multiclass-alzheimer

June 16, 2025

[1]:
```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```
IMG_SIZE=224
BATCH_SIZE=32
```

[3]:
```
train_datagen=ImageDataGenerator(rescale=1./255,validation_split=0.2)
```

[7]:
```
train_generator=train_datagen.flow_from_directory('/content/drive/MyDrive/
 ↪segmented_images/train',
target_size=(IMG_SIZE, IMG_SIZE),
batch_size=BATCH_SIZE,
class_mode='categorical',
subset='training')
```

```
Found 8 images belonging to 4 classes.
```

[8]:
```
val_generator=train_datagen.flow_from_directory('/content/drive/MyDrive/
 ↪segmented_images/train',
target_size=(IMG_SIZE, IMG_SIZE), # Add target_size here
batch_size=BATCH_SIZE,
class_mode='categorical',
subset='validation')
```

```
Found 2 images belonging to 4 classes.
```

[9]:
```
class_indices = train_generator.class_indices
class_names=list(class_indices.keys())
print(class_names)
print(class_indices)
```

```
['MildDemented', 'ModerateDemented', 'NonDemented', 'VeryMildDemented']
{'MildDemented': 0, 'ModerateDemented': 1, 'NonDemented': 2, 'VeryMildDemented':
3}
```

[10]:
```
from tensorflow.keras.layers import Input

model = keras.Sequential([
    Input(shape=(IMG_SIZE, IMG_SIZE, 3)),  # Explicitly define input shape
    layers.Conv2D(32, (3, 3), activation='relu'),
    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')
])
```

[11]:
```
model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 222, 222, 32)	896
max_pooling2d (MaxPooling2D)	(None, 111, 111, 32)	0
conv2d_1 (Conv2D)	(None, 109, 109, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 54, 54, 64)	0
conv2d_2 (Conv2D)	(None, 52, 52, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 26, 26, 128)	0
flatten (Flatten)	(None, 86528)	0
dense (Dense)	(None, 128)	11,075,712
dense_1 (Dense)	(None, 4)	516

Total params: 11,169,476 (42.61 MB)

Trainable params: 11,169,476 (42.61 MB)

Non-trainable params: 0 (0.00 B)

[12]:
```
model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])
```

[13]:
```
model.fit(train_generator, epochs=4, validation_data=val_generator,
batch_size= BATCH_SIZE)
```

```
/usr/local/lib/python3.11/dist-
packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121:
UserWarning: Your `PyDataset` class should call `super().__init__(**kwargs)` in
its constructor. `**kwargs` can include `workers`, `use_multiprocessing`,
`max_queue_size`. Do not pass these arguments to `fit()`, as they will be
ignored.
  self._warn_if_super_not_called()
Epoch 1/4
1/1             4s 4s/step -
accuracy: 0.0000e+00 - loss: 1.3793 - val_accuracy: 1.0000 - val_loss: 0.0017
Epoch 2/4
1/1             1s 753ms/step -
accuracy: 1.0000 - loss: 0.0017 - val_accuracy: 1.0000 - val_loss: 1.7881e-07
Epoch 3/4
1/1             1s 641ms/step -
accuracy: 1.0000 - loss: 1.6391e-07 - val_accuracy: 1.0000 - val_loss:
0.0000e+00
Epoch 4/4
1/1             1s 737ms/step -
accuracy: 1.0000 - loss: 0.0000e+00 - val_accuracy: 1.0000 - val_loss:
0.0000e+00
```

[13]:
```
<keras.src.callbacks.history.History at 0x79e3590eed10>
```

[14]:
```
model.save('/content/drive/MyDrive/segmented_images/train.h5')
```

```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or
`keras.saving.save_model(model)`. This file format is considered legacy. We
recommend using instead the native Keras format, e.g.
`model.save('my_model.keras')` or `keras.saving.save_model(model,
'my_model.keras')`.
```

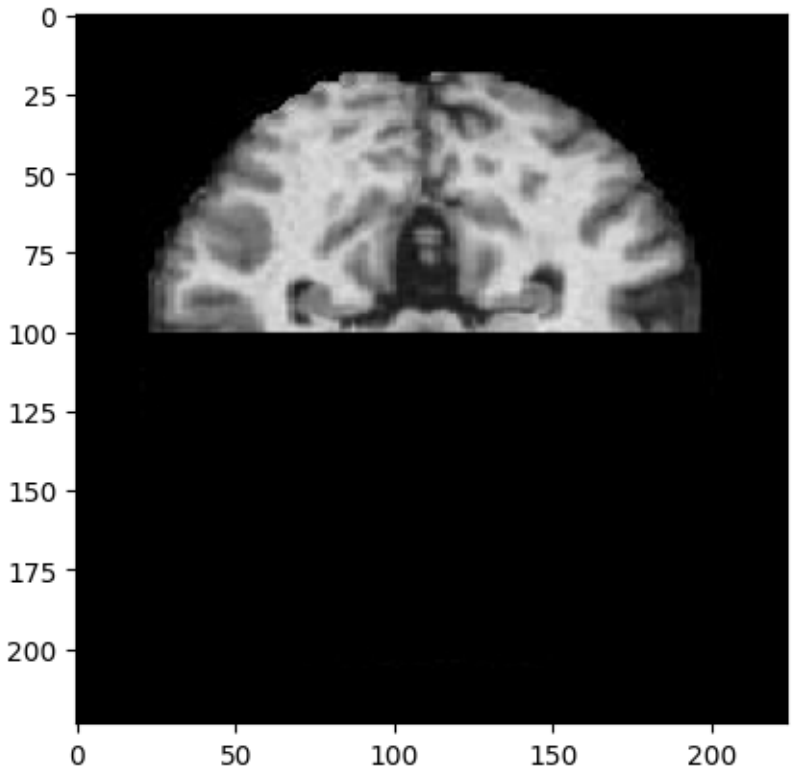
[15]:
```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
model = load_model('/content/drive/MyDrive/segmented_images/train.h5')
print("Model Loaded")
```

```
WARNING:absl: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.

Model Loaded

[16]:
```
test_image_path="/content/drive/MyDrive/segmented_images/train/VeryMildDemented/
  ↪verymildDem0.jpg"
img=image.load_img(test_image_path,target_size=(224,224))
plt.imshow(img)
plt.axis()
plt.show()
```



[17]:
```
img_array=image.img_to_array(img)
img_array=np.expand_dims(img_array,axis=0)
```

[18]:
```
img_array /= 255.
```

[19]:
```
prediction = model.predict(img_array)
print(prediction)
```

```
1/1             0s 104ms/step
[[4.3765406e-20 2.5735474e-16 9.4075019e-17 1.0000000e+00]]
```

[21]:
```
prediction=model.predict(img_array)
ind=np.argmax(prediction[0])
print(class_names[ind])
```

```
1/1             0s 67ms/step
VeryMildDemented
```

[ ]:
```

```