multiclass-proj-1

June 28, 2024

[12]:

```python
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 = 224
BATCH_SIZE = 32
```

[13]:

```python
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator # Import␣
 ↪ImageDataGenerator

# Define image size and batch size
IMG_SIZE = 224
BATCH_SIZE = 32

# Define data generators for train, validation and test sets
train_datagen = ImageDataGenerator(rescale=1./255,validation_split=0.2)

train_generator = train_datagen.flow_from_directory(
    r"/content/drive/MyDrive/multiclass_dataset/Dataset/train",
    target_size=(IMG_SIZE, IMG_SIZE),
    batch_size=BATCH_SIZE,
    class_mode='categorical',
    subset='training'
)

val_generator = train_datagen.flow_from_directory(
    r"/content/drive/MyDrive/multiclass_dataset/Dataset/train",
    target_size=(IMG_SIZE, IMG_SIZE),
    batch_size=BATCH_SIZE,
    class_mode='categorical',
    subset='validation'
)
```

```
Found 1175 images belonging to 3 classes.
Found 293 images belonging to 3 classes.
```

[14]:

```
# Get the class indices from the training generator
class_indices = train_generator.class_indices

# Extract class names
class_names = list(class_indices.keys())

print("Class indices:", class_indices)
print("Class names:", class_names)
```

```
Class indices: {'airplanes': 0, 'cars': 1, 'ship': 2}
Class names: ['airplanes', 'cars', 'ship']
```

[16]:

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers

# Define a Sequential 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(3, activation='softmax')
])
```

[17]:

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

[18]:

```
model.fit(train_generator,validation_data=val_generator,epochs=10)
```

```
Epoch 1/10
37/37 [==============================] - 182s 5s/step - loss: 0.3542 - accuracy:
0.7898 - val_loss: 0.2285 - val_accuracy: 0.8737
Epoch 2/10
37/37 [==============================] - 150s 4s/step - loss: 0.1804 - accuracy:
0.8996 - val_loss: 0.1816 - val_accuracy: 0.8942
Epoch 3/10
37/37 [==============================] - 152s 4s/step - loss: 0.1341 - accuracy:
0.9234 - val_loss: 0.1662 - val_accuracy: 0.8942
Epoch 4/10
37/37 [==============================] - 151s 4s/step - loss: 0.1149 - accuracy:
```

```
0.9234 - val_loss: 0.1833 - val_accuracy: 0.8874
Epoch 5/10
37/37 [==============================] - 159s 4s/step - loss: 0.0669 - accuracy:
0.9651 - val_loss: 0.2855 - val_accuracy: 0.8669
Epoch 6/10
37/37 [==============================] - 151s 4s/step - loss: 0.0489 - accuracy:
0.9762 - val_loss: 0.2269 - val_accuracy: 0.8874
Epoch 7/10
37/37 [==============================] - 155s 4s/step - loss: 0.0400 - accuracy:
0.9830 - val_loss: 0.2930 - val_accuracy: 0.8737
Epoch 8/10
37/37 [==============================] - 150s 4s/step - loss: 0.0201 - accuracy:
0.9940 - val_loss: 0.3095 - val_accuracy: 0.8669
Epoch 9/10
37/37 [==============================] - 167s 5s/step - loss: 0.0055 - accuracy:
0.9983 - val_loss: 0.4311 - val_accuracy: 0.8430
Epoch 10/10
37/37 [==============================] - 158s 4s/step - loss: 0.0026 - accuracy:
1.0000 - val_loss: 0.4329 - val_accuracy: 0.8635
```

[18]:

```
<keras.src.callbacks.History at 0x7cb1f42278b0>
```

[19]:

```python
model.save('Alzheimer.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(
```

[20]:

```python
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
model = load_model('Alzheimer.h5')
print("Model Loaded")
```

```
Model Loaded
```

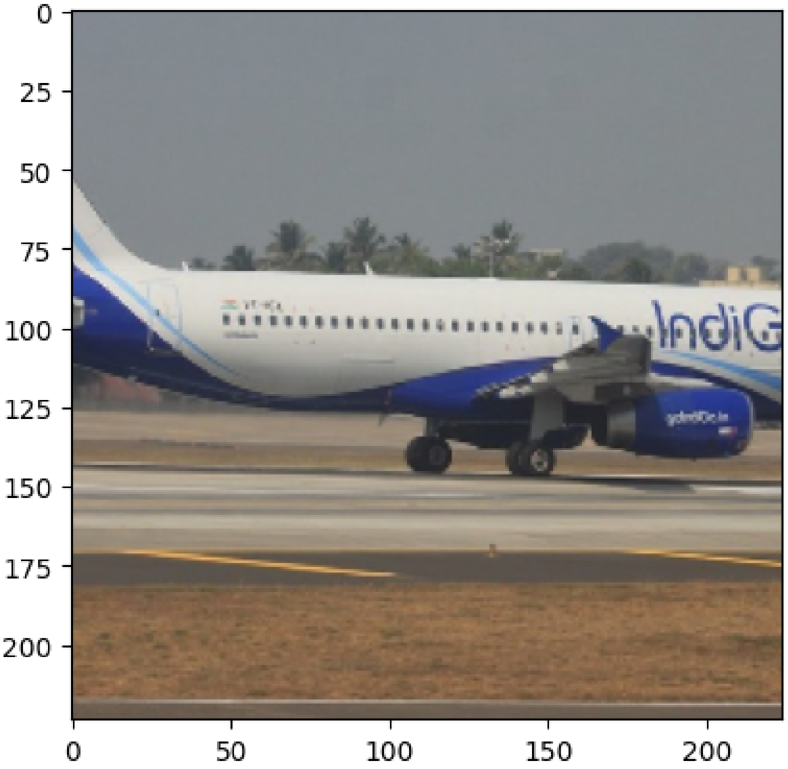
[26]:

```python
# Load and view the image
from matplotlib import pyplot as plt
test_image_path = r"/content/drive/MyDrive/multiclass_dataset/Dataset/test/
  ↪airplanes/airplane1.jpg"
img = image.load_img(test_image_path, target_size=(224, 224))

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 179ms/step
[[1.0000000e+00 1.4200763e-10 5.1258334e-14]]
```

[28]:

```
#interprete the results
prediction = model.predict(img_array)
ind = np.argmax(prediction[0])
print(class_names[ind])
```

```
1/1 [==============================] - 0s 45ms/step
airplanes
```