Brain Tumour Detection

The main goal of this project was to detect brain tumours using Convolutional Neural Networks (CNNs). This model classifies brain images into two categories 'Healthy' and 'Brain Tumor'. The original dataset was split into training and validation sets. The model was trained on the training set and its performance was evaluated using the validation set.

Dataset

The dataset consisted of 4526 MRI brain scans which were classified into two categories 'Healthy' and 'Brain Tumor'. There were 54.02% images labelled as 'Brain Tumor' and 45.98% labelled as 'Healthy'.

Preprocessing of data

All the images were converted in grayscale to ensure uniformity and reduce computational complexity. As the images in the dataset varied in size, it was necessary to reshape them. All of them were resized to 128 x 128 pixels.

Splitting of data

The dataset was split into 80% training data and 20% validation data in order to evaluate the model's performance on unseen data. In order to ensure random order of images before being passed to the model, it was necessary to shuffle the train set before each epoch. Shuffling is of utmost importance when it comes to preventing models from learning patterns based on the order of the images instead of their features. The images were processed in batches of 64 in order to optimize memory usage and fasten the model training.

Architecture of CNN

CNNs consist of convolutional layers, pooling layers and fully connected layers. This CNN has two 2 convolutional and pooling layers.

- **Convolutional Layer**

The main aim of this layer was to extract significant visual features from the raw input images. This was achieved by sliding the kernel or filter over the image. In this model, a 3x3 kernel was used. At each position a dot product was calculated between kernel's weights and the corresponding pixel values of the image.

- **Pooling Layer**

The primary goal of this layer was to reduce the spatial dimensions of the feature maps, which helps with decreasing the number of parameters, speeding up computation, and preventing overfitting.There are several methods of pooling for example max and average pooling. In this case max pooling was used. Max pooling selects the maximum value within each region of the feature map, while discarding other values.

- **Fully Connected Layer**

The fully connected layer integrates the various features extracted in the previous convolutional and pooling layers and maps them to specific classes or outcomes.The softmax activation function was used to obtain probabilities for each class.

- **Dropout layer**

The purpose of the dropout layer is to reduce overfitting by dropping neurons from the neural network during training. This reduces the size of the model and helps prevent overfitting.

Model summary analysis

Conv2D Layer 1

Output shape : 126x126

This layer applies filters of size 3x3 to the input image (128x128) and produces 126x126 feature maps.

MaxPooling2D Layer 1

Output shape: 63x63

This layer performs the max pooling operation, reducing the dimensions by half from 126x126 to 63x63.

Conv2D Layer 2

Output shape : 61x61

This layer applies filters of size 3x3 to the feature maps from the previous layer.

MaxPooling2D Layer 2

Output Shape: 30x30

This layer performs another max pooling operation, reducing the dimensions from 61x61 to 30x30.

Training Performance

After training for 5 epochs, the model achieved a training accuracy of 98.32% with a training loss of 0.0585. This indicates that the model is learning the features of the training dataset effectively and minimizing the error over time.

Validation Performance

Once training was complete, the model was evaluated on the validation set. The final validation accuracy reached 95.87%, while the validation loss dropped to 0.0197. These results suggest that the model is generalizing well to unseen data without overfitting.

The complete implementation of the project can be found on my GitHub repository: https://github.com/Jovana21082003/Brain-Cancer-