000e+00
Epoch 2/5
46/46 [==============================] - 348s 8s/step - loss: -6498495488.0000 -
accuracy: 0.0000e+00 - val_loss: -16334635008.0000 - val_accuracy: 0.0000e+00
Epoch 3/5
46/46 [==============================] - 358s 8s/step - loss: -51591274496.0000
- accuracy: 0.0000e+00 - val_loss: -102183010304.0000 - val_accuracy: 0.0000e+00
Epoch 4/5
46/46 [==============================] - 345s 8s/step - loss: -240289267712.0000
- accuracy: 0.0000e+00 - val_loss: -409262227456.0000 - val_accuracy: 0.0000e+00
Epoch 5/5
46/46 [==============================] - 360s 8s/step - loss: -808488468480.0000
- accuracy: 0.0000e+00 - val_loss: -1234512248832.0000 - val_accuracy:
0.0000e+00
```

[ ]:
```
<keras.src.callbacks.History at 0x7f744ca8d030>
```

[5]:
```
model = keras.Sequential([
    layers.Conv2D(32, (3,3), activation='relu',␣
 ↪input_shape=(IMG_SIZE,IMG_SIZE,3)),
    layers.MaxPooling2D((2,2)),
    layers.Conv2D(64, (3,3), activation='relu'),
    layers.MaxPooling2D((2,2)),
    layers.Conv2D(128, (3,3), activation='relu'),
    layers.MaxPooling2D((2,2)),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(4, activation='softmax')
])
```

[6]:
```
model.save('pistachio_model.h5')
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103:
UserWarning: You are saving your model as an HDF5 file via `model.save()`. This
file format is considered legacy. We recommend using instead the native Keras
format, e.g. `model.save('my_model.keras')`.
  saving_api.save_model(
WARNING:tensorflow:Compiled the loaded model, but the compiled metrics have yet
to be built. `model.compile_metrics` will be empty until you train or evaluate
the model.
```

[7]:
```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
model = load_model('pistachio_model.h5')
print("Model Loaded")
```

```
WARNING:tensorflow:No training configuration found in the save file, so the
model was *not* compiled. Compile it manually.
```

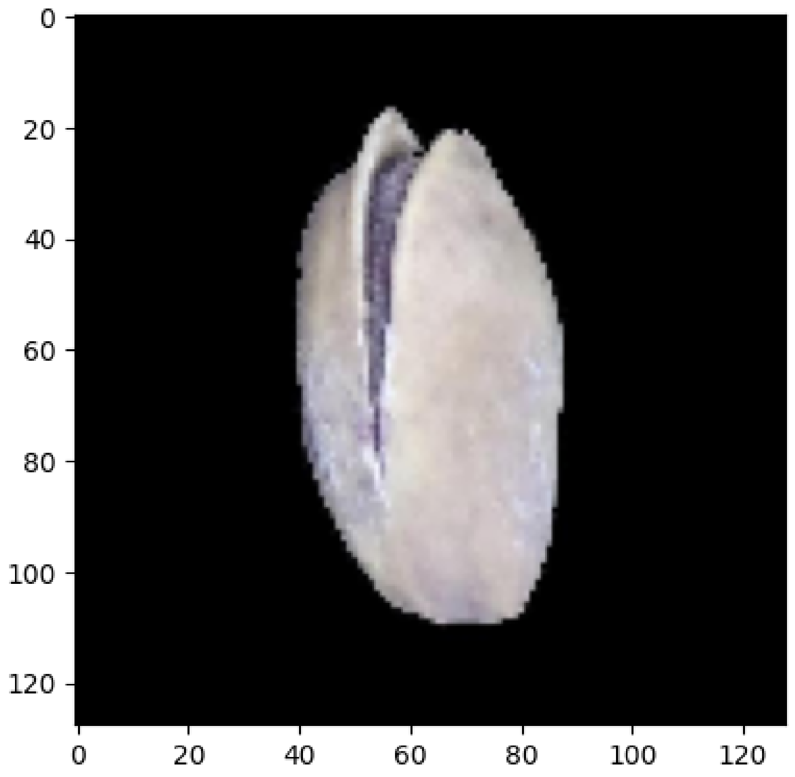
```
Model Loaded
```

[9]:
```
# Load and view the image
from matplotlib import pyplot as plt
test_image_path = r"/content/drive/MyDrive/Pistachio_Image_Dataset/
 ↪Pistachio_Image_Dataset/Kirmizi_Pistachio/kirmizi (1).jpg"
img = image.load_img(test_image_path, target_size=(128, 128)) # Resize image to␣
 ↪match model input

plt.imshow(img)
plt.axis()
plt.show()

#convert image into array
img_array = image.img_to_array(img)
img_array = np.expand_dims(img_array, axis=0)
img_array /= 255.  # Normalize the pixel values


# Make predictions
prediction = model.predict(img_array)
# Print the prediction
print(prediction)
```



```
1/1 [==============================] - 0s 298ms/step
[[0.2559754  0.25510195 0.2489587  0.2399639 ]]
```

```
[ ]:
```