NeuroVision

MRI-based Brain Tumor Detection using Deep Learning

INTRODUCTION

- Brain tumors are abnormal growth of brain cells that can be **life-threatening**.
- MRI is the most widely used, safe, and effective imaging technique for brain diagnosis.
- Manual interpretation of MRI scans requires expertise and is time-consuming.
- Deep Learning provides an **automated, accurate, and faster solution** for classifying tumors.

Problem Statement

- Current MRI tumor diagnosis faces challenges:
  - **Time-consuming** manual analysis.
  - **Inter-observer variability** among radiologists.
  - Difficulty in distinguishing between different tumor types.
- There is a need for a **reliable automated system** to classify tumors accurately.

Proposed Solution

- Use **ResNet50 (Transfer Learning)** for automatic classification of brain tumors into:
  - **Glioma**
  - **Meningioma**
  - **Pituitary**
  - **No Tumor**
- Improve generalization with **data augmentation** and **class weighting.**

Dataset Resource

- **Kaggle Brain Tumor MRI Dataset**
  - Images categorized into Training (Augmented) and Testing sets.
  - Training Data contains 5049 images.
  - Testing Data contains 1311 images.
- Dataset distribution:
  - **Glioma Tumor**
  - **Meningioma Tumor**
  - **No Tumor**
  - **Pituitary Tumor**
- Each image resized to 224×224 pixels for uniformity.

Design

a. **Dataset Design**

- Preprocessing steps:
  - Image resizing (224×224).
  - Normalization (pixel values scaled to 0–1).
  - Data augmentation: rotation, zoom, width/height shifts, horizontal flips.

b. **Algorithm**

- Transfer Learning with ResNet50.
- Architecture:
  - ResNet50 base (frozen)
  - Global Average Pooling layer
  - Dense (256 neurons, ReLU)
  - Dropout (0.5 to reduce overfitting)
  - Output Dense (4 neurons, softmax for multi-class).

c. **Workflow**

1. Data Loading & Preprocessing
2. Model Building with ResNet50
3. Training with EarlyStopping + ReduceLROnPlateau callbacks
4. Fine-tuning of ResNet50 layers
5. Evaluation using Accuracy, Confusion Matrix, ROC curves
6. Prediction on unseen MRI scans

Implementation Details

a. Tools & Frameworks Used

- Python, TensorFlow/Keras, OpenCV, Matplotlib, Seaborn.
- Training with GPU support for faster computation.

b. Models Used

- ResNet50 (pretrained on ImageNet) for feature extraction.
- Fully connected layers for classification.

Evaluation Results

Test Accuracy: ~97.94% (as per notebook runs).

Validation Test Accuracy: ~99.08%

Loss Accuracy: ~5.71%

Validation Loss Accuracy: ~3.5%

Results

Accuracy: ~97.94% (as per notebook runs).

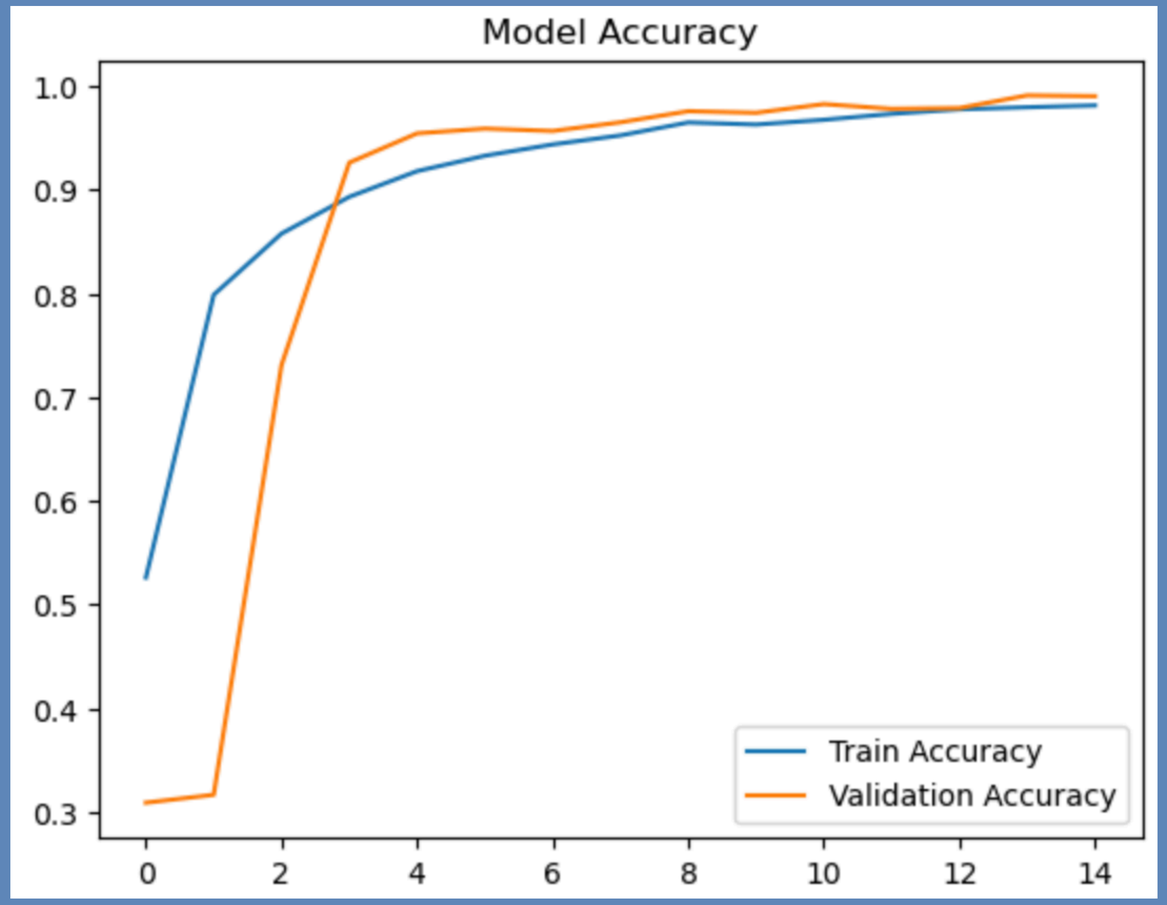
```
421/421 ━━━━━━━━━━━━━━━━━━━━ 453s 1s/step - accuracy: 0.9628 - loss: 0.1017 - val_accuracy: 0.9741 - val_loss: 0.07
68 - learning_rate: 1.0000e-05
Epoch 11/15
421/421 ━━━━━━━━━━━━━━━━━━━━ 438s 1s/step - accuracy: 0.9673 - loss: 0.0863 - val_accuracy: 0.9825 - val_loss: 0.04
75 - learning_rate: 1.0000e-05
Epoch 12/15
421/421 ━━━━━━━━━━━━━━━━━━━━ 439s 1s/step - accuracy: 0.9731 - loss: 0.0708 - val_accuracy: 0.9779 - val_loss: 0.05
30 - learning_rate: 1.0000e-05
Epoch 13/15
421/421 ━━━━━━━━━━━━━━━━━━━━ 452s 1s/step - accuracy: 0.9774 - loss: 0.0649 - val_accuracy: 0.9786 - val_loss: 0.05
99 - learning_rate: 1.0000e-05
Epoch 14/15
421/421 ━━━━━━━━━━━━━━━━━━━━ 442s 1s/step - accuracy: 0.9794 - loss: 0.0571 - val_accuracy: 0.9908 - val_loss: 0.03
50 - learning_rate: 1.0000e-05
Epoch 15/15
421/421 ━━━━━━━━━━━━━━━━━━━━ 456s 1s/step - accuracy: 0.9812 - loss: 0.0578 - val_accuracy: 0.9901 - val_loss: 0.03
29 - learning_rate: 1.0000e-05
Restoring model weights from the end of the best epoch: 14.
```

```python
# Evaluate on test set
loss, accuracy = model.evaluate(test_data)
print(f"\nTest Accuracy: {accuracy*100:.2f}%")
```

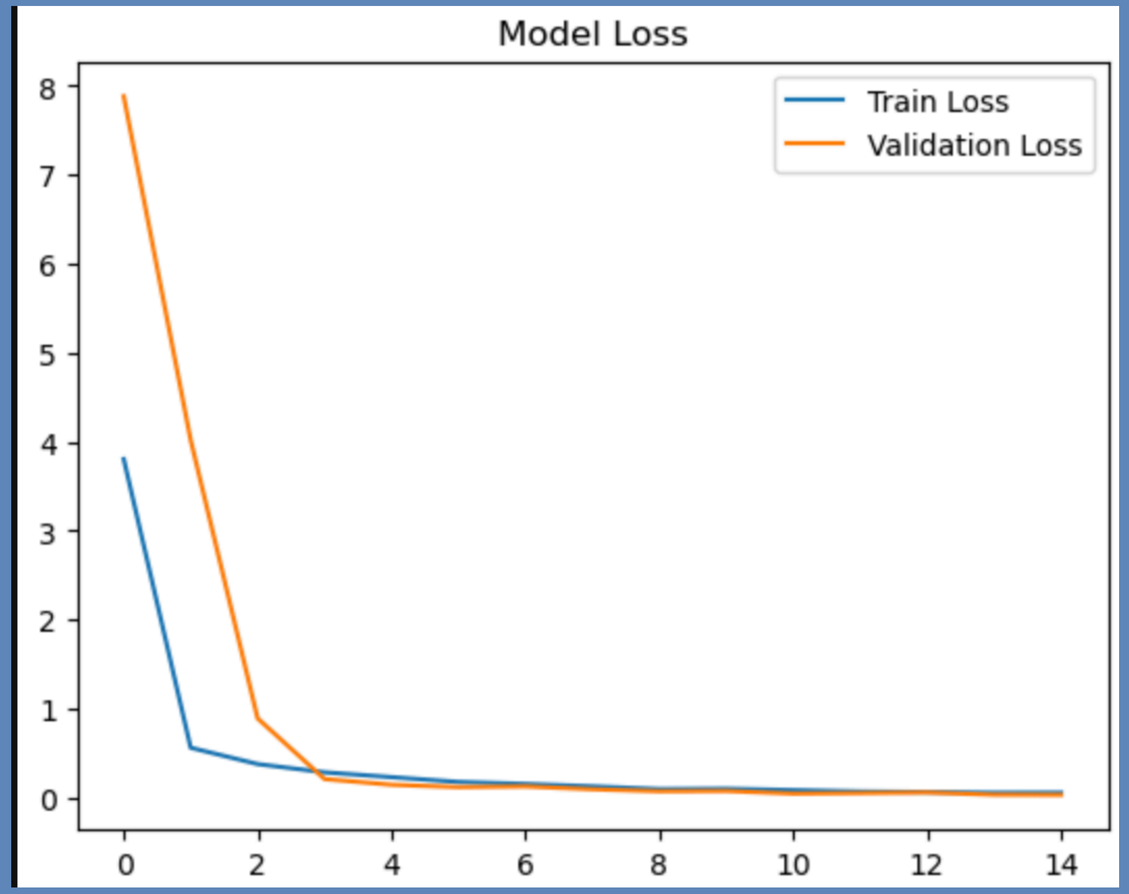
```
110/110 ━━━━━━━━━━━━━━━━━━━━ 17s 154ms/step - accuracy: 0.9908 - loss: 0.0350

Test Accuracy: 99.08%
```

Evaluation Results

Training/Validation Accuracy Curve

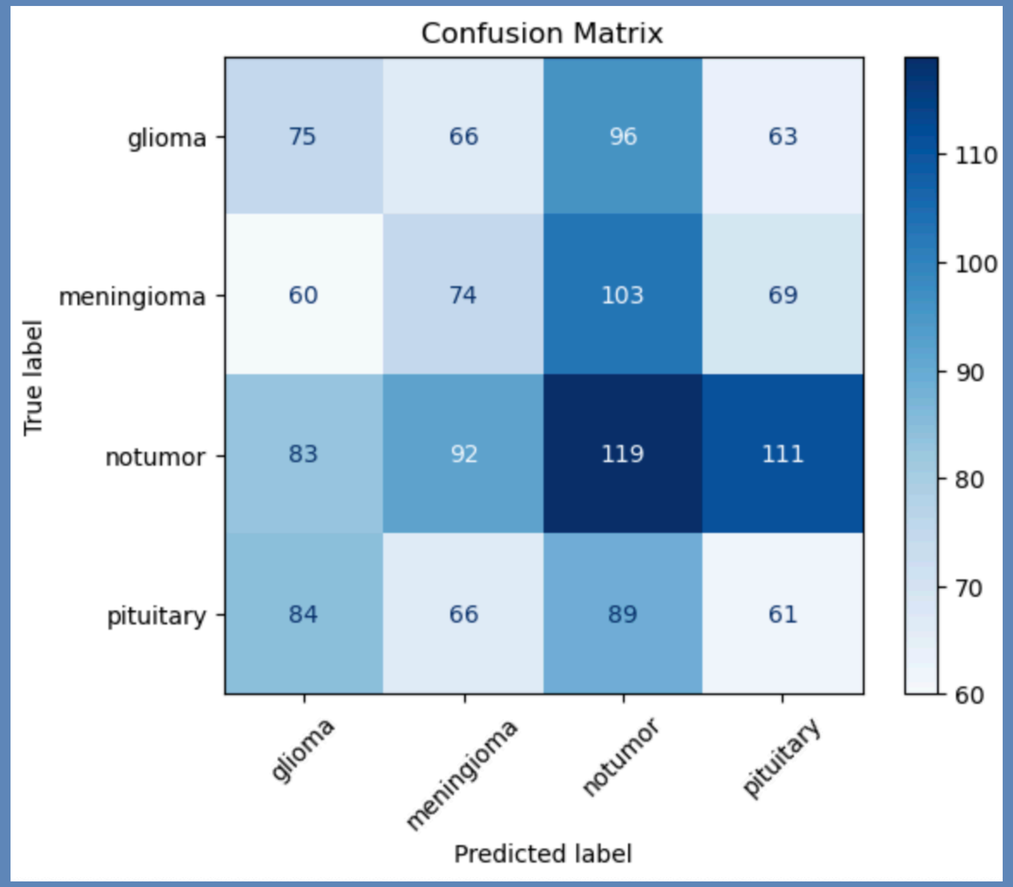


Training/Validation Loss Curve

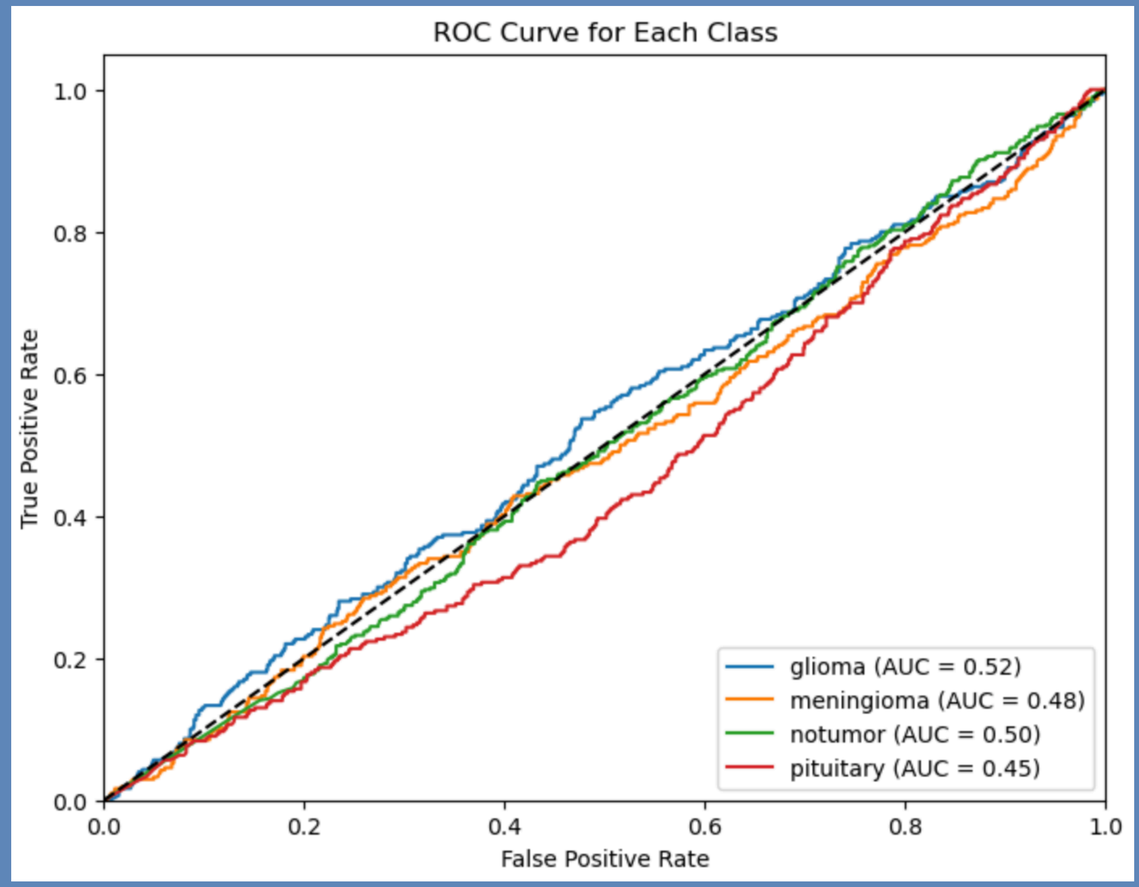


Evaluation Results

Training/Validation Accuracy Curve

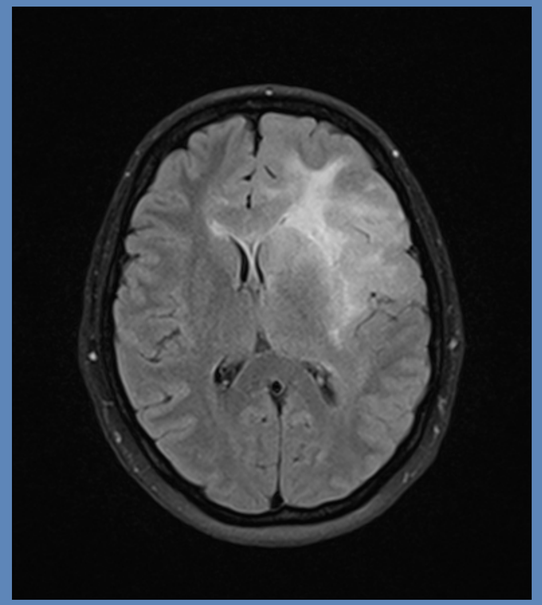


Training/Validation Loss Curve



Test Results

0 - Glioma, 1- Meningioma, 2-No tumor, 3-Pituitary



```
img = image.load_img("/Users/amithajayan/Downloads/glioma1.jpeg", target_size=(150, 150))
img_array = image.img_to_array(img)/255.0
img_array = np.expand_dims(img_array, axis=0)

prediction = model.predict(img_array)
predicted_class = np.argmax(prediction)
print(predicted_class)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 51ms/step
2
```

Glioma 1

Test Results

0 - Glioma, 1- Meningioma, 2-No tumor, 3-Pituitary

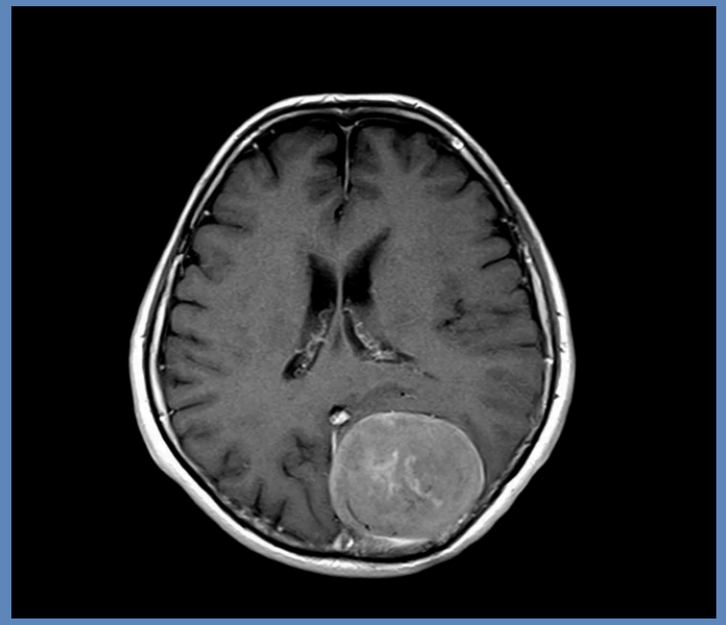
Glioma 2

```
img = image.load_img("/Users/amithajayan/Downloads/glioma2.png", target_size=(150, 150))
img_array = image.img_to_array(img)/255.0
img_array = np.expand_dims(img_array, axis=0)

prediction = model.predict(img_array)
predicted_class = np.argmax(prediction)
print(predicted_class)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 90ms/step
2
```

Test Results

0 - Glioma, 1- Meningioma, 2-No tumor, 3-Pituitary



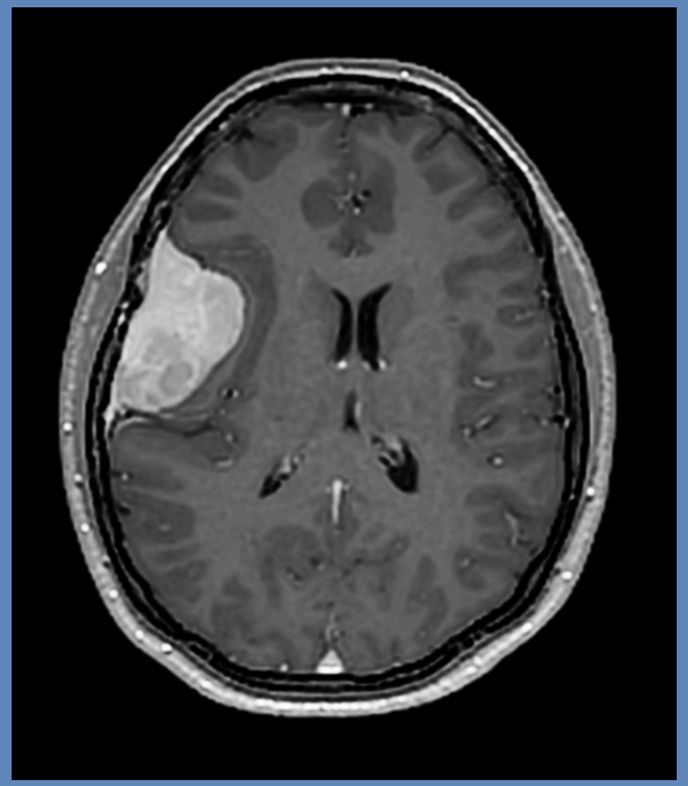
```
img = image.load_img("/Users/amithajayan/Downloads/mengioma1.png", target_size=(150, 150))
img_array = image.img_to_array(img)/255.0
img_array = np.expand_dims(img_array, axis=0)

prediction = model.predict(img_array)
predicted_class = np.argmax(prediction)
print(predicted_class)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 51ms/step
1
```

Meningioma 1

Test Results

0 - Glioma, 1- Meningioma, 2-No tumor, 3-Pituitary



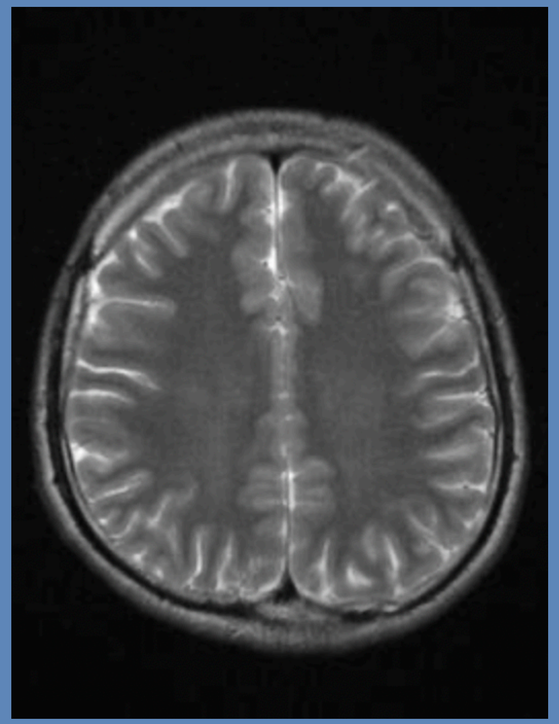
```
img = image.load_img("/Users/amithajayan/Downloads/mengioma2.png", target_size=(150, 150))
img_array = image.img_to_array(img)/255.0
img_array = np.expand_dims(img_array, axis=0)

prediction = model.predict(img_array)
predicted_class = np.argmax(prediction)
print(predicted_class)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 86ms/step
1
```

Meningioma 2

Test Results

0 - Glioma, 1- Meningioma, 2-No tumor, 3-Pituitary



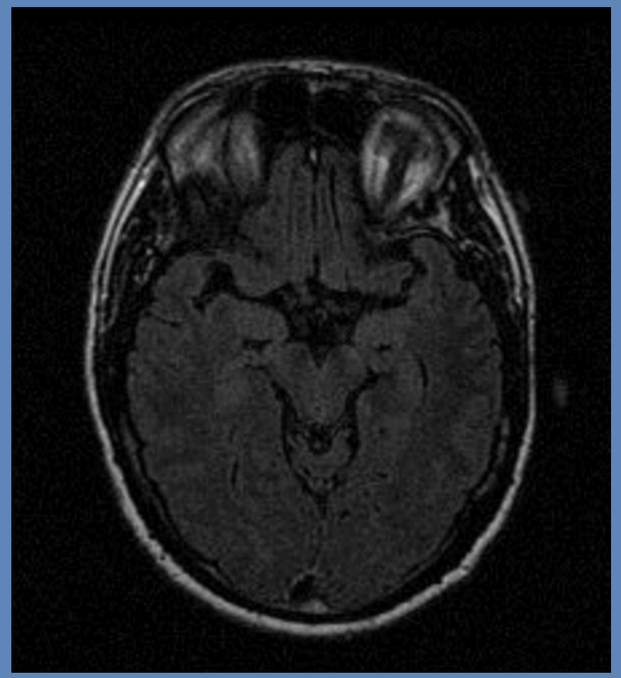
No Tumor 1

```
img = image.load_img("/Users/amithajayan/Downloads/notumor1.png", target_size=(150, 150))
img_array = image.img_to_array(img)/255.0
img_array = np.expand_dims(img_array, axis=0)

prediction = model.predict(img_array)
predicted_class = np.argmax(prediction)
print(predicted_class)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 166ms/step
2
```

Test Results

0 - Glioma, 1- Meningioma, 2-No tumor, 3-Pituitary



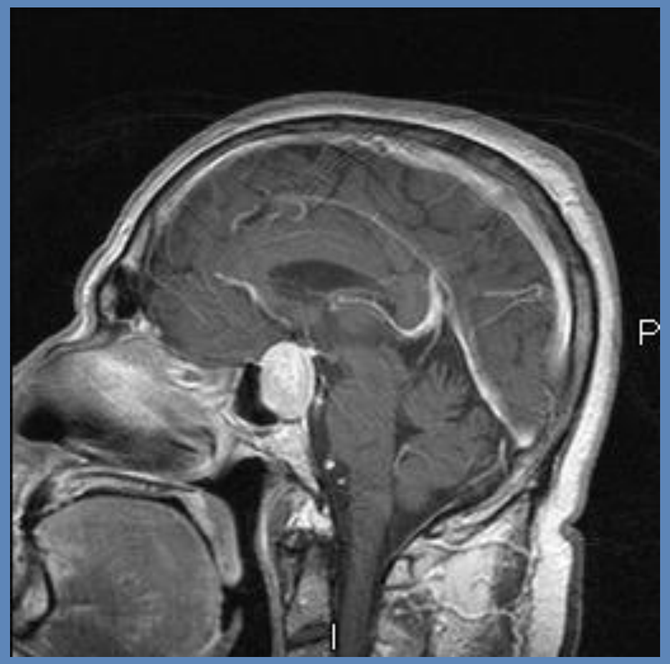
No Tumor 2

```
img = image.load_img("/Users/amithajayan/Downloads/notumor2.png", target_size=(150, 150))
img_array = image.img_to_array(img)/255.0
img_array = np.expand_dims(img_array, axis=0)

prediction = model.predict(img_array)
predicted_class = np.argmax(prediction)
print(predicted_class)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 51ms/step
2
```

Test Results

0 - Glioma, 1- Meningioma, 2-No tumor, 3-Pituitary



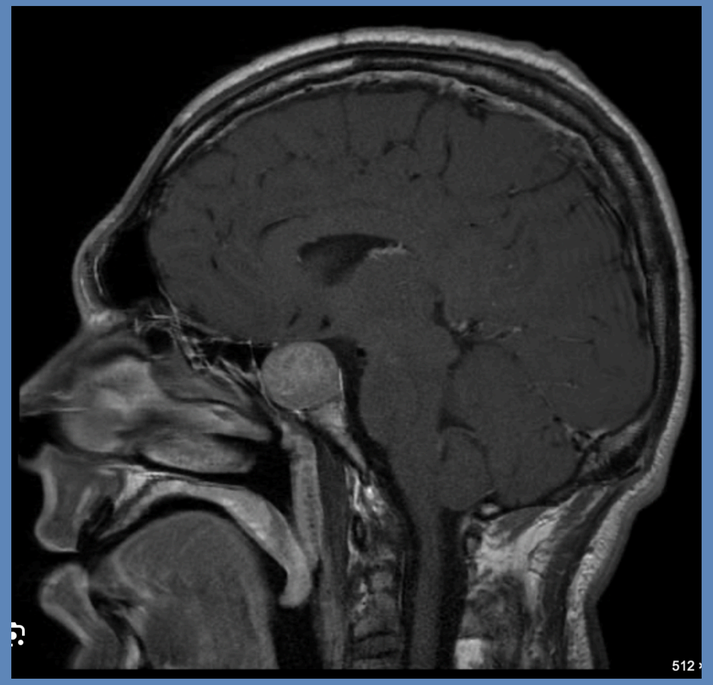
Pituitary 1

```
img = image.load_img("/Users/amithajayan/Downloads/pituitary1.jpg", target_size=(150, 150))
img_array = image.img_to_array(img)/255.0
img_array = np.expand_dims(img_array, axis=0)

prediction = model.predict(img_array)
predicted_class = np.argmax(prediction)
print(predicted_class)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 52ms/step
3
```

Test Results

0 - Glioma, 1- Meningioma, 2-No tumor, 3-Pituitary



Pituitary 2

```
img = image.load_img("/Users/amithajayan/Downloads/pituitary2.png", target_size=(150, 150))
img_array = image.img_to_array(img)/255.0
img_array = np.expand_dims(img_array, axis=0)

prediction = model.predict(img_array)
predicted_class = np.argmax(prediction)
print(predicted_class)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 147ms/step
3
```

Conclusion

- The project demonstrates that **ResNet50 transfer learning** is highly effective for brain tumor classification.
- The model reduces reliance on radiologists for initial tumor screening.
- Achieves **high accuracy and robustness** with limited preprocessing.

Future Scope

- **Binary → Multi-class → Segmentation**: Extend from classification to precise tumor segmentation.
- Deploy model as a **web/desktop medical application**.
- Use larger, multi-institutional datasets for better generalization.
- Explore ensemble models and attention mechanisms for improved accuracy.

References

- Kaggle: Brain Tumor MRI Dataset.
- Quest old projects/resources
- Senior data engineers
- Keras/TensorFlow Documentation.
- Research papers on Deep Learning in medical imaging.
- Chatgpt