multiclass-1

June 28, 2024

[22]:
```
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
```

[23]:
```
train_datagen = ImageDataGenerator(rescale=1./255,validation_split=0.2)


train_generator = train_datagen.flow_from_directory(
    r"/content/drive/MyDrive/1SV21CS049/dataset_padang_food",
    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/1SV21CS049/dataset_padang_food",
    target_size=(IMG_SIZE, IMG_SIZE),
    batch_size=BATCH_SIZE,
    class_mode='categorical',
    subset='validation'
)
```

```
Found 799 images belonging to 9 classes.
Found 194 images belonging to 9 classes.
```

[24]:
```
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: {'ayam_goreng': 0, 'ayam_pop': 1, 'daging_rendang': 2,
'dendeng_batokok': 3, 'gulai_ikan': 4, 'gulai_tambusu': 5, 'gulai_tunjang': 6,
'telur_balado': 7, 'telur_dadar': 8}
Class names: ['ayam_goreng', 'ayam_pop', 'daging_rendang', 'dendeng_batokok',
'gulai_ikan', 'gulai_tambusu', 'gulai_tunjang', 'telur_balado', 'telur_dadar']
```

[25]:
```
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(9, activation='softmax') # Change the number of neurons to␣
 ↪match the number of classes in your dataset
])
```

[26]:
```
model.compile(optimizer='adam',
              loss='categorical_crossentropy', # Use categorical_crossentropy␣
 ↪for multi-class
              metrics=['accuracy'])
```

[27]:
```
model.fit(train_generator,validation_data=val_generator,epochs=10)
```

```
Epoch 1/10
25/25 [==============================] - 135s 5s/step - loss: 2.3573 - accuracy:
0.1790 - val_loss: 1.7999 - val_accuracy: 0.3351
Epoch 2/10
25/25 [==============================] - 106s 4s/step - loss: 1.5724 - accuracy:
0.4243 - val_loss: 1.3449 - val_accuracy: 0.4948
Epoch 3/10
25/25 [==============================] - 105s 4s/step - loss: 1.2186 - accuracy:
0.5469 - val_loss: 1.2349 - val_accuracy: 0.5670
Epoch 4/10
25/25 [==============================] - 108s 4s/step - loss: 0.8667 - accuracy:
0.7109 - val_loss: 1.2062 - val_accuracy: 0.5876
Epoch 5/10
25/25 [==============================] - 104s 4s/step - loss: 0.6075 - accuracy:
0.7997 - val_loss: 1.1683 - val_accuracy: 0.6392
Epoch 6/10
25/25 [==============================] - 105s 4s/step - loss: 0.3865 - accuracy:
0.8698 - val_loss: 1.3496 - val_accuracy: 0.6237
Epoch 7/10
25/25 [==============================] - 107s 4s/step - loss: 0.2348 - accuracy:
```

```
0.9262 - val_loss: 1.5012 - val_accuracy: 0.5979
Epoch 8/10
25/25 [==============================] - 105s 4s/step - loss: 0.1210 - accuracy:
0.9599 - val_loss: 1.6394 - val_accuracy: 0.6134
Epoch 9/10
25/25 [==============================] - 107s 4s/step - loss: 0.0345 - accuracy:
0.9950 - val_loss: 1.9941 - val_accuracy: 0.6392
Epoch 10/10
25/25 [==============================] - 107s 4s/step - loss: 0.0179 - accuracy:
0.9975 - val_loss: 1.9763 - val_accuracy: 0.6082
```

[27]:
```
<keras.src.callbacks.History at 0x7a7058968ee0>
```

[28]:
```
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(
```

[29]:
```
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
```

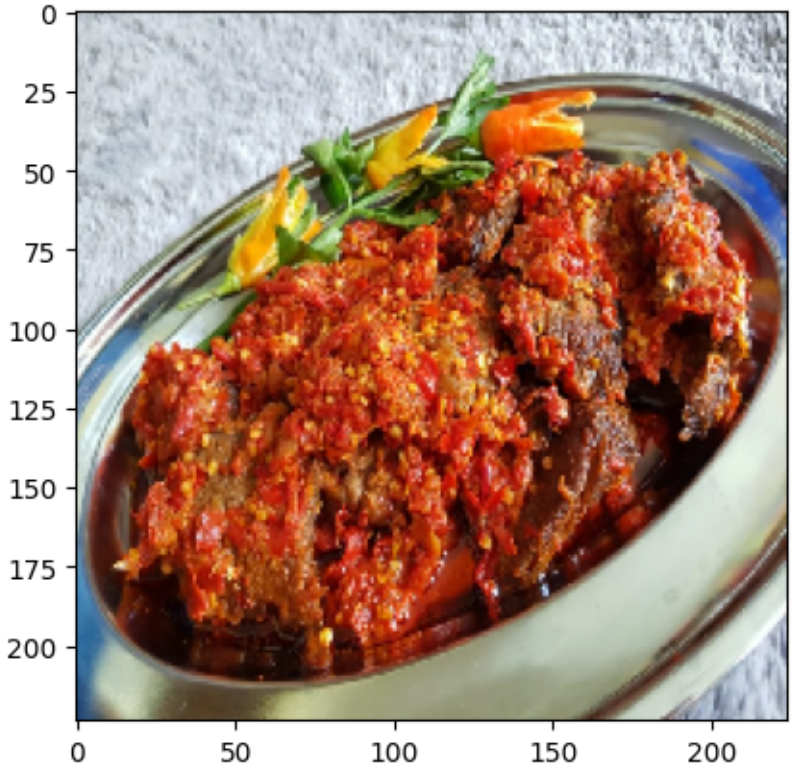
[50]:
```
from matplotlib import pyplot as plt
test_image_path = r"/content/drive/MyDrive/1SV21CS049/dataset_padang_food/
 ↪dendeng_batokok/dendeng_batokok (100).jpg" # Add the filename of an image in␣
 ↪the ayam_goreng directory
img = image.load_img(test_image_path, target_size=(224, 224))

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 52ms/step
[[2.7645697e-07 3.3368230e-20 1.0330305e-05 9.9998939e-01 2.1162450e-13
  3.3025784e-14 8.7538837e-11 2.9191138e-08 2.2171969e-09]]
```

[43]:
```
prediction = model.predict(img_array)
ind = np.argmax(prediction[0])
print(class_names[ind])
```

```
1/1 [==============================] - 0s 51ms/step
telur_dadar
```