Mini Project Report on

Brain Tumor Detection

Submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

Submitted by:

Arsh **2021129**

Under the Mentorship of

Dr. Manoj Diwaker

Professor CSIT

Department of Computer Science and Engineering

Graphic Era (Deemed to be University)

Dehradun, Uttarakhand

January-2024

Graphic Era
Deemed to be
University
DEHRADUN

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the project report entitled **"Brain Tumor Detection"** in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering of the Graphic Era (Deemed to be University), Dehradun shall be carried out by the under the mentorship of **Dr. Manoj Diwaker, Professor CSIT**, Department of Computer Science and Engineering, Graphic Era (Deemed to be University), Dehradun.

Name: **Arsh** University Roll no: **2021129**

Table of Contents

Chapter 1

Introduction

1.1 Problem Statement

Brain tumors are becoming more common, causing a big health problem. Current ways of finding them often involve invasive procedures, which can slow down getting the right diagnosis and treatment. This project's goal is to make a smart tool using advanced machine learning that can quickly tell if an MRI image shows a tumor or not. This way, we aim to improve the efficiency and accuracy of brain tumor detection, especially in the early stages.

1.2 Overview of Brain Tumors

Brain tumors, the most prevalent form of cancer in humans, require extensive research for timely detection and treatment, posing a significant threat to both children and adults. Manifesting through symptoms like headaches, nausea, and behavioral changes, precise diagnosis and intervention are imperative. This emphasizes the urgency in developing effective diagnostic methodologies for distinguishing between benign and malignant tumors. Magnetic Resonance Imaging (MRI) has become pivotal in neuroimaging, offering non-invasive, detailed three-dimensional images crucial for analyzing structural changes linked to brain tumors. This paper aims to enhance image processing algorithms, creating an improved strategy for tumor diagnosis with a reliable graphical user interface through Computerized Image Processing (CIP). Early detection is crucial for effective treatment planning, potentially mitigating the severity of neurological disorders. Traditional manual examination of radiological images is time-consuming, prompting the need for sophisticated and efficient approaches. Recent advancements in medical imaging, such as functional MRI (fMRI), diffusion-weighted imaging (DWI), and spectroscopy, enrich the diagnostic arsenal, providing complementary information for a comprehensive understanding of tumor characteristics.

1.3 Overview of Image Classification with Python

Image classification serves as a foundational task in computer vision, involving the categorization of images into predefined classes or labels. Python has significantly contributed to the accessibility and efficiency of image classification, with frameworks like TensorFlow, Keras, and scikit-learn empowering the implementation of diverse image classification algorithms, particularly emphasizing Convolutional Neural Networks (CNNs). This technology spans across multiple sectors, including healthcare, autonomous vehicles, retail, and security.

In the healthcare sector, image classification plays a crucial role in aiding disease diagnosis, such as the detection of skin cancer. Meanwhile, in autonomous vehicles, it ensures safer navigation through the recognition of objects. Python, known for its adaptability and extensive ecosystem, emerges as the preferred choice for researchers and practitioners. It serves as a go-to tool for the development, implementation, and refinement of image classification models across a myriad of real-world applications.

Chapter 2

Literature Survey

2.1 Related Work

In 2018, [1] introduced capsule algorithms networks (DCNet) and diverse capsule networks (DCNet++). DCNet incorporated a deeper convolutional network for learning distinctive feature maps, while DCNet++ adopted a hierarchical architecture for efficient learning of complex data. The dataset consisted of 3064 MRI images of 233 brain tumor patients, focusing on three tumor types. DCNet achieved a test accuracy of 93.04%, and DCNet++ achieved 95.03%.

Abiwinanda (2018) [2] employed a CNN with the "adam" optimizer to diagnose the three most common brain tumor types. Training on 3064 T-1 weighted CE-MRI images from Cheng's dataset, the model achieved accuracy rates of 98.51% for training and 84.19% for validation.

Mittal et al. [3] combined Stationary Wavelet Transform (SWT) and a Growing CNN (GCNN) for automated brain tumor segmentation, outperforming genetic algorithms, K-NN, SVM, and CNN.

In another study (2018) [4], CNNs were applied for automatic brain tumor diagnosis, distinguishing between healthy brains and tumor images. AlexNet exhibited the best performance, achieving 99.55% accuracy in the first stage of the two-stage multi-model system.

Rehman et al. [5] investigated AlexNet, GoogLeNet, and VGGNet to differentiate meningioma, glioma, and pituitary tumors. VGG16 demonstrated the highest accuracy of 98.69%.

Paul et al. [6] employed deep learning to classify brain images related to meningioma, glioma, and pituitary tumors. Using the same dataset, they designed fully connected CNNs, achieving an accuracy of 91.43% with general methods in a fivefold cross-validation with Python.

In a 2020 study, Badža and Barjaktarovic´ utilized a CNN to classify glioma, meningioma, and pituitary tumors. Their network architecture comprised an input layer, two "A" blocks, two "B" blocks, a classification block, and an output layer, totaling 22 layers. The study employed k-fold cross-validation, achieving a best accuracy of 96.56%. The dataset included 3064 T1-weighted contrast-enhanced MRI images from Nanfang Hospital, General Hospital, and Tianjin Medical University in China [7].

Gumaei et al. [8] proposed an automated approach for brain tumor identification in three steps: brain image preprocessing, feature extraction using PCA-NGIST, and classification using Regularized Extreme Learning Machine (RELM). The dataset included 3064 MRI images from 233 patients, and a fivefold cross-validation resulted in 94.23% accuracy.

Pashaei et al. [9] developed a CNN model to identify meningioma, glioma, and pituitary tumors, comparing it with MLP, Stacking, XGBoost, SVM, and RBF. The proposed method achieved a high accuracy of 93.68% using a tenfold cross-validation on the dataset provided by Cheng.

Chapter 3

Methodology

3.1 Proposed Methodology

3.1.1 Data Collection and Extraction

The Br35H [10] Dataset is collected from Kaggle, which comprises two folders, namely "yes" indicating positive brain tumor cases, and "no" indicating no brain tumor cases. Each folder contains 1500 MRI scans. During the data extraction phase, we created a way to find folder paths in the 'Dataset' directory. Afterward, we developed a process to input a folder name, determine its path, and create a list of image files inside. This straightforward approach is crucial for managing and retrieving data efficiently from large datasets, ensuring smooth processing in our project.

```python
# Step 1 - Data Extraction
def relative_to_Dataset(foldername)->str:
    """
    Returns the path of the foldername in the Dataset folder
    """
    return str(Path.cwd() / 'Dataset' / foldername)

def list_of_images(foldername)->list:
    """
    Returns the list of images in the foldername
    """
    folder = relative_to_Dataset(foldername)
    images = os.listdir(folder)
    return images
```

Figure 3.1.1 Methods Proposed for Data Extraction

3.1.2 Data Preprocessing

Before inputting data into the model, it's essential to conduct preprocessing steps to enhance data quality and prepare it for effective training. These steps involve resizing images to a consistent resolution, normalizing pixel values to the [0, 1] range, managing class imbalances, and augmenting the dataset using techniques like rotation, flipping, or zooming to introduce diversity. Our scenario involves the utilization of OpenCV for resizing and PIL for achieving a standardized input size of 64x64 pixels for the images in the dataset. So, the images from both the "NOT BRAIN TUMOR" and "BRAIN TUMOR" categories are resized to the specified input size. Subsequently, the dataset and corresponding labels are converted into NumPy arrays for compatibility with deep learning frameworks.

3.1.3 Data Splitting

For training and evaluation we split our dataset into training and testing sets (80%-20 %) using scikit-learn, ensuring that the model generalizes well to unseen data. Before feeding the data into the neural network, we normalize the pixel values using the normalize function from Keras. It is a crucial preprocessing step that ensures consistent and effective learning during the training phase.

3.1.4 Building the Sequential Model

It involves a systematic approach. The model starts with a convolutional layer using activation and max-pooling operations. This repeats for a specified number of convolutional layers, leading to flattening. Connecting to dense layers with activation functions and dropout aids in regularization. The final layer, with softmax activation, classifies inputs into two categories. In our case,

we don't utilize pre-trained models like VGG16, ResNet, or Inception. The model is initiated with convolutional layers and activation functions, providing a foundation for diverse image classification tasks.

3.1.5 Model Compilation

Setting up the model includes defining key elements like the optimizer, loss function, and evaluation metrics. In tasks such as tumor detection, binary crossentropy is commonly chosen as the loss function. The Adam optimizer is frequently used, with accuracy or area under the ROC curve (AUC-ROC) serving as common metrics for assessment. In our scenario, the model is configured using categorical crossentropy as the loss function, the Adam optimizer, and accuracy as the metric.

3.1.6 Model Training

In the training phase, we input the prepared data from **3.1.2** into the configured model. Through successive forward and backward passes, the model grasps patterns and characteristics linked to brain tumors. We also monitor the training process by regularly checking metrics and validation data to avoid potential overfitting issues.

3.1.7 Model Evaluation

In the evaluation phase, we assess the model's performance by predicting classes for the test dataset and comparing them with the actual classes. The resulting confusion matrix visually summarizes correct and incorrect classifications. Additionally, a detailed classification report is generated, offering insights into precision, recall, F1-score, and support for each class.

These metrics collectively provide us with an understanding of the model's effectiveness in categorizing brain tumor images.

3.1.7 Graphical User Interface

To make a good-looking interface, we utilized Flask with HTML, CSS, and Tailwind CSS to achieve the desired design. The incorporation of generative AI from platforms like DALL·E and IdeogramAI for enhancing graphic elements, and optimizing the visual appeal of the user interface. The backend is powered by TensorFlow and uses the Model we trained from **3.1.6**. Rigorous and continuous testing has been employed to guarantee that the GUI aligns with user expectations, establishing an interactive platform for seamless user interaction.

3.2 Flow of Data

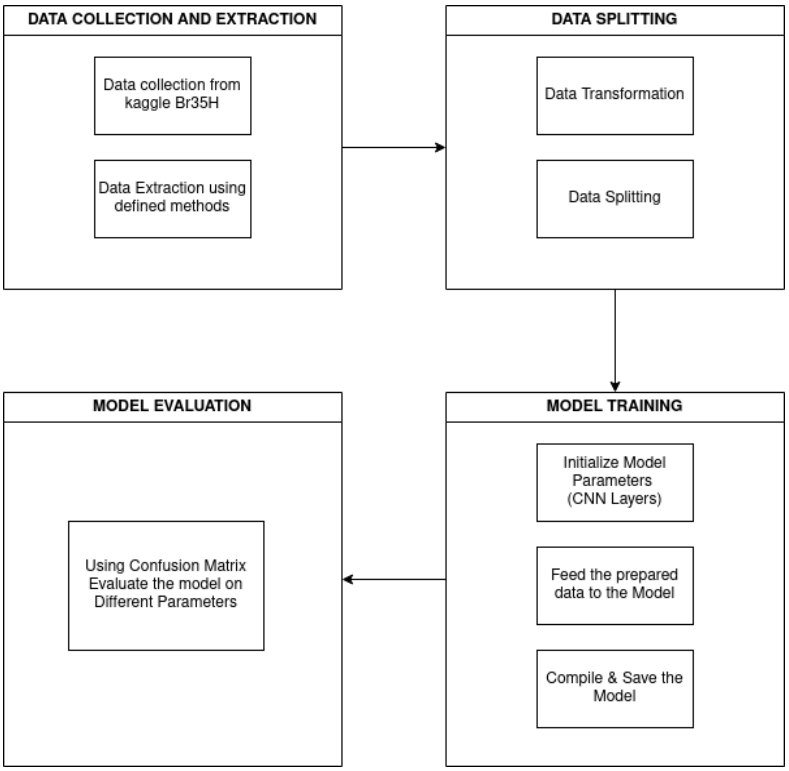

Figure 3.2.1 Data Flow

Chapter 4

Result and Discussion

4.1 Training Approach

In the training phase of the brain tumor detection model, a systematic batch processing approach was employed, utilizing 78 batches per epoch, each containing 30 images. This method aimed to efficiently train the model on a carefully curated dataset, ensuring a balanced representation of tumor-afflicted and normal conditions.

4.2 Batch Size Strategy

The initial decision to use a batch size of 30 was established with a smaller dataset of 357 images. Recognizing the necessity for dataset enrichment and improved accuracy, the model transitioned to a larger dataset of 3000 images, addressing imbalances observed in the initial dataset. This adjustment optimized the model's capacity to distinguish between brain images indicative of tumor presence and those reflecting normal conditions by combining a thoughtful choice of batch size with dataset expansion.

4.3 Model Performance and Accuracy

The model, trained on an Intel© Core™ i3-5005U CPU @ 2.00GHz with 4 GB RAM, exhibited an accuracy of 98%. A consistent increase in accuracy was observed up to the 7th epoch, maintaining stability between 97.5% and 98% from the 7th to the 10th epoch. This suggests the model's learning progression reached saturation after 7 epochs, with subsequent epochs resulting in marginal

fluctuations. The sustained accuracy after 10 epochs demonstrates the model's effectiveness in distinguishing between tumor-affected and normal brain images.

```
Confusion Matrix:
[[321  22]
 [  4 253]]
Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.94      0.96       343
           1       0.92      0.98      0.95       257

    accuracy                           0.96       600
   macro avg       0.95      0.96      0.96       600
weighted avg       0.96      0.96      0.96       600
```

Figure 3.2.1 Classification Report

4.4 Loss Analysis

The concluding point on the loss versus epoch curve reveals a final loss value of 0.12 (rounded up to 2 decimal places). This signifies the extent of how well the model has minimized its error during the entire training process. The decreasing trend in loss values over epochs indicates successful learning, with the model consistently improving its ability to make accurate predictions. This final loss value serves as a key metric, reflecting the overall effectiveness of the model in capturing patterns and making informed decisions.

Chapter 5

Conclusion and Future Work

5.1 Final Analysis

To boost the accuracy of the brain tumor detection model, exploring a few options can be beneficial. Firstly, examining hyperparameter tuning effects, such as adjusting learning rate, batch size, or CNN layers, may optimize model performance. Conducting a systematic grid search or using automated tools can assist in identifying effective combinations.

Expanding the dataset is another key factor for improved accuracy. A larger and diverse dataset equips the model with a broader range of features, enhancing its ability to generalize to different patterns in brain images. Maintaining a balanced representation of tumor and non-tumor cases helps prevent bias in learning.

5.2 Future Work

Future research could look into using real-time monitoring with dynamic imaging. This might help track how tumors progress and respond to treatment in real-time. Working closely with clinicians and neuroscientists can help us better understand the practical impact of model predictions. This collaboration is essential for creating diagnostic tools that prioritize the needs of patients.

References

[1] Proposed capsule algorithms networks (DCNet) and diverse capsule networks (DCNet++), achieving test accuracies of 93.04% and 95.03%, respectively .

[2]Abiwinanda (2018) used a CNN with the "adam" optimizer on 3064 T-1 weighted CE-MRI images, achieving accuracy rates of 98.51% (training) and 84.19% (validation) .

[3]Mittal et al. combined Stationary Wavelet Transform (SWT) and a Growing CNN (GCNN) for automated brain tumor segmentation..

[4]In another 2018 study, CNNs, particularly AlexNet, were applied for automatic brain tumor diagnosis, achieving 99.55% accuracy.

[5]Rehman et al. explored AlexNet, GoogLeNet, and VGGNet to differentiate meningioma, glioma, and pituitary tumors. VGG16 demonstrated the highest accuracy of 98.69% .

[6]Paul et al. used deep learning to classify brain images related to meningioma, glioma, and pituitary tumors, achieving 91.43% accuracy in a fivefold cross-validation with Python .

[7]In a 2020 study, Badža and Barjaktarovic´ utilized a CNN to classify glioma, meningioma, and pituitary tumors, achieving a best accuracy of 96.56% .

[8]Gumaei et al. proposed an automated approach for brain tumor identification in three steps: brain image preprocessing, feature extraction using PCA-NGIST, and classification using Regularized Extreme Learning Machine (RELM) .

[9]Pashaei et al. developed a CNN model to identify meningioma, glioma, and pituitary tumors, achieving a high accuracy of 93.68% using a tenfold cross-validation on the dataset provided by Cheng.

[10] (BR35H) brain tumor detection dataset 2020 by Ahmedhamada. https://www.kaggle.com/datasets/ahmedhamada0/brain-tumor-detection?resource=download

[11] Various internet sources like google.com , youtube.com, kaggle.com , github.com