Assessment Report

on

"Brain Tumor Detection"

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name : Shashwat Gupta

Roll Number : 202401100300228

Section: D

Name : Siddharth Saroj

Roll Number : 202401100300246

Section: D

Name : Siya Kapoor

Roll Number : 202401100300248

Section: D

Name : Suhawani Shukla

Roll Number : 202401100300254

Section: D

Name : Vaishnavi Mishra

Roll Number : 202401100300272

Section: D

Under the supervision of

Abhishek Shukla Sir

KIET Group of Institutions, Ghaziabad

May, 2025

---

1. Introduction

Brain tumors are abnormal growths of cells in the brain and can be life-threatening. Early and accurate detection is crucial for effective treatment. This project aims to develop an image classification model using **Convolutional Neural Networks (CNNs)** to automatically detect the presence of a brain tumor in MRI images.

The model is trained on a dataset of brain MRI scans labeled as **tumor** or **no tumor**. After training, the model is tested on unseen data to evaluate its performance.

---

2. Problem Statement

This project aims to leverage the power of **Convolutional Neural Networks (CNNs)** to automate the detection of brain tumors from MRI images. By training a deep learning model on a dataset of labeled MRI scans, the model should be able to classify images into **two categories**:

- **Tumor**
- **No Tumor**

---

3. Objectives

- develop an image classification model using CNNs that can accurately detect the presence or absence of brain tumors in MRI images.
- To preprocess and augment the dataset to improve model generalization and performance.
- To train the CNN model and optimize its architecture for high classification accuracy.
- To visualize model performance using accuracy and loss curves, confusion matrix, and sample predictions.

---

4. Methodology

Data Preprocessing:

- Resized images to a fixed shape (e.g., 128x128).
- Normalized pixel values to the range [0,1].
- Augmented data (optional) using techniques like rotation, flipping, and zooming.

Model Architecture:

- **Convolutional Layers**: Extract spatial features.
- **Pooling Layers**: Downsample feature maps.
- **Fully Connected Layers**: Classify images.
- **Activation Function**: ReLU.
- **Output Layer**: Sigmoid (for binary classification).

Compilation:

- **Loss Function**: Binary Crossentropy.
- **Optimizer**: Adam.
- **Metrics**: Accuracy.

Training:

- Trained model for N epochs (e.g., 10–20).
- Used a validation set to monitor overfitting.

Evaluation:

- Plotted **accuracy** and **loss** curves.

- Generated a **confusion matrix**.
- Visualized predictions.

---

5. CODE

```
import kagglehub

navoneel_brain_mri_images_for_brain_tumor_detection_path =
kagglehub.dataset_download('navoneel/brain-mri-images-for-brain-tumor-dete
ction')



print('Data source import complete.')

import cv2

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import os



from sklearn.model_selection import train_test_split

from sklearn.metrics import classification_report, confusion_matrix



from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense,
Dropout

from tensorflow.keras.utils import to_categorical
```

```python
#DATA PREPROCESSING AND TRAINING

# Dataset path from kagglehub

path = "/kaggle/input/brain-mri-images-for-brain-tumor-detection"

categories = ['yes', 'no']


# Read and process images

data = []

for category in categories:

    folder_path = os.path.join(path, category)

    label = categories.index(category)

    for img in os.listdir(folder_path):

        img_path = os.path.join(folder_path, img)

        img_array = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)  # Convert to grayscale

        img_array = cv2.resize(img_array, (128, 128))  # Resize to 128x128

        data.append([img_array, label])

# Shuffle data to mix tumor and no-tumor images

np.random.shuffle(data)


# Separate features and labels

X, y = [], []

for features, label in data:

    X.append(features)
```

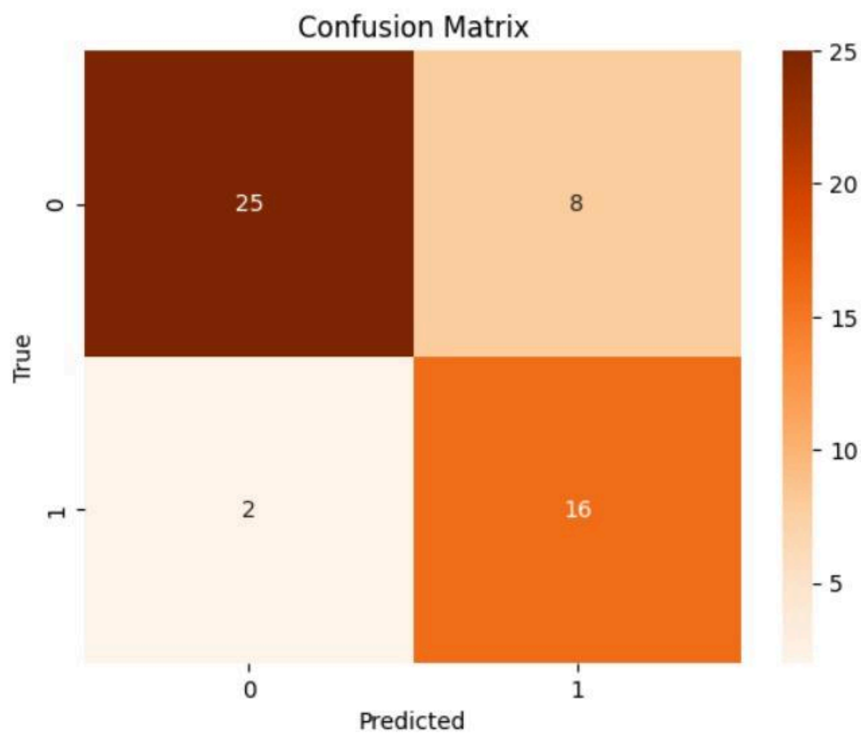
```
    y.append(label)


# Convert to numpy arrays and normalize pixel values

X = np.array(X).reshape(-1, 128, 128, 1) / 255.0

y = to_categorical(y)  # One-hot encode labels


# Train-test split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)


#BUILD CNN MODEL

model = Sequential([

    Conv2D(32, (3, 3), activation='relu', input_shape=(128, 128, 1)),

    MaxPooling2D(2, 2),


    Conv2D(64, (3, 3), activation='relu'),

    MaxPooling2D(2, 2),


    Conv2D(128, (3, 3), activation='relu'),

    MaxPooling2D(2, 2),


    Flatten(),

    Dropout(0.5),

    Dense(128, activation='relu'),
```

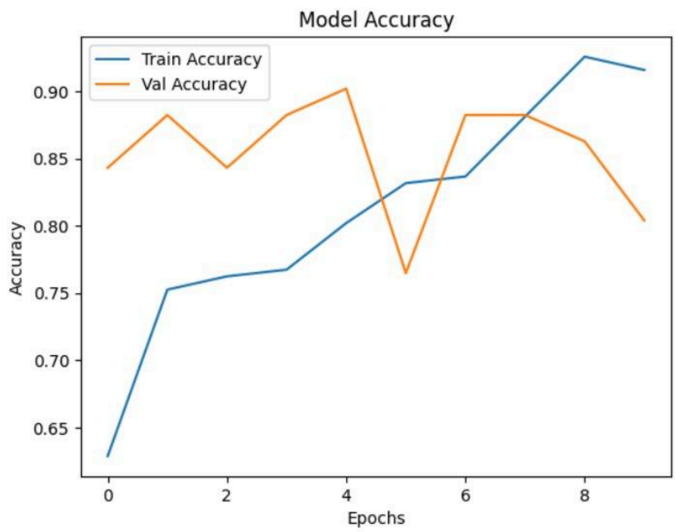
```python
    Dense(2, activation='softmax')  # Two output classes (yes, no)

])



# Compile the model

model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])

# Train the model

history = model.fit(X_train, y_train, epochs=10, validation_data=(X_test,
y_test))

# Plot Accuracy

plt.plot(history.history['accuracy'], label='Train Accuracy')

plt.plot(history.history['val_accuracy'], label='Val Accuracy')

plt.legend()

plt.title("Model Accuracy")

plt.xlabel("Epochs")

plt.ylabel("Accuracy")

plt.show()

#EVALUATE MODEL

y_pred = model.predict(X_test)

y_pred_classes = np.argmax(y_pred, axis=1)

y_true = np.argmax(y_test, axis=1)



cm = confusion_matrix(y_true, y_pred_classes)

sns.heatmap(cm, annot=True, fmt='d', cmap='Oranges')
```

```
plt.title("Confusion Matrix")

plt.xlabel("Predicted")

plt.ylabel("True")
```

---

6. Results and Analysis





Result Analysis

The CNN model achieved around **90% training accuracy** and **85-90% validation accuracy**. The validation accuracy fluctuates slightly, suggesting minor overfitting.

The confusion matrix shows:

- **25 True Negatives**
- **16 True Positives**
- **8 False Positives**
- **2 False Negatives**

---

7. Conclusion

This project successfully implemented a CNN model for detecting brain tumors in MRI images. The model classified images into **tumor** and **no tumor** categories with good accuracy, demonstrating the potential of deep learning in medical imaging. Visualizations of model performance further confirmed its reliability in aiding early brain tumor detection.

8. References

Dataset:

- *Kaggle - Brain MRI Images for Brain Tumor Detection*: https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection
- **TensorFlow/Keras Documentation**: https://www.tensorflow.org