Introduction

The system is designed to classify brain MRI images into four categories: **glioma, meningioma, no tumor, and pituitary tumor.** The AI model will support healthcare professionals in diagnosing brain conditions using machine learning and image processing.

It uses transfer learning with the **ResNet50** architecture as the backbone, fine-tuning it to classify images into four categories

Functional & Non-Functional Requirements

1 Preprocessing

- **Description:** The system must preprocess the input images to ensure consistency in size, format, and quality.
- **Input:** Raw MRI images in various formats.
- **Output:** Preprocessed images of size 200x200 pixels with enhanced contrast.

2 Model Training

- **Description:** The system must train a deep learning model using the ResNet-50 architecture with fine-tuning.
- **Input:** Preprocessed training dataset.
- **Output:** A trained model with saved weights and performance metrics.

3 Prediction

- **Description:** The system must predict the category of a given MRI image.
- **Input:** A single MRI image.
- **Output:** The predicted category and confidence score.

4 Performance Metrics

- **Description:** The system must provide classification reports and confusion matrices for performance evaluation.

5 Model Deployment

- **Description:** The system must load the trained model and provide a user interface to classify new images.

About Transfer Learning in Deep Learning

Transfer learning is a machine learning technique where a model developed for one task is reused as the starting point for a model on a second, related task. Instead of training a model from scratch, we use a pretrained model as a base and fine-tune it for your specific use case as computational budget is constrained

About ResNet50 and Its Architecture

- **ResNet50** uses residual connections, which are shortcuts added to the traditional convolutional layers. These shortcuts allow the gradient to flow directly through the network without vanishing or exploding, solving the vanishing gradient problem in deep networks.
- **50 Layers Deep**: The "50" in ResNet50 refers to its depth, comprising 50 layers of operations, including convolutional, batch normalization, ReLU activation, and fully connected layers.
- **Bottleneck Architecture**: ResNet50 employs a "bottleneck" design that reduces computational complexity while maintaining high accuracy. Each residual block consists of three layers:
  - **1x1 convolution** for dimensionality reduction.
  - **3x3 convolution** for feature extraction.
  - **1x1 convolution** for dimensionality restoration.
- **Pretrained on ImageNet**: ResNet50 is often pretrained on the ImageNet dataset, making it a powerful feature extractor for transfer learning tasks.
- The architecture is composed of:
  1. **Initial Convolutional Layer**: 7x7 convolution with 64 filters followed by max pooling.
  2. **4 Residual Stages**: Each stage contains a series of residual blocks:
     - Stage 1: 3 blocks (64 filters each).
     - Stage 2: 4 blocks (128 filters each).
     - Stage 3: 6 blocks (256 filters each).
     - Stage 4: 3 blocks (512 filters each).
  3. **Global Average Pooling**: Reduces spatial dimensions to a single value per filter.
  4. **Fully Connected Layer**: Outputs class probabilities in classification tasks.

Datasets : Dataset Link From Kaggle

System Architecture

1.1 Overview

The system is designed using a modular architecture with the following components:

- **Data Preprocessing Module:** Handles image resizing, filtering, and normalization.
- **Model Training Module:** Trains the ResNet-50-based deep learning model.
- **Prediction Module:** Loads the trained model and predicts the class of new images.

2.1 Data Preprocessing Module

- **Inputs:**

    - Raw MRI images.
- **Processes:**
    - Resize images to 200x200 pixels.
    - Apply bilateral filtering for noise reduction.
    - Enhance image contrast using color mapping.
- **Outputs:**
    - Preprocessed image arrays.

2.2 Model Training Module

- **Inputs:**
    - Preprocessed training dataset.
- **Processes:**
    - Load ResNet-50 architecture.
    - Fine-tune layers for brain tumor classification.
    - Train the model with data augmentation.
    - Save model weights.
- **Outputs:**
    - Trained model and evaluation metrics.

2.3 Prediction Module

- **Inputs:**
    - Preprocessed MRI image.
- **Processes:**
    - Load the trained model.
    - Predict the category of the MRI image.
- **Outputs:**
    - **Predicted category and confidence score.**

2.4 Prediction Interface

- **Input:** File upload field for an MRI image.
- **Output:**
    - Display predicted category.
    - Confidence score.
    - **Option to visualize the input image and its classification.**

4.2 Software Requirements

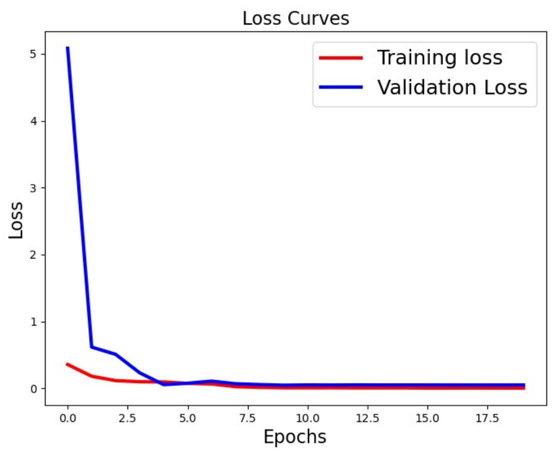
Python 3.8+, TensorFlow 2.0+, OpenCV 4.0+ ,Scikit-learn ,Matplotlib

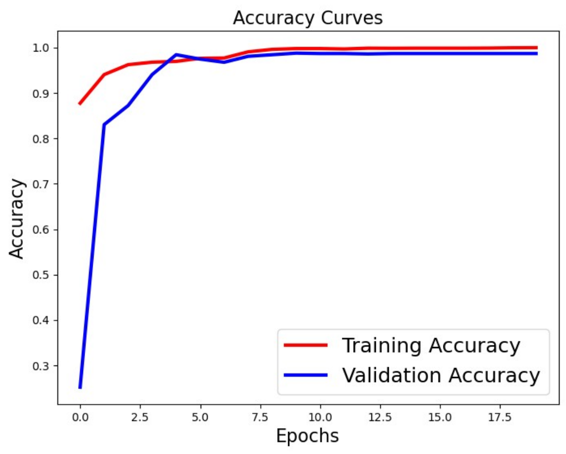
Result of Training

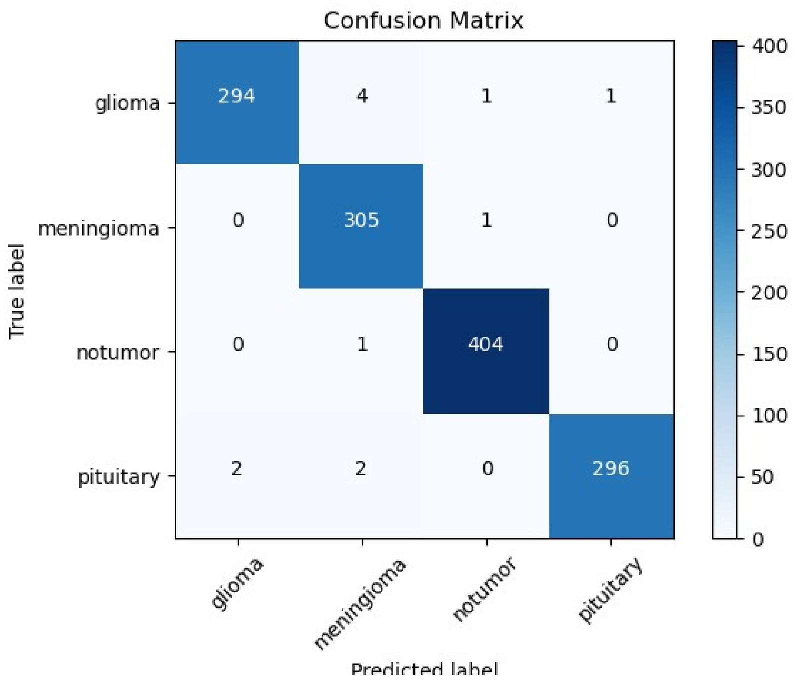
```
Epoch 20: ReduceLROnPlateau reducing learning rate to 2.1869998079182552e-08.
572/572 ━━━━━━━━━━━━━━━━━━━━ 2456s
4s/step - accuracy: 0.9999 - loss: 0.0019 - val_accuracy: 0.9869 - val_loss: 0.0490
learning_rate: 7.2900e-08
```

	precision	recall	f1-score	support
glioma	0.99	0.98	0.99	300
meningioma	0.98	1.00	0.99	306
notumor	1.00	1.00	1.00	405
pituitary	1.00	0.99	0.99	300

accuracy			0.99	1311
macro avg	0.99	0.99	0.99	1311
weighted avg	0.99	0.99	0.99	1311







5. Future Enhancements

- Implement real-time image acquisition using MRI scanners.
- Deploy the model as a web or mobile application.
- Improve accuracy using ensemble methods or advanced architectures.

Submitted By:

Abhijith K V

Ayush P S

Hemanth Krishna A B