```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.preprocessing.image import ImageDataGenerator
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).
```

```
IMG_SIZE = (256, 256)
BATCH_SIZE = 32

train_datagen = ImageDataGenerator(rescale=1./255,
validation_split=0.2)
train_generator=train_datagen.flow_from_directory(
    '/content/drive/MyDrive/Brain Tumour',
    target_size = IMG_SIZE,
    batch_size=BATCH_SIZE,
    class_mode='binary',
    subset='training'
)
```

```
Found 800 images belonging to 1 classes.
```

```
val_generator=train_datagen.flow_from_directory(
    '/content/drive/MyDrive/Brain Tumour',
    target_size = IMG_SIZE,
    batch_size=BATCH_SIZE,
    class_mode='binary',
    subset='validation'
)
```

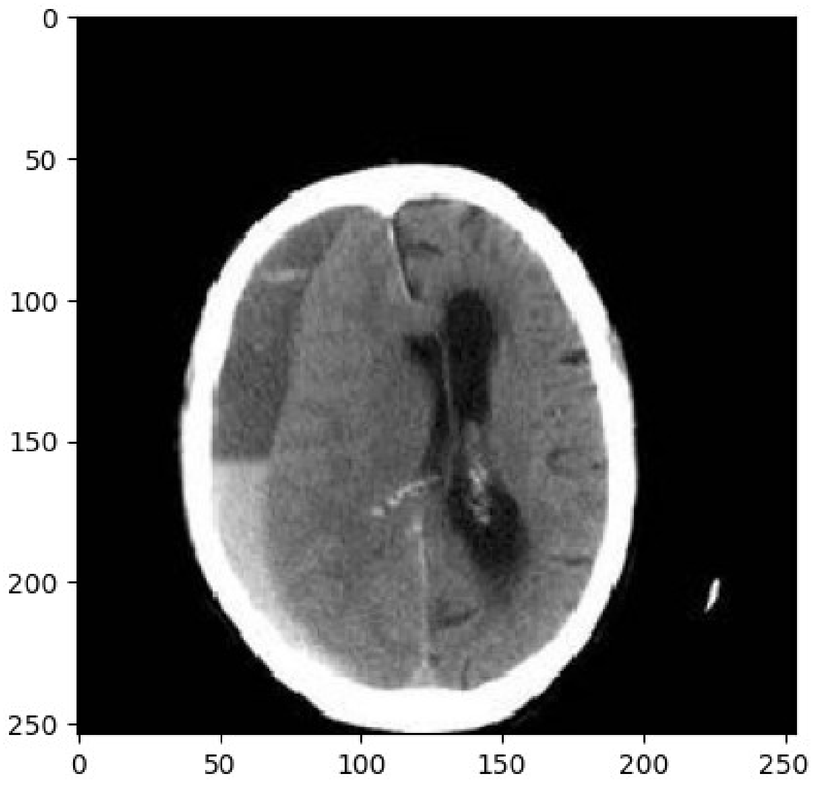
```
Found 200 images belonging to 1 classes.
```

```
model = keras.Sequential([
    layers.Conv2D(32, (3,3), activation='relu',
input_shape=(254,254,3)),
    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')
])
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/layers/
convolutional/base_conv.py:107: UserWarning: Do not pass an
`input_shape`/`input_dim` argument to a layer. When using Sequential
models, prefer using an `Input(shape)` object as the first layer in
the model instead.
  super().__init__(activity_regularizer=activity_regularizer,
**kwargs)

model.summary()

Model: "sequential_18"

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━
━━━━━┓
┃ Layer (type)                     ┃ Output Shape           ┃
Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━
━━━━━┩
│ conv2d_57 (Conv2D)               │ (None, 252, 252, 32)   │
896 │
├──────────────────────────────────┼────────────────────────┼──────────
─────┤
│ max_pooling2d_57 (MaxPooling2D) │ (None, 126, 126, 32)   │
0 │
├──────────────────────────────────┼────────────────────────┼──────────
─────┤
│ conv2d_58 (Conv2D)               │ (None, 124, 124, 64)   │
18,496 │
├──────────────────────────────────┼────────────────────────┼──────────
─────┤
│ max_pooling2d_58 (MaxPooling2D) │ (None, 62, 62, 64)     │
0 │
├──────────────────────────────────┼────────────────────────┼──────────
─────┤
│ conv2d_59 (Conv2D)               │ (None, 60, 60, 128)    │
73,856 │
├──────────────────────────────────┼────────────────────────┼──────────
─────┤
│ max_pooling2d_59 (MaxPooling2D) │ (None, 30, 30, 128)    │
0 │
├──────────────────────────────────┼────────────────────────┼──────────
─────┤
│ flatten_18 (Flatten)             │ (None, 115200)         │
0 │
├──────────────────────────────────┼────────────────────────┼──────────
─────┤
│ dense_36 (Dense)                 │ (None, 128)            │
14,745,728 │
├──────────────────────────────────┼────────────────────────┼──────────
─────┤
```

```
│ dense_37 (Dense)                │ (None, 1)              │
129 │
└─────────────────────────────────┴────────────────────────┴───────────
──────┘

 Total params: 14,839,105 (56.61 MB)

 Trainable params: 14,839,105 (56.61 MB)

 Non-trainable params: 0 (0.00 B)

model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
model.fit(train_generator, epochs=5, validation_data=val_generator,
batch_size=BATCH_SIZE)

Epoch 1/5
25/25 ━━━━━━━━━━━━━━━━━━━━ 147s 6s/step - accuracy: 1.0000 - loss:
3.7039e-04 - val_accuracy: 1.0000 - val_loss: 0.0000e+00
Epoch 2/5
25/25 ━━━━━━━━━━━━━━━━━━━━ 136s 5s/step - accuracy: 1.0000 - loss:
0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00
Epoch 3/5
25/25 ━━━━━━━━━━━━━━━━━━━━ 133s 5s/step - accuracy: 1.0000 - loss:
0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00
Epoch 4/5
25/25 ━━━━━━━━━━━━━━━━━━━━ 144s 6s/step - accuracy: 1.0000 - loss:
0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00
Epoch 5/5
25/25 ━━━━━━━━━━━━━━━━━━━━ 192s 5s/step - accuracy: 1.0000 - loss:
0.0000e+00 - val_accuracy: 1.0000 - val_loss: 0.0000e+00

<keras.src.callbacks.history.History at 0x780870ed4390>

model.save('/MyDrive/Brain Tumor.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')`.

from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import matplotlib.pyplot as plt
import numpy as np
model = load_model('/MyDrive/Brain Tumor.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

test_image_path="/content/drive/MyDrive/Brain Tumour/data/No/Te-
noTr_0000.jpg"
img = image.load_img(test_image_path, target_size=(254, 254))
plt.imshow(img)
plt.axis()
plt.show()
```



```
img_array=image.img_to_array(img)
img_array=np.expand_dims(img_array, axis=0)
img_array /=255.

prediction=model.predict(img_array)
print(prediction)

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 178ms/step
[[0.]]

if prediction >= 0.5:
  print("You have brain tumour")
```

```python
else:
  print("You do not have brain tumour")
```

```
You do not have brain tumour
```