Automated-tumor-detection-using-densenet201

Team Members:

1. Vaishnavi Chilumuru
2. Surya Kiran Varma Vegesna
3. Bharath
4. Srujani Mareddy
5. Naveen Kumar Reddy Singam

Link to the google colab with complete outputs:

https://colab.research.google.com/drive/1eyeDpaOfSEmXlrBahaQ4YmnkYnDbgj61?usp=sharing

Part - I

Business Problem:

Early Detection and Localization of Brain Tumors:

Current medical practice faces significant challenges in the early detection and localization of brain tumors. Traditional diagnostic methods rely heavily on the expertise of neuroradiologists to interpret complex MRI scans, a process that is both time-consuming and prone to human error. The variability in tumor appearance and the subtlety of early-stage growths can lead to delayed diagnoses and treatment initiation, adversely affecting patient outcomes. As the volume of imaging studies increases, the medical community struggles to keep pace, which can lead to longer waiting times and added stress for both patients and healthcare providers.

Proposed Solution:

Automated Deep Learning-Assisted Radiology:

Our proposal is to develop an automated, deep learning-based system that will augment radiologists' capabilities by rapidly detecting and accurately localizing brain tumors in MRI scans. By leveraging the DenseNet201 architecture known for its efficiency in image classification tasks, in tandem with ResUNet for precise segmentation, the system will provide the following advantages:

- **Rapid Analysis**: The deep learning system will significantly reduce the time required to analyze MRI scans, offering quick preliminary assessments.
- **Consistency**: It will provide consistent analysis, avoiding the variability inherent in human interpretation.
- **Diagnostic Accuracy**: By utilizing advanced pattern recognition, the system will assist in detecting tumors that might be missed or mischaracterized during manual review.
- **Early Diagnosis**: Accurate and early localization of tumors will aid in better treatment planning, potentially improving patient outcomes through timely intervention.
- **Workload Reduction**: This system will alleviate the heavy workload on radiologists, allowing them to focus on more complex cases and patient care.
- **Scalability**: Such an automated system can easily scale to handle large volumes of scans, making it a robust solution for high-demand healthcare systems.

Introduction to DenseNet201:
DenseNet201 is a convolutional neural network that is well-regarded for its efficiency and effectiveness, especially in image classification tasks. It features dense connections between layers which promote feature reuse, making it highly suitable for medical image analysis due to its ability to detect subtle features in images.

Application in Brain Tumor Detection:
In this project, DenseNet201 is employed to classify MRI scans to determine the presence of brain tumors. How DenseNet201 can be applied specifically for detecting brain tumors, possibly including modifications or adaptations to the architecture for this purpose? The proposed model processes images to predict whether a tumor is present, utilizing its deep learning capabilities to interpret complex image patterns that are characteristic of brain tumors.

Understanding DenseNet201 Architecture:

- DenseNet201 consists of multiple dense blocks where each layer is connected to every other layer in a feed-forward fashion. Within each dense block, layers are connected through 'composite functions' which typically consist of Batch Normalization (BN), followed by a Rectified Linear Unit (ReLU) activation and a Convolution (Conv) operation.

- The distinctive feature of DenseNet201 is that it allows each layer to access feature-maps from all preceding layers, thus encouraging feature reuse and drastically reducing the number of parameters. This connectivity pattern leads to substantial improvements in efficiency and effectiveness in image recognition tasks.

- After each dense block, a transition layer composed of a convolution and pooling operation reduces the size of the feature-map and prepares the data for the next block, helping to control the model complexity.

- The final layers of the network include a fully connected (FC) layer that maps the features learned by the network into the final output categories, followed by a classification layer which provides the probability distribution over these categories, identifying the most likely classification for the input image.

Data Set Description: The project uses a dataset comprising MRI scans with designated paths for images and their corresponding tumor masks.

Two important variables in the DataFrame:

- **image_path**
- **mask_path**

Data Preparation:

Image Data Pre-processing:

Step 1: Dataset Collection

The process begins with the assembly of a comprehensive dataset consisting of MRI scan images. These scans form the primary data input for the model, showcasing a variety of brain images, some of which contain tumors.

Step 2: Preprocessing Pipeline

Upon collecting the dataset, the images are subjected to a preprocessing pipeline. This pipeline is crucial for preparing the data for optimal model performance and involves the following key steps:

- **Normalization**: Each pixel value in the MRI scans is rescaled from its original range (which is typically 0 to 255 for image data) to a range of 0 to 1. This normalization step is essential for neural network models as it ensures that the gradient descent algorithm – used for optimizing the model – can converge more quickly.
- **Validation Split**: The dataset is split into different subsets. A specified percentage (in this case, 10%) of the data is separated to form the validation set. The validation set is not used during the training phase. Instead, it serves to evaluate the model's performance on unseen data, providing an estimate of how well the model is likely to perform on external data.

Step 3: ImageDataGenerator Utility

TensorFlow's ImageDataGenerator utility is employed to automate the data preparation process. This utility augments the training dataset by generating additional training examples through various transformations, such as rotations, shifts, and flips. The goal is to expose the model to a wider variety of training scenarios, thus improving its generalization capabilities.

Step 4: Dataset Transformation and Augmentation
The transformed dataset is now ready for the training phase. The ImageDataGenerator applies the preprocessing steps to each image in the batch before the model sees it. This means that the model is always training on slightly different variations of the training data, which helps prevent overfitting and improves the model's ability to generalize from its training data to new, unseen data.

The data preparation stage is foundational to the success of the automated tumor detection system. Properly preprocessed images ensure that the DenseNet201 model can learn relevant features without bias from scale or color variations. The result is a preprocessed training dataset of images, now ready for the model to learn from.

Training the Model:

- **Model Configuration**: DenseNet201 serves as the base model, with weights initially frozen to leverage pre-trained patterns before fine-tuning for specific tumor characteristics.
- **Training Process**: The model is trained to classify images as having a tumor or not, and a separate ResUNet model is used for precise tumor segmentation.

i. An image generator is created using TensorFlow's ImageDataGenerator class, which enhances the dataset by introducing variations in the training images, such as rotations and flips, to improve model robustness.

ii. A separate data generator is set up for test images to ensure that the model is evaluated on data it has not seen during training, thereby providing an unbiased assessment of its performance.

iii. The DenseNet201 architecture is instantiated as the base model for feature extraction.

iv. The weights of the pre-trained DenseNet201 model are frozen to leverage learned features from a different dataset, which can then be fine-tuned for the specific task of tumor detection.

v. A classification head, consisting of one or more additional layers, is added to the base model to perform the task of classifying whether an MRI scan contains a tumor.

vi. Finally, the model is compiled, which involves specifying the optimizer, loss function, and metrics to be used for training.This streamlined approach to model training ensures a balance between leveraging pre-trained knowledge and adapting to the specific features of brain tumor MRI scans.

Performance and Evaluation

- **Metrics**: The model shows high precision (97-98%) and recall (95-99%), indicating excellent accuracy and the ability to identify most true positives.
- **F1-Score**: With an F1-score between 0.96 and 0.98, the model demonstrates a strong balance between precision and recall, which is crucial for medical diagnostics.

Challenges and Limitations

- **Variability of Tumor Appearance**: Tumors can vary greatly in terms of their location, shape, and size within the brain, which makes it challenging to develop a model that can accurately detect and classify all types of tumors.
- **Standardization**: There's a need for standardization in the datasets used for training these models. Inconsistent data or the absence of standardized reporting can undermine the robustness and generalizability of the models.
- **Segmentation Difficulty**: Accurate segmentation of the Region of Interest (ROI) is challenging due to tumor heterogeneity and the polymorphic nature of brain tumors. Manual segmentation is time-consuming and subjective, whereas automatic segmentation can lack precision.

Case Study:

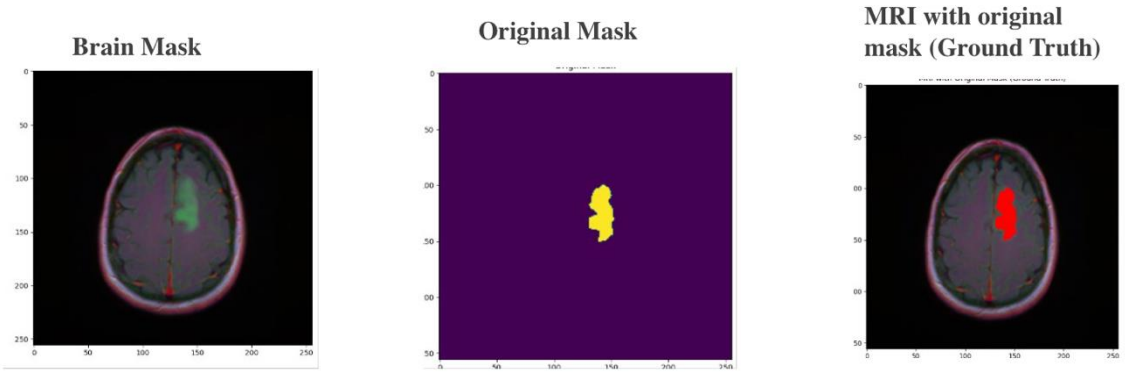
National Library of Science: Brain Tumor Detection Based on Deep Learning Approaches and Magnetic Resonance Imaging

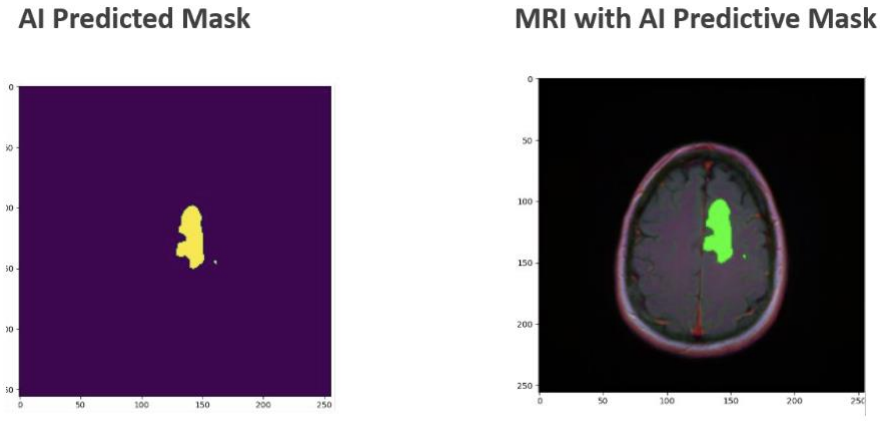
There are a large number of patients in the United States who have been diagnosed with primary brain tumors; roughly 700,000. In addition, in the United States alone, approximately 85,000 new cases of brain tumors were detected in 2021. A patient's age is just one of several factors that affect prognosis and survival rates when dealing with a brain tumor. Patients aged 55–64 had a 46.1% one-year survival rate, whereas those aged 65–74 had a 29.3% survival rate. The authors highlight the importance of early tumor detection in increasing the likelihood of survival.

Effective treatment planning and patient outcomes depend on a quick and precise diagnosis of brain tumors. However, radiologists may spend a lot of effort on image analysis when dealing with brain tumors.Today's radiologists must rely on their own skills and subjective interpretation of pictures to make detection and decisions manually.
Artificial intelligence (AI) plays a significant role in the detection and diagnosis of brain tumors, making it a useful complement to the notoriously difficult field of brain tumor surgery. Recent developments in deep learning have led to a wide range of useful applications in fields as disparate as pattern classification, object detection, voice recognition, and decision making

Results:

- **Performance Analysis**:





Future Directions

- **Enhancing Model Accuracy**: Future work could explore combining DenseNet201 with other models or using ensemble methods to improve diagnostic accuracy.
- **Expanding Dataset and Features**: Increasing the dataset size and diversity, including more varied tumor types and stages, could help in generalizing the model's capabilities.

Part II: Reinforcement Learning Strategy

Strategy Design for AI in Simplified **'Flappy Bird'** Game

Game Description:
A simplified version of 'Flappy Bird' is selected, where the AI controls a bird avatar. The bird must navigate through a series of pipes appearing at random heights with varying gaps between them. The game progressively speeds up, making it more challenging to avoid collisions.

Reinforcement Learning Elements:

- **Actions**: The AI can either make the bird 'flap' and ascend or do nothing and let it descend due to gravity.
- **States**: Each state is defined by the bird's position, the height of the next pipes, the bird's velocity, and the distance to the next pipe gap.
- **Rewards/Penalties**: The AI receives a positive reward for every set of pipes it successfully navigates without collision. Collisions with pipes or falling to the ground result in a negative reward and the end of the game.

Hypothesis for 'Action Rule' and 'Reward and State Distribution':

- The 'action rule' hypothesizes that the bird should flap based on a threshold distance from the pipe gap and its current velocity.
- A convolutional neural network (CNN) with ReLU activation functions will be utilized to interpret the game state images and decide on the action to take.
- The 'state distribution' will likely follow a Gaussian distribution centered around the pipe gaps, assuming most states will involve approaching or leaving a pipe gap.

Final Reward:
The final reward for the AI is cumulative, based on the number of pipes successfully passed. The game also considers the risk—colliding with a pipe or hitting the ground results in a penalty that ends the current game session. Hence, the reward is a combination of points earned for surviving and a terminal negative reward for failure.

Analysis Procedure:

- Collect data by manually playing the game to generate initial training data for the AI.
- Implement a CNN to process the game's visual input and predict the likelihood of survival for possible actions.
- Train the CNN using reinforcement learning, where actions taken are based on a policy derived from the current predicted survival probabilities.
- Continuously test the AI in the game environment, updating the model weights based on the received rewards or penalties.
- Iterate the process until the AI consistently achieves high scores or shows no further improvement.
- The strategy outlines a basic approach for an AI to learn and master the simplified version of 'Flappy Bird.' The CNN will analyze the current state of the game and decide when to flap the bird's wings to maintain flight and avoid obstacles. The reinforcement learning process is iterative, improving the AI's performance over time.