KANTIPUR ENGINEERING COLLEGE

(Affiliated to Tribhuvan University)

Dhapakhel, Lalitpur

[Subject Code: CT755]

A MAJOR PROJECT MID-TERM REPORT ON

BRAIN TUMOR DETECTOR

Submitted by:

Aliz Shrestha	**[KAN076BCT009]**
Apurva Sharma Subedi	**[KAN076BCT016]**
Binit Shakya	**[KAN076BCT022]**
Kashyap Ghimire	**[KAN076BCT037]**

A MAJOR PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR IN COMPUTER ENGINEERING

Submitted to:

Department of Computer and Electronics Engineering

December, 2023

BRAIN TUMOR DETECTOR

Submitted by:

Aliz Shrestha	**[KAN076BCT009]**
Apurva Sharma Subedi	**[KAN076BCT016]**
Binit Shakya	**[KAN076BCT022]**
Kashyap Ghimire	**[KAN076BCT037]**

Supervised by:

Er. Binod Wosti

Computer Engineer

CIAA

A MAJOR PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF BACHELOR IN COMPUTER ENGINEERING

Submitted to:

Department of Computer and Electronics Engineering

Kantipur Engineering College

Dhapakhel, Lalitpur

December, 2023

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CNN:	Convolution Neural Network
SVM:	Support Vector Machine
YOLO:	You Only Look Once
MRI:	Magnetic Resonance Imaging

CHAPTER 1
INTRODUCTION

1.1 Background

In the field of medical imaging, Magnetic Resonance Imaging (MRI) has emerged as a powerful tool for diagnosing and monitoring various medical conditions. MRI scans provide detailed images of internal organs, tissues, and structures within the body, enabling healthcare professionals to detect abnormalities and plan appropriate treatment strategies. However, the interpretation of MRI scans can be complex and can be in large numbers, requiring expertise and experience to accurately analyze the voluminous data generated by these scans[1].

The detection of brain tumors plays a crucial role in early diagnosis and treatment planning. In recent years, deep learning algorithms have shown remarkable performance in medical image analysis[2]. One such algorithm is AlexNet, a convolutional neural network (CNN) that has proven effective in various computer vision tasks. The objective of this project is to leverage the power of AlexNet for brain tumor detection[1]. The proposed method involves training the AlexNet architecture on a large dataset of brain MRI images, consisting of both tumor and non-tumor cases. The network learns to extract distinctive features from the images through multiple convolutional and pooling layers. Once the features are obtained from the AlexNet model, they are fed into an SVM classifier. SVM is a popular machine learning algorithm known for its effectiveness in binary classification tasks[3]. The SVM learns to differentiate between tumor and non-tumor cases based on the extracted features from the AlexNet model.The final classification layer is modified to output a binary prediction indicating the presence or absence of a brain tumor where ‘0’ will show that the test image is tumor free and ‘1’ will show that the test image has a tumor.

The introduction of “Brain Tumor Detector” provides the feature of detection of various kinds of brain tumor using deep learning techniques in order to provide accurate detection and representation of the portion of the brain that has the tumor cell present in it.

1.2 Problem Statement

The development of a computer system capable of automatically and accurately detecting brain tumors in MRI scans is the focus of this project. The conventional manual analysis of medical images for brain tumor detection is time-consuming and prone to human errors. To overcome these limitations, the project aims to leverage the capabilities of the AlexNet convolutional neural network (CNN) algorithm.

By training the AlexNet model on a large dataset of labeled brain MRI images, the model can learn to extract essential features for distinguishing between tumor and non-tumor cases. The complexity of MRI images and the need to minimize false alarms pose challenges that must be addressed.

1.3 Objectives

The main objectives of this application are mentioned below:

i. To accurately detect tumors using deep learning.

1.4 Project Features

The application is targeted toward the general population. So the features of this application can be listed below:

i. Detect tumor based on input MRI scanned images
ii. Provides the interface for loading the MRI images and check the results
iii. Use deep learning techniques for training the data sets for higher accuracy.

1.5 Application Scope

Brain Tumor Detector has a diverse range of applications in medical imaging, diagnosis, treatment planning, patient education, and research. These advanced tools enhance the interpretation of MRI scans by providing accurate prediction, enabling accurate and early diagnoses and improved surgical planning[4]. They also contribute to medical research and education by facilitating detailed anatomical study of possible cases of brain tumors.

1.6 System Requirement

1.6.1 Development Requirements

1.6.1.1 Software Requirements

- Windows/Linux/Mac
- Python IDE
- QT Designer

1.6.1.2 Hardware Requirements

- PC with 16 GB RAM
- Graphics with 4 GB dedicated memory

1.6.2 Deployment Requirements

1.6.2.1 Software Requirements

- Windows 10

1.6.2.2 Hardware Requirements

- CPU with 1.5 GHz clock speed
- 8 GB RAM

1.7 Project Feasibility

1.7.1 Technical Feasibility

Technically, the system is feasible enough for most users, but a basic familiarity with the applications is expected from the users.

1.7.2 Operational Feasibility

The user will not need any formal knowledge about programming so our project is operationally feasible.

1.7.3 Economic Feasibility

The purpose of the economic feasibility assessment is to determine the positive economic benefits to the user that the proposed system will provide. Most of the software

used for the development is free. Thus, the project is economically feasible.

1.7.4 Schedule Feasibility

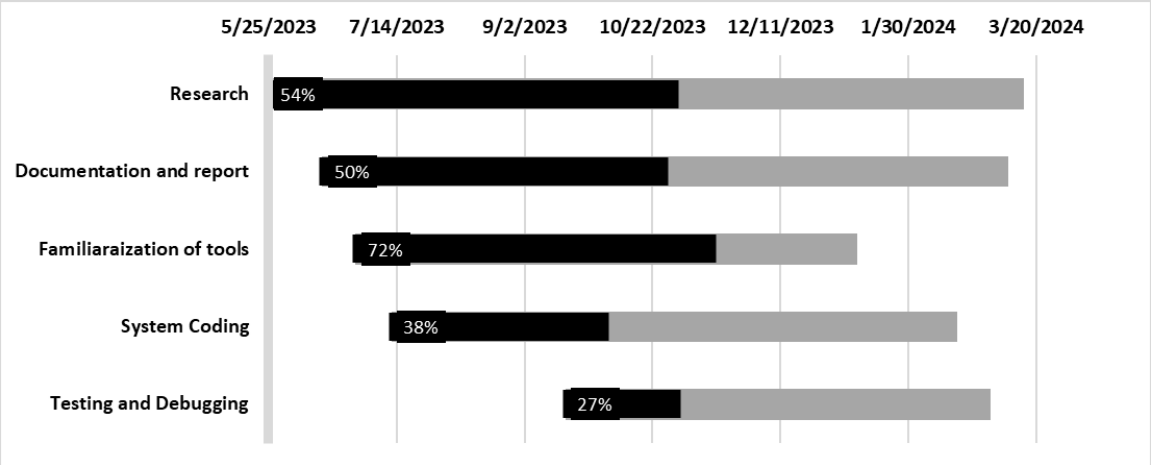

Figure 1.1: Gantt Chart

CHAPTER 2
LITERATURE REVIEW

2.1 Related Projects

2.1.1 Brain Tumor Segmentation (BraTS) Software:

BraTS is a widely used software for brain tumor segmentation, which involves identifying and delineating tumor regions in MRI scans. It utilizes CNN-based algorithms to analyze the image data and generate precise tumor segmentation maps.

2.1.2 MIPAV:

Medical Image Processing, Analysis, and Visualization (MIPAV) is a software package developed by the National Institutes of Health (NIH). It offers a range of tools for tumor detection, including automated segmentation algorithms, image registration, and quantitative analysis.

2.1.3 ImageJ:

ImageJ is an open-source image processing software widely used in medical research and clinical practice. It provides a range of plugins and tools for tumor detection and analysis, including segmentation algorithms and quantitative measurements.

2.2 Related Research

A CNN-SVM based method is proposed to classify brain tumor with higher accuracy. Firstly, a convolutional neural network having 19 layers is constructed using three convolutional 2D layers, three max-pooling layers, two fully-connected layers, three batch normalization layers with activation functions reLu. Secondly softmax is used as a classifier and implemented over a dataset containing 3064 images on three class of tumor images (glioma tumors,meningioma tumors, and pituitary tumors). After that, another classifier named support vector machine is used to improve the accuracy of the CNN model using the features extracted from the model. The final accuracy of this proposed CNN-SVM based method is found 97.1% [3].

For the next research work we looked did a study whose aim was to utilize three well-known CNN-based algorithms, AlexNet, Faster R-CNN, and YOLOv4 to perform multiclass classification of brain MRI scans of Alzheimer Disease patients.

Classifiers	AlexNet	Faster R-CNN	YOLOv4
Loss	0.02	0.3	0.7
Accuracy	99%	41%	86%

[1]

Table 2.1: Comparision table Of CNN architecture

CHAPTER 3
METHODOLOGY

3.1 Working Mechanism

3.1.1 Block Diagram

A block diagram is a drawing illustration of a system whose parts are illustrated by blocks. These blocks are joined by lines to display the relationship between the blocks. Block diagrams are used to visualize the functionality of a system.

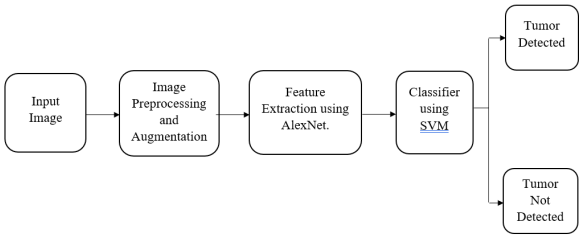

Figure 3.1: System Block Diagram

Input images of MRI scan are provided to the system. Then the data is preprocessed to make it easy for classification purposes. Then the data is augmented after which AlexNet architecture of CNN model extracts the features of the provided image.The output from AlexNet is then fed to the SVM model that has learned from the training dataset, it classifies the images as tumor detected or not.

3.1.2 Input data

Firstly, the dataset that is used to train the model is taken and classified into testing, validation and training datasets. The testing dataset is used in order to train the deep learning algorithm, the validation dataset is then used to validate the model and the test dataset is used to test the model performance metrics such as accuracy, sensitivity and specificity. The training data we will be using has been taken from kaggle and has already been seperated into two category in relation to the presence of tumor.There are 155 jpg image that have tumor and 98 jpg image that are tumor free.

3.1.3 Data preprocessing

After the dataset has been gathered and classified into training and testing sets, they need to be preprocessed. Data preprocessing is an essential step in CNN image classification, aimed at improving the quality and efficiency of the classification process. It involves several techniques to prepare the input data before feeding it into the network.

3.1.4 Data augmentation

Data augmentation is a technique used in deep learning for image classification to create more training examples by making changes to the original images. These changes help improve the model's ability to understand different variations of the same object. The common ways of doing data augmentation are:

1. Flipping

The images are flipped horizontally or vertically to help the model recognize objects from different angles.

2. Rotating

The images are rotated by different angles to help the model handle objects that are not upright.

3. Cropping

Parts of the images are randomly cut out to teach the model to focus on important parts of an object.

4. Scaling

The size of the images is changed to help the model recognize objects at different scales.

5. Adjusting Colors

The colors of the images are changed slightly to help the model handle variations in lighting and color. Random noise is added to the images to make the model more robust to imperfections in real-world images.

By using these techniques, we can create more diverse training data, which helps the

model learn better and perform well on new and unseen images.

3.1.5 Feature Extraction

For feature extraction of the datasets we will be using AlexNet model of CNN. This model of CNN is chosen because of its better performance compared to similar models such as YOLOv4 and R-CNN [1].

AlexNet

AlexNet is a deep convolutional neural network (CNN) that was introduced in 2012 by Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. It gained significant attention and achieved breakthrough performance in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) competition.

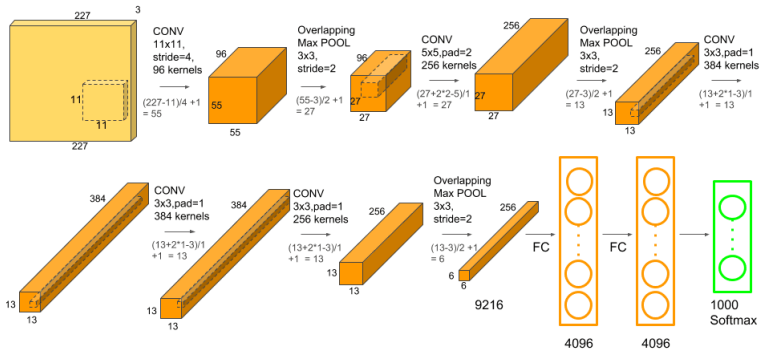

Figure 3.2: Architecture of Alexnet
Source: *https://towardsdatascience.com/transfer-learning-using-pre-trained-alexnet-model-and-fashion-mnist-43898c2966fb*

Here's a step-by-step overview of how AlexNet works up to the feature extraction stage:

1. **Input Image:** The network takes an input image with dimensions typically set to 224x224 pixels.

2. **Convolutional Layers:** The input image passes through a series of convolutional layers. These layers consist of learnable filters that scan the input image, extract

local features, and produce feature maps. AlexNet has a total of five convolutional layers. When we apply a filter of size [f*f] with stride 's' on an padded image of size [n1*n2] we obtain an output of size:

$$\left\lfloor \frac{n_1 - 2p - f}{s} + 1 \right\rfloor * \left\lfloor \frac{n_2 - 2p - f}{s} + 1 \right\rfloor \tag{3.1}$$

3. ReLU Activation: After each convolutional layer, a rectified linear unit (ReLU) activation function is applied element-wise to introduce non-linearity. ReLU sets all negative values in the feature maps to zero while leaving the positive values unchanged.

4. Pooling Layers: The feature maps are then passed through pooling layers to downsample the spatial dimensions. AlexNet uses max pooling, where the maximum value within each pooling window is retained, reducing the spatial size while preserving important features.

5. Local Response Normalization (LRN): LRN is applied after some of the pooling layers in AlexNet. It normalizes the responses across different feature maps to enhance the contrast between local responses. This helps with generalization and provides a form of competition between features.

After several convolutional and pooling layers, the output is flattened into a one-dimensional vector that is fed into the SVM model for classification

3.1.6 Classification

After we have extracted the feature from the given images we then feed the flattened one-dimensional vector into the SVM so that it can be classified.

SVM

SVM operates in the context of a classification problem, where the goal is to assign an input sample to one of the predefined classes

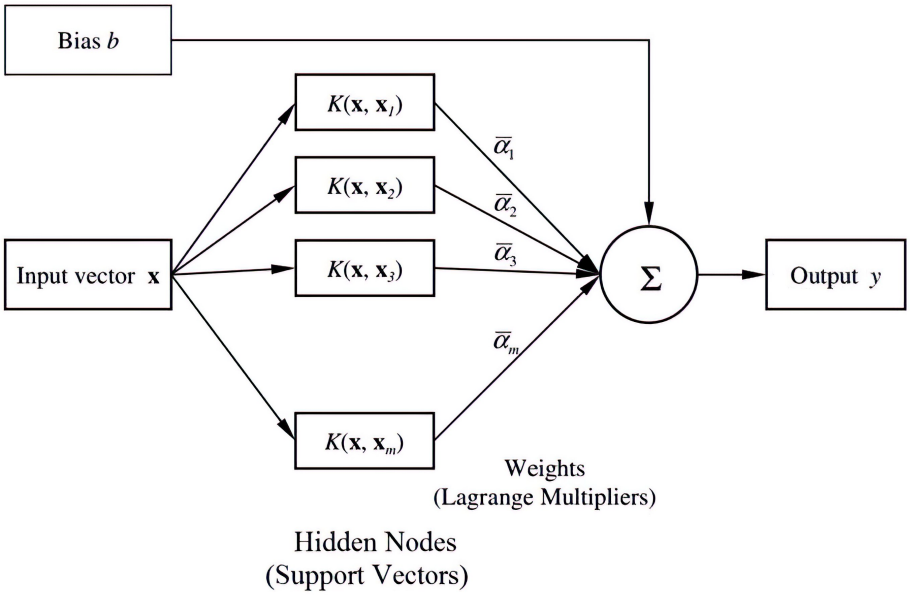

Figure 3.3: Schematic diagram of SVM
Source:*https://www.researchgate.net/figure/Schematic-diagram-of-SVM-architecture*$_{f}ig1_{3}17701295$

1. **Feature Vector:** The feature vector obtained from the fully connected layers serves as the input to the SVM classifier. This feature vector represents the extracted features from the input image.

2. **Training Data:** A labeled training dataset is required to train the SVM. The dataset consists of feature vectors from various samples along with their corresponding class labels.

3. **Feature Scaling:** Before training the SVM, it's common practice to perform feature scaling on the feature vectors. This ensures that all features have similar scales and prevents any particular feature from dominating the learning process.

4. **Support Vectors:** During the training phase, SVM identifies a subset of training samples called support vectors. These support vectors are the samples that lie closest to the decision boundaries between different classes.

5. **Hyperplane:** SVM finds an optimal hyperplane in the high-dimensional feature space that best separates the different classes. This hyperplane maximizes the margin or distance between the closest support vectors from each class.

6. **Kernel Trick: :** SVM can employ a kernel function to transform the feature vectors into a higher-dimensional space, enabling the classification of non-linearly separable data. Common kernel functions include linear, polynomial, radial basis function (RBF), and sigmoid.

7. **Decision Function:** After training, SVM constructs a decision function based on the support vectors and their associated weights. This function takes a new feature vector as input and assigns it to one of the classes based on its position relative to the learned decision boundary.

8. **Prediction:** To predict the class label of a new sample, the SVM classifier applies the decision function to its feature vector. The output of the decision function determines the class assignment.

3.2 System Diagram

3.2.1 Use case diagram

A use case diagram is a way to summarize the details of a system and the users within that system. It is generally shown as a graphic depiction of interactions among different elements in a system. Use Case Diagrams will specify the events in a system and how those events flow. However, Use Case Diagram does not describe how those events are implemented.

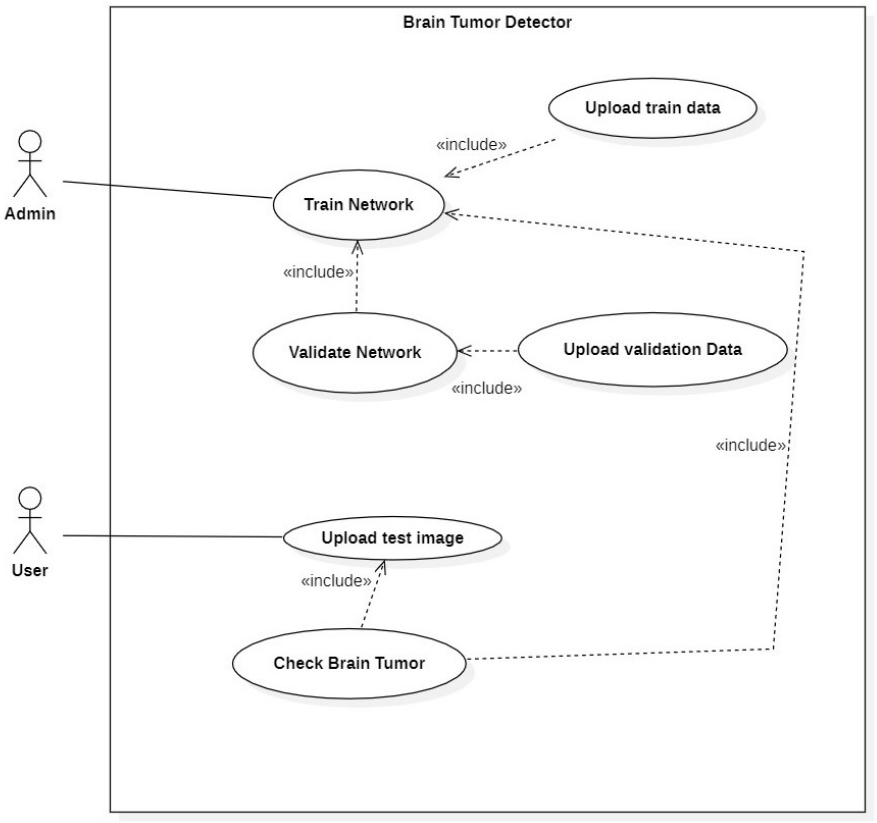

Figure 3.4: Use Case Diagram of Brain Tumor Detector

3.2.2 Software Development Model

One of the software development life cycle models that is easy to apply is the incremental approach. In some cases, the original or fundamental software requirements are well recognized, but the project's actual scope or complete set of features is not. Additionally, the software development company might opt not to provide the entire functionality of the program at once. Instead, they want to distribute it through routine

updates, or if the client demands some functionality upgrades while the project is still being developed Then in these situations, the incremental model is applied.

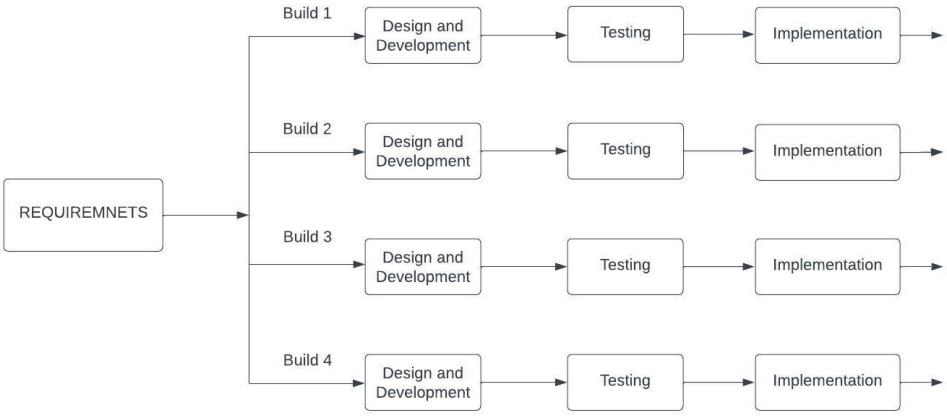

Figure 3.5: Incremental Model

CHAPTER 4
EPILOGUE

4.1 Work Completed

In our recent work, we have incorporated four well-known pre-trained models: Xception, InceptionV3, VGG-19, and ResNetV50 and compared their accuracies to gain insights into their respective strengths and weaknesses. Additionally, we implemented a custom AlexNet variant and evaluated its performance in comparison to the pre-trained models.

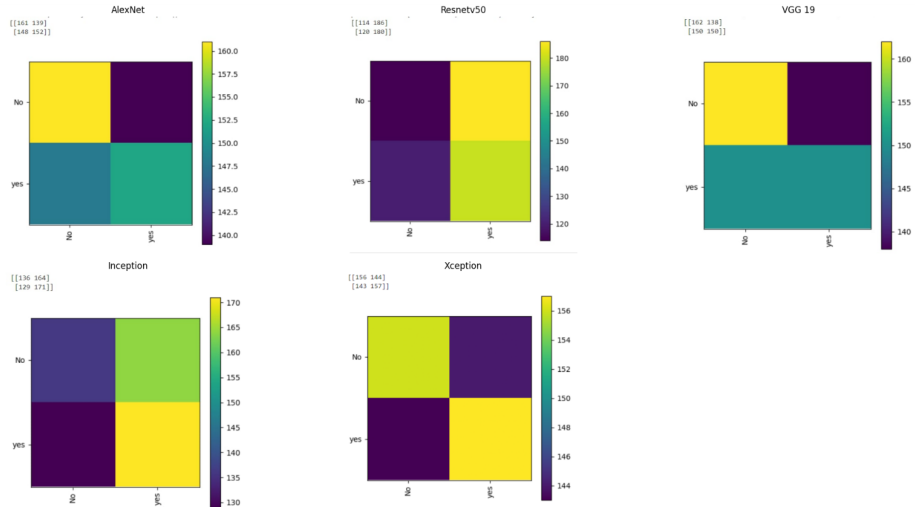

Figure 4.1: Comparision of confusion matrix

```
(1, 227, 227, 3)
[[0. 1.]]
The category =  Tumor
<matplotlib.image.AxesImage at 0x20e13d45dc0>
```

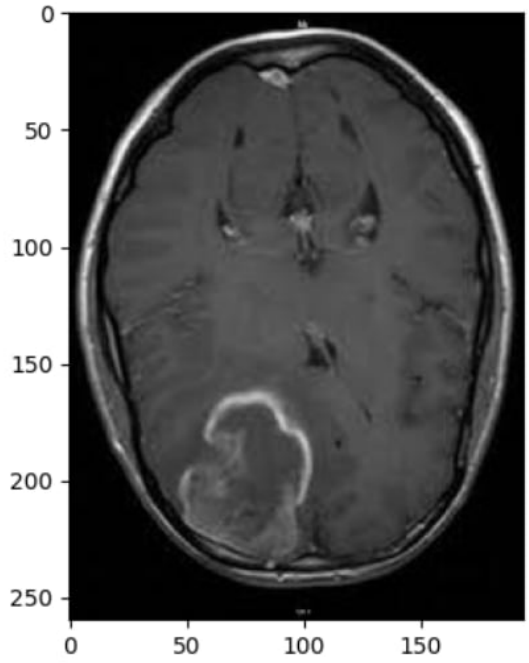

Figure 4.2: Output Image

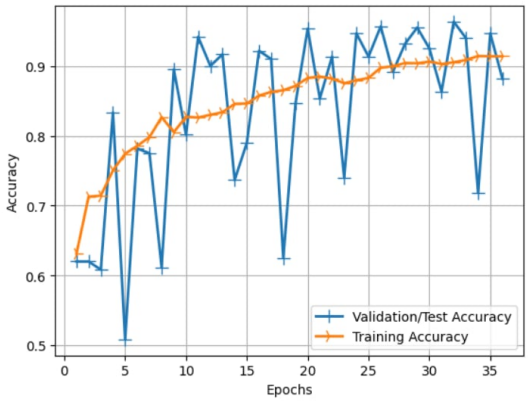

Figure 4.3: Testing and Training Accuracy

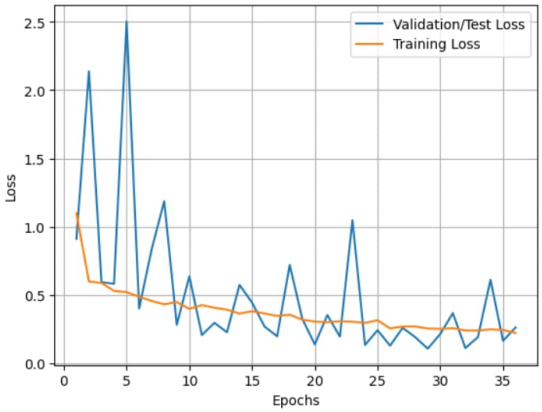

Figure 4.4: Testing and Training Loss

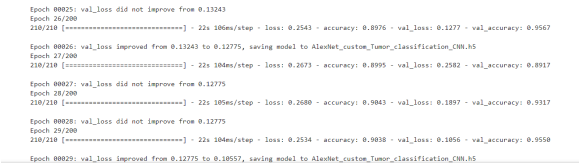

Figure 4.5: Epochs with Early Stopping

4.2 Work Remaining

- Integrating SVM with the model
- Creating UI for the model
- Tuning the hyperparameters

REFERENCES

[1] R. Mirchandani, C. Yoon, S. Prakash, A. Khaire, and A. Naran, "Comparing the Architecture and Performance of AlexNet , Faster R-CNN , and YOLOv4 in the Multiclass Classification of Alzheimer Brain MRI Scans," pp. 1–10.

[2] H. ZainEldin, S. A. Gamel, E. S. M. El-Kenawy, A. H. Alharbi, D. S. Khafaga, A. Ibrahim, and F. M. Talaat, "Brain Tumor Detection and Classification Using Deep Learning and Sine-Cosine Fitness Grey Wolf Optimization," *Bioengineering*, vol. 10, no. 1, pp. 1–19, 2023.

[3] Z. A. Sejuti and M. S. Islam, "An Efficient Method to Classify Brain Tumor using CNN and SVM," *International Conference on Robotics, Electrical and Signal Processing Techniques*, pp. 644–648, 2021.

[4] S. Saeedi, S. Rezayi, H. Keshavarz, and S. R. Niakan Kalhori, "MRI-based brain tumor detection using convolutional deep learning methods and chosen machine learning techniques," *BMC Medical Informatics and Decision Making*, vol. 23, no. 1, pp. 1–17, 2023. [Online]. Available: https://doi.org/10.1186/s12911-023-02114-6