Understanding Convolutional Neural Networks (CNN)

Introduction

Convolutional Neural Networks (CNNs) are a specialized type of deep learning model designed for image-related tasks such as classification, object detection, and segmentation. In this project, we utilized CNNs to classify MRI scans into several tumor and non-tumor categories using the **Brain Tumor MRI Dataset** from Kaggle (source).

Why Use CNNs?

Traditional neural networks struggle with image data as they do not account for spatial hierarchies. CNNs, however, we can use convolutional layers to detect essential patterns such as edges, textures, and shapes. This makes them highly effective for medical image recognition and classification tasks.

CNN Architecture Used

Our CNN model consists of the following layers:

1. **Convolutional Layers –** Extracts features from MRI images using filters/kernels.
2. **Activation Function (ReLU) –** Introduces non-linearity to enhance learning capability.
3. **Pooling Layers –** Reduces spatial dimensions while preserving crucial information.
4. **Fully Connected Layers –** Interprets the extracted features and makes final predictions.
5. **Softmax Layer –** Outputs a probability distribution for classification tasks.

Implementation Steps

1. Data Preprocessing

- Loaded the MRI dataset from Kaggle.
- Normalized pixel values to improve training efficiency.
- Split the dataset into training and validation sets.

2. Building the CNN Model

- Defined the CNN architecture using TensorFlow/Keras.
- Configured convolutional layers, pooling layers, and dense layers.
- Implemented dropout to prevent overfitting.

3. Training the Model

- Compiled the model using the Adam optimizer and Sparse categorical cross entropy as loss function.
- Trained the model using multiple epochs and batch sizes.
- Monitored training loss and accuracy for optimization.

4. Evaluating Performance

- Assessed the model using the test dataset.
- Measured accuracy and loss metrics to determine effectiveness.
- Visualized predictions and performance using matplotlib.

5. Saving and Deploying the Model

- Saved the trained model for future inference.
- Loaded the saved model to make predictions on new MRI scans.

Results

- Achieved an accuracy of **94%** on the test dataset.
- Successfully classified MRI images with high precision.
- Analyzed misclassified images to identify performance gaps and areas for improvement.

Future Enhancements

- Experimenting with advanced architectures like ResNet or VGG.
- Fine-tuning the model with transfer learning for enhanced accuracy.

Conclusion

CNNs are highly effective for medical image classification tasks. In this project, we built and trained a CNN model that successfully recognized patterns in MRI scans. By optimizing the architecture and fine-tuning hyperparameters, we can further improve the model's accuracy and robustness.