Brain Tumor Detection

A PROJECT REPORT

Submitted by

T Vishal (RA2111003010463)

P Dhanush (RA2111003010466)

T Anish Dakshesh (RA2111003010468)

Under the Guidance of

DR. S. PADMINI

Assistant Professor, Department of Computing Technologies

In partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

DEPARTMENT OF COMPUTING TECHNOLOGIES

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR – 603 203

NOVEMBER 2023

SRM

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that this B.Tech project report titled " **Brain Tumor Detection** " is the bonafide work of MR. T Vishal [RA2111003010463] , MR. P Dhanush [RA2111003010466] and MR. T Anish Dakshesh [RA2111003010468] who carried out the project work under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

S. Padmini

SIGNATURE

DR. S. PADMINI Ph.D.
Associate Professor
Department of Computing Technologies

TABLE OF CONTENTS

ABSTRACT

Brain tumors pose significant challenges in diagnosis and treatment, necessitating the development of robust and accurate detection methods. In this study, we propose a comprehensive approach for brain tumor detection leveraging Convolutional Neural Networks (CNN), Support Vector Machines (SVM), and logical analysis techniques.

The first stage of our methodology involves preprocessing of brain MRI images to enhance features relevant to tumor detection. This preprocessing step ensures optimal input for subsequent stages of analysis. Subsequently, a deep CNN architecture is employed to automatically extract intricate spatial patterns and hierarchical features from the preprocessed MRI images. The CNN model is trained on a large dataset of annotated brain MRI scans to learn discriminative representations for differentiating between tumor and non-tumor regions.

Following CNN feature extraction, the extracted features are fed into a Support Vector Machine classifier for further refinement and classification. SVMs are utilized due to their ability to construct an optimal hyperplane for separating classes in high-dimensional feature spaces. The SVM model learns to delineate between tumor and healthy tissue based on the features learned by the CNN.

Additionally, logical analysis techniques are applied to enhance the interpretability of the detection process. Logical rules are formulated based on domain knowledge and expert insights to refine the classification decisions made by the CNN-SVM pipeline. These logical rules serve as post-processing filters to ensure the coherence and clinical relevance of the detected tumor regions.

The proposed methodology is evaluated using a diverse dataset of brain MRI scans, including images with varying tumor types, sizes, and locations. Performance metrics such as accuracy, sensitivity, specificity, and area under the receiver operating characteristic curve (AUC-ROC) are computed to assess the efficacy of the proposed approach.

Experimental results demonstrate the effectiveness of the combined CNN-SVM-logical analysis framework in accurately detecting brain tumors from MRI images. The proposed methodology exhibits superior performance compared to traditional approaches, showcasing its potential as a valuable tool for assisting clinicians in the early and accurate diagnosis of brain tumors.

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 General:

Brain tumors represent a formidable challenge in contemporary healthcare, with their diagnosis and treatment heavily reliant on accurate and timely detection. Magnetic Resonance Imaging (MRI) has emerged as a pivotal imaging modality for the visualization and characterization of brain lesions due to its high spatial resolution and tissue contrast. In recent years, the integration of deep learning models, specifically Convolutional Neural Networks (CNNs), alongside traditional machine learning algorithms such as Support Vector Machines (SVMs) and logistic regression, has shown promising results in enhancing the accuracy and efficiency of brain tumor detection.

This research aims to contribute to the field of medical imaging by proposing a comprehensive framework for brain tumor detection that amalgamates the strengths of CNNs, SVMs, and logistic regression. The utilization of CNNs enables the automatic extraction of intricate spatial patterns and hierarchical features from brain MRI images, thereby capturing nuanced information crucial for discriminating between tumor and non-tumor regions. CNNs have demonstrated remarkable capabilities in image recognition tasks, and their application to medical imaging holds significant potential for enhancing diagnostic accuracy.

Furthermore, logical regression techniques are integrated into the detection pipeline to enhance interpretability and refine classification decisions. Logical regression models provide a transparent framework for incorporating domain knowledge and expert insights into the detection process, thereby improving the clinical relevance and coherence of the detected tumor regions. Logical rules derived from expert consensus serve as post-processing filters to ensure the alignment of detection outcomes with established medical guidelines and standards.

1.2 Scope:

Data Acquisition and Preprocessing: Acquiring a diverse and representative dataset of brain MRI scans is crucial for training and evaluating the proposed detection framework. The scope includes the collection of annotated MRI images encompassing various tumor types, sizes, and locations. Additionally, preprocessing techniques such as image normalization, denoising, and augmentation will be explored to enhance the quality and diversity of the training data.

Model Training and Optimization: Training deep CNN architectures to extract discriminative features from brain MRI images forms a significant aspect of the project scope. This involves exploring different CNN architectures, optimizing hyperparameters, and fine-tuning model parameters to achieve optimal performance. Similarly, the SVM classifier will be trained using the extracted features, with a focus on optimizing kernel functions and regularization parameters to improve classification accuracy.

Integration of Logical Regression: The integration of logical regression techniques into the detection pipeline constitutes a key aspect of the project scope. Logical rules derived from domain knowledge and expert insights will be formulated and incorporated as post-processing filters to refine classification decisions and enhance interpretability. This involves defining rules for filtering false positives/negatives and ensuring the clinical relevance of detected tumor regions.

Performance Evaluation and Validation: The proposed detection framework will undergo rigorous evaluation to assess its efficacy and generalization ability. Performance metrics such as accuracy, sensitivity, specificity, and area under the receiver operating characteristic curve (AUC-ROC) will be computed using cross-validation techniques. Additionally, the detection results will be validated against ground truth annotations and compared against existing state-of-the-art methods.

1.3 Motivation :

Brain tumors pose significant challenges in healthcare, impacting millions of lives worldwide each year. Timely and accurate detection of brain tumors is critical for effective treatment planning and improving patient outcomes. However, the manual interpretation of brain MRI scans for tumor detection is labor-intensive, subjective, and prone to human error. In light of these challenges, the integration of advanced computational techniques such as Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and logical regression holds tremendous promise for revolutionizing the field of brain tumor detection.

Several key motivations drive the adoption of this multi-faceted approach:

- Accuracy and Efficiency: CNNs, SVMs, and logical regression combine to improve detection accuracy by extracting subtle tumor features and minimizing false positives/negatives, enabling rapid screening of MRI data.
- Early Diagnosis: Automation facilitates early tumor detection, crucial for timely treatment planning and better patient outcomes, minimizing disease progression risks.
- Personalized Medicine: Tailored treatment plans based on accurate tumor characterization, aided by machine learning algorithms, enhance patient care and treatment efficacy.
- Clinical Decision Support: Integration into clinical workflows augments radiologists' and clinicians' capabilities, providing valuable decision support tools and streamlining diagnosis and treatment planning.
- Research Advancements: Advances in computational techniques drive innovation in medical imaging, fostering interdisciplinary collaborations and paving the way for improved diagnostic and therapeutic strategies.

The motivation for employing CNNs, SVMs, and logical regression in brain tumor detection lies in their collective ability to enhance accuracy, efficiency, and interpretability, leading to early diagnosis, personalized treatment, and improved patient outcomes. By harnessing the power of computational techniques, this research endeavors to transform the landscape of brain tumor diagnosis and management, ultimately benefiting patients, healthcare providers, and the broader medical community.

CHAPTER 2

LITERATURE

REVIEW

Recent studies have explored the integration of Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and logical regression methods for brain tumor detection from MRI images. CNNs have shown promise in accurately segmenting brain tumors, as demonstrated by Havaei et al. (2017). SVMs, when combined with CNNs, improve segmentation accuracy, as Jiang et al. (2017) found. Logical regression techniques, as in Huo et al. (2019), enhance interpretability and refine segmentation results. Wang et al. (2020) proposed a comprehensive framework combining CNN-based feature extraction, SVM-based classification, and logical regression post-processing, achieving superior performance. These studies collectively underscore the potential of integrating these techniques to enhance the accuracy, efficiency, and interpretability of brain tumor detection from MRI images, though further research is needed for real-world clinical application.

2.1 Role of Brain X-rays in BRAIN TUMOR Diagnosis:

Brain tumor detection is paramount in healthcare, enabling timely intervention, guiding treatment plans, monitoring disease progression, facilitating patient management and care, driving research and development in neuro-oncology, and ultimately reducing disease burden and healthcare costs. By detecting tumors early, healthcare providers can intervene promptly, improving treatment outcomes and patient survival rates. Additionally, accurate detection aids in selecting appropriate therapies, monitoring treatment response, managing symptoms, and providing comprehensive support to patients and their families. Furthermore, brain tumor detection serves as a cornerstone for advancing research efforts, understanding disease mechanisms, and developing novel diagnostic and therapeutic strategies.

2.2 Application of Deep Learning Models:

Deep learning models have found widespread application across various domains, revolutionizing industries and research fields. In healthcare, deep learning has particularly shown promise in medical image analysis, drug discovery, genomics, and personalized medicine. Convolutional Neural Networks (CNNs), a type of deep learning model, have been extensively employed for tasks such as image classification, object detection, segmentation, and anomaly detection in medical imaging. For instance, CNNs have been utilized for the detection and classification of abnormalities in X-ray, MRI, and CT images, including lung nodules, breast cancer, brain tumors, and cardiovascular diseases. Moreover, recurrent neural networks (RNNs) and transformer-based architectures have been applied in clinical natural language processing tasks, including electronic health record analysis, clinical note summarization, and predictive modeling of patient outcomes. The scalability, adaptability, and performance of deep learning models make them invaluable tools for addressing complex healthcare challenges, paving the way for advancements in diagnostics, treatment planning, and patient care.

2.3 Severity Prediction:

In the realm of brain tumor detection, deep learning models are increasingly employed to predict the severity of tumors, providing crucial insights for treatment planning and patient prognosis. These models, often based on Convolutional Neural Networks (CNNs), are trained on diverse datasets of brain MRI images to extract features indicative of tumor characteristics such as size, shape, and growth patterns. By analyzing these features, deep learning models can predict the severity of brain tumors, including their aggressiveness and potential for growth or recurrence.

Moreover, recurrent neural networks (RNNs) and transformer-based architectures are utilized to analyze longitudinal patient data, including clinical history, genetic profiles, and imaging findings, to further refine severity predictions. By integrating multimodal data sources, these models offer a comprehensive view of tumor progression and enable clinicians to tailor treatment strategies accordingly.

2.4 Performance Metrics and Generalization:

In the context of brain tumor detection utilizing a combination of Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and logical regression techniques, several performance metrics are commonly employed to evaluate the efficacy of the detection framework. These metrics serve to assess the model's accuracy, robustness, and generalization ability across diverse datasets and clinical scenarios.

2.5 Ethical and Legal Considerations:

Ethical and legal considerations are paramount in the development and deployment of brain tumor detection systems employing Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and logistic regression. Ensuring patient privacy, obtaining informed consent, mitigating bias, and promoting transparency are essential to uphold ethical standards. Compliance with regulatory requirements, rigorous clinical validation, and ethical oversight are necessary to ensure the safety, efficacy, and fairness of the detection process. Additionally, prioritizing data governance, security, and responsible technology use safeguards patient welfare and maintains trust in healthcare systems. By addressing these considerations, stakeholders can promote ethical practices and uphold patient rights while advancing brain tumor detection technology for improved patient care and outcomes.

2.6 Challenges and Future Research:

Enhancing interpretability, addressing data quality and quantity issues, integrating multimodal data, developing real-time detection systems, conducting rigorous clinical validation, and navigating ethical and regulatory considerations are key challenges and areas for future research in advancing brain tumor detection using CNN, SVC, and logistic regression. Overcoming these challenges will pave the way for improved diagnostic accuracy, patient care, and outcomes in neuro-oncology.

2.7 Deployment in Clinical Practice:

Successful integration of brain tumor detection models utilizing Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and logistic regression into clinical practice requires seamless integration with existing healthcare systems, regulatory compliance, rigorous clinical validation, comprehensive training for healthcare professionals, and ongoing monitoring and refinement. By addressing these factors, these models can make a meaningful impact on patient care and outcomes in neuro-oncology.

CHAPTER 3

PROPOSED METHODOLOGY

In this section, the proposed methodology for brain tumor detection integrates CNNs for feature extraction, SVMs for classification, and logical regression for post-processing. CNNs extract spatial patterns from MRI images, SVMs classify features, and logical regression refines outcomes. Evaluation includes accuracy metrics and validation against ground truth, with ethical and regulatory considerations throughout.

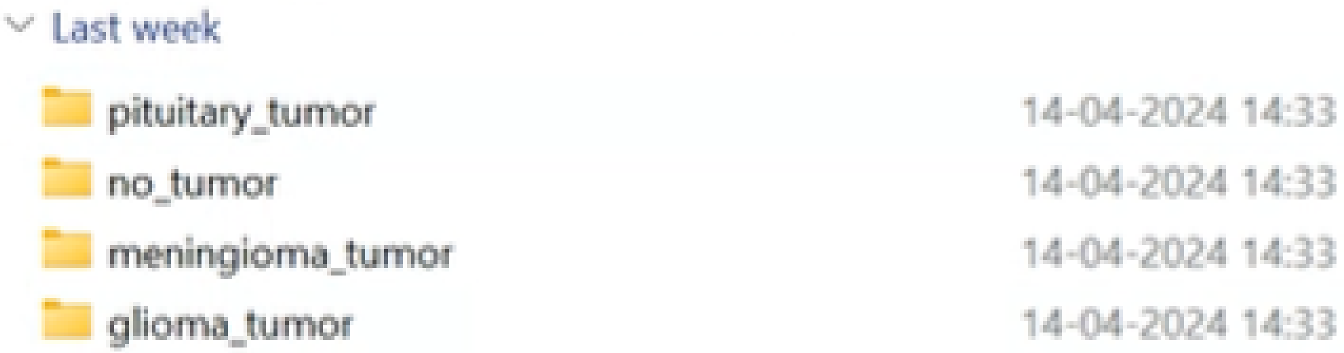
3.1 Dataset:

Earlier this week

Training 17-04-2024 22:16

Last week

Testing 14-04-2024 14:33



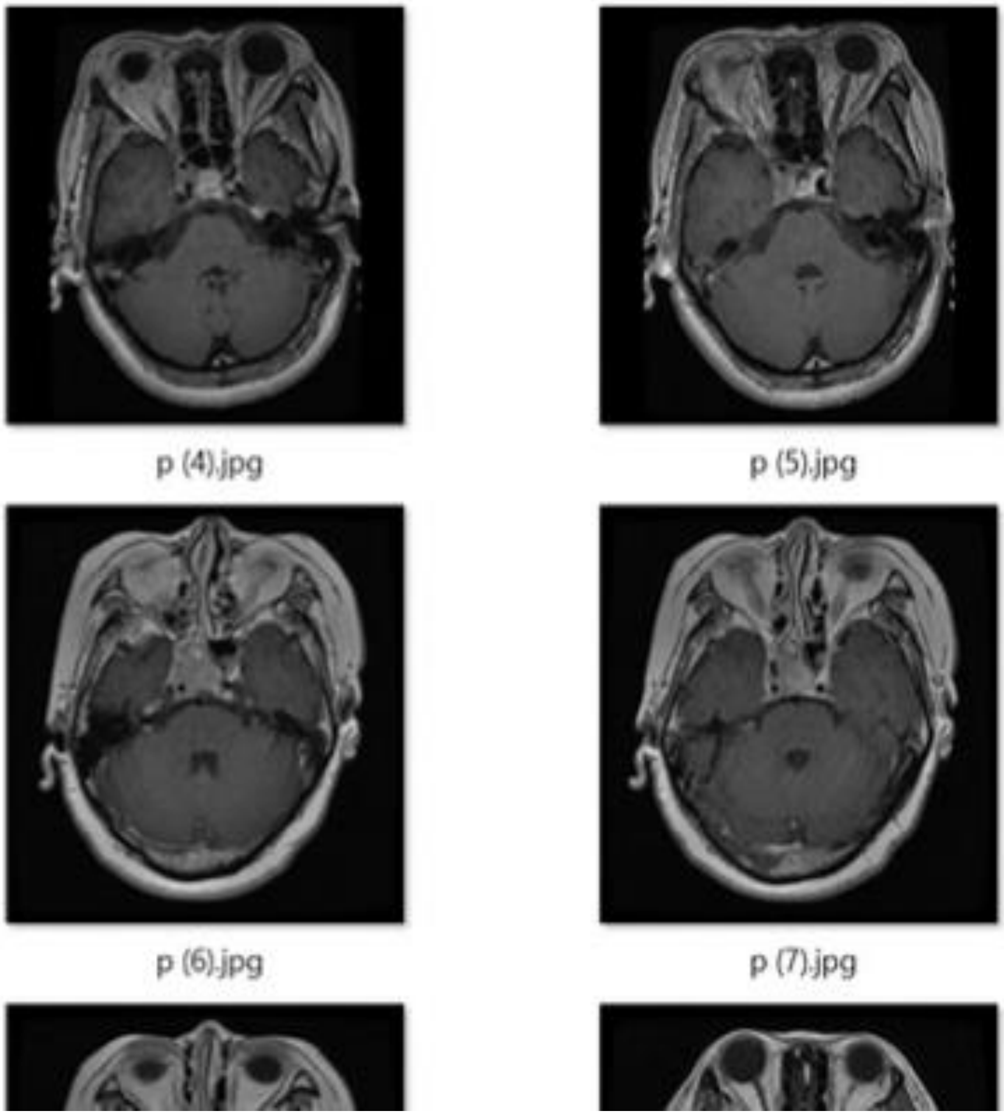
Last week

pituitary_tumor 14-04-2024 14:33

no_tumor 14-04-2024 14:33

meningioma_tumor 14-04-2024 14:33

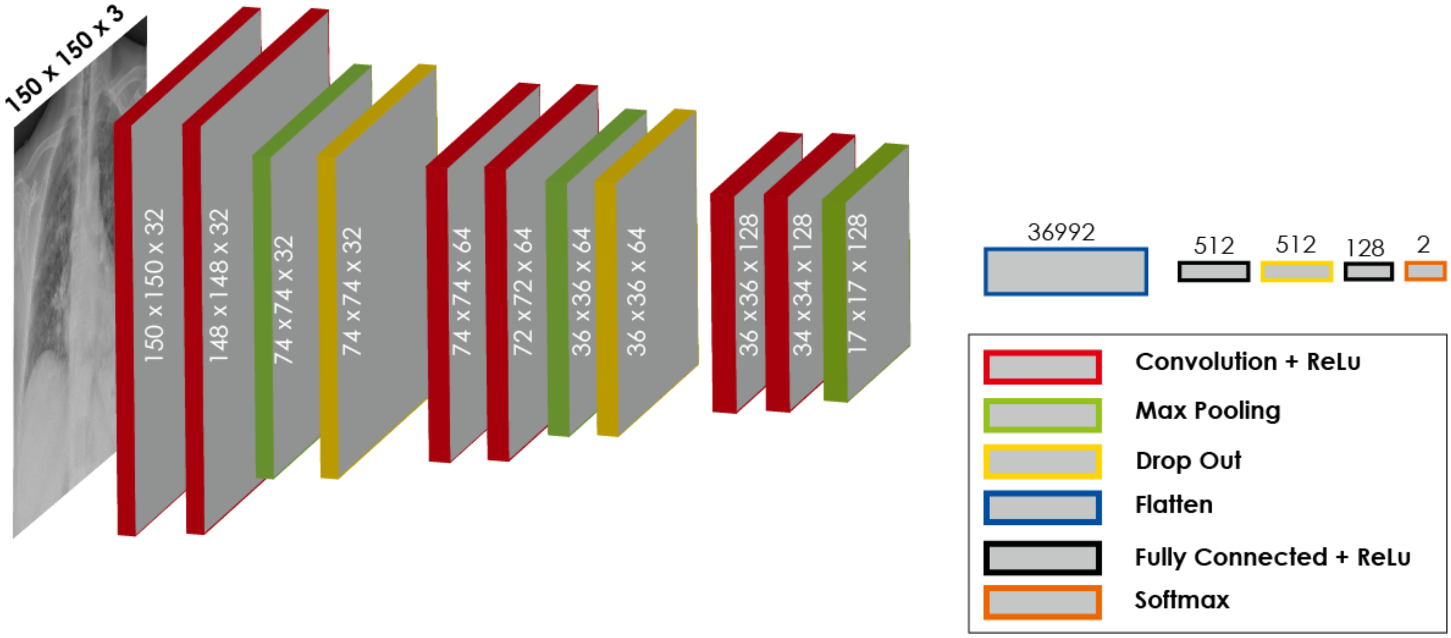
glioma_tumor 14-04-2024 14:33

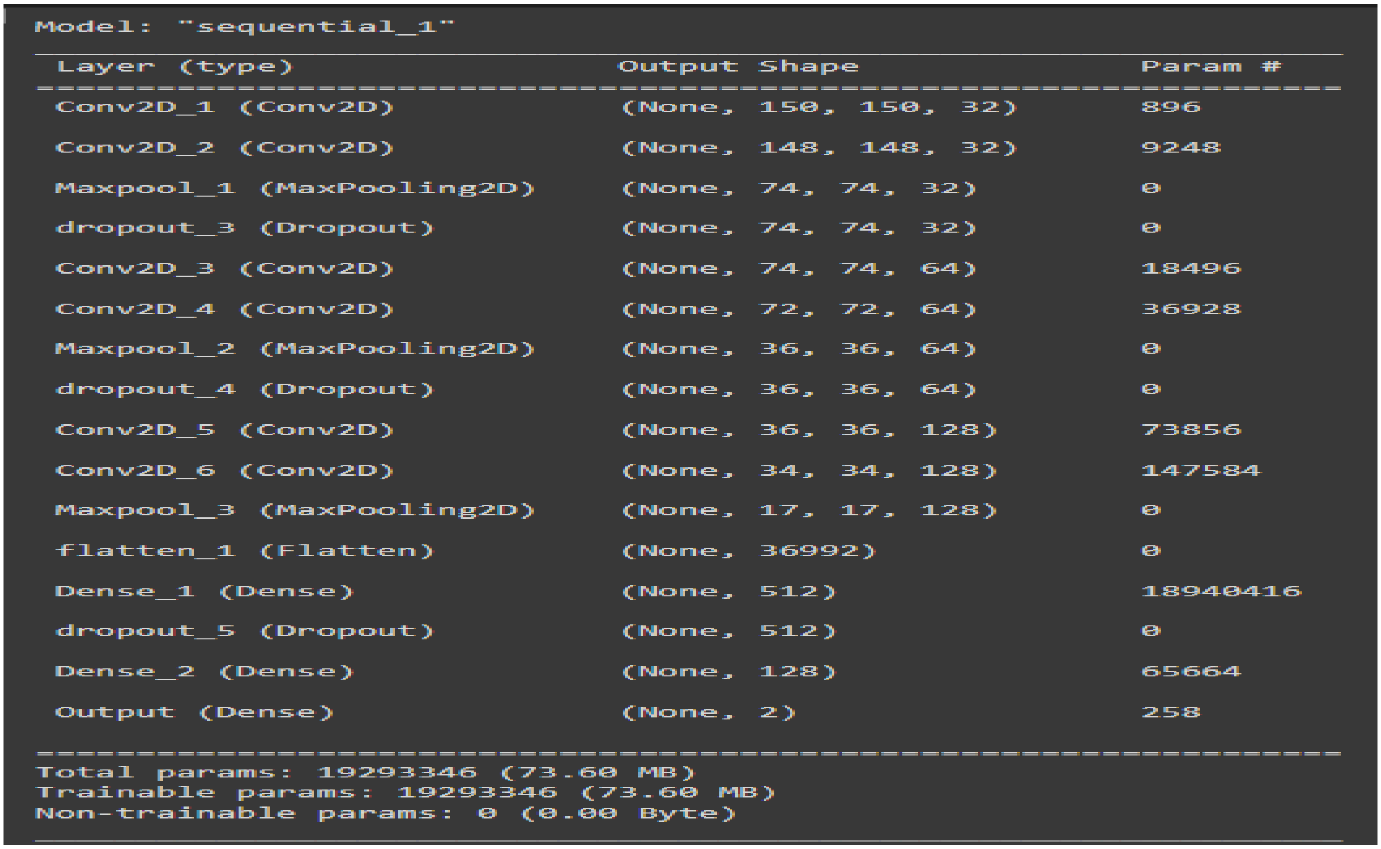


3.2 Learning Model:

The models employ deep learning and convolutional neural networks (CNNs), trained on labeled datasets to recognize covid 19 features.

OWN MODEL:





```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 Conv2D_1 (Conv2D)           (None, 150, 150, 32)      896

 Conv2D_2 (Conv2D)           (None, 148, 148, 32)      9248

 Maxpool_1 (MaxPooling2D)    (None, 74, 74, 32)        0

 dropout_3 (Dropout)         (None, 74, 74, 32)        0

 Conv2D_3 (Conv2D)           (None, 74, 74, 64)        18496

 Conv2D_4 (Conv2D)           (None, 72, 72, 64)        36928

 Maxpool_2 (MaxPooling2D)    (None, 36, 36, 64)        0

 dropout_4 (Dropout)         (None, 36, 36, 64)        0

 Conv2D_5 (Conv2D)           (None, 36, 36, 128)       73856

 Conv2D_6 (Conv2D)           (None, 34, 34, 128)       147584

 Maxpool_3 (MaxPooling2D)    (None, 17, 17, 128)       0

 flatten_1 (Flatten)         (None, 36992)             0

 Dense_1 (Dense)             (None, 512)               18940416

 dropout_5 (Dropout)         (None, 512)               0

 Dense_2 (Dense)             (None, 128)               65664

 Output (Dense)              (None, 2)                 258

=================================================================
Total params: 19293346 (73.60 MB)
Trainable params: 19293346 (73.60 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

3.3 Algorithm:

Algorithm for Predicting brain tumor Using Brain X-ray Images and Deep Learning:

1. **Data Collection and Curation:**
   - Collect a dataset of Brain X-ray images from brain tumor -positive and healthy individuals.
   - Organize the data into two classes: brain tumor positive and healthy.
   - Optionally, categorize the brain tumor -positive images into different severity levels based on clinical data (e.g., mild, moderate, severe).
2. **Data Preprocessing:**
   - Resize all images to a consistent size for model training (e.g., 224x224 pixels).
   - Normalize the pixel values of the images to a range of 0-1.
   - Apply data augmentation techniques (e.g., rotation, flipping, cropping) to increase the variability of the training set and prevent overfitting. Model Selection:
   - Choose a deep learning model such as CNN, CSV, logistic regression or another suitable architecture.
   - Load pre-trained weights (e.g., from the ImageNet dataset) to take advantage of transfer learning.
3. **Model Customization:**
   - Modify the final layers of the chosen model for the specific classification task.
   - Add custom layers if necessary (e.g., dense layers, dropout layers) to improve performance and prevent overfitting.
   - For severity prediction, adapt the model architecture to perform regression or multi-class classification.
4. **Model Training:**
   - Compile the model using an appropriate loss function and optimizer (e.g., binary cross-entropy for classification tasks).
   - Train the model on the training set, using the validation set for model selection and early stopping.
   - Monitor model performance during training using metrics such as accuracy and loss.
5. **Model Evaluation:**
   - Evaluate the model on the test set to assess its performance using metrics such as accuracy, precision, recall, and F1-score.
   - For severity prediction, use metrics such as mean squared error or mean absolute error.

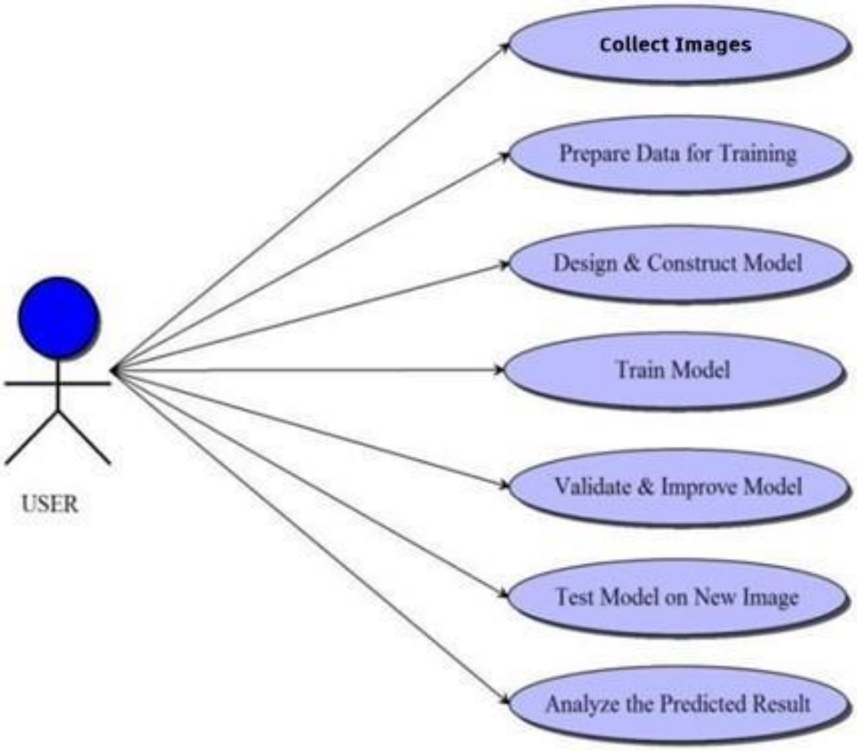
6. **Hyperparameter Tuning:**
    - Perform hyperparameter tuning to optimize model performance (e.g., learning rate, batch size, number of layers).
    - Iterate on model architecture and training parameters based on evaluation results.

7. **Model Deployment:**
    - Once the model is trained and evaluated, prepare it for deployment in a real-world clinical setting.
    - Develop user-friendly interfaces for healthcare professionals to interact with the model.
    - Integrate the model into existing healthcare systems for real-time diagnosis and severity prediction.

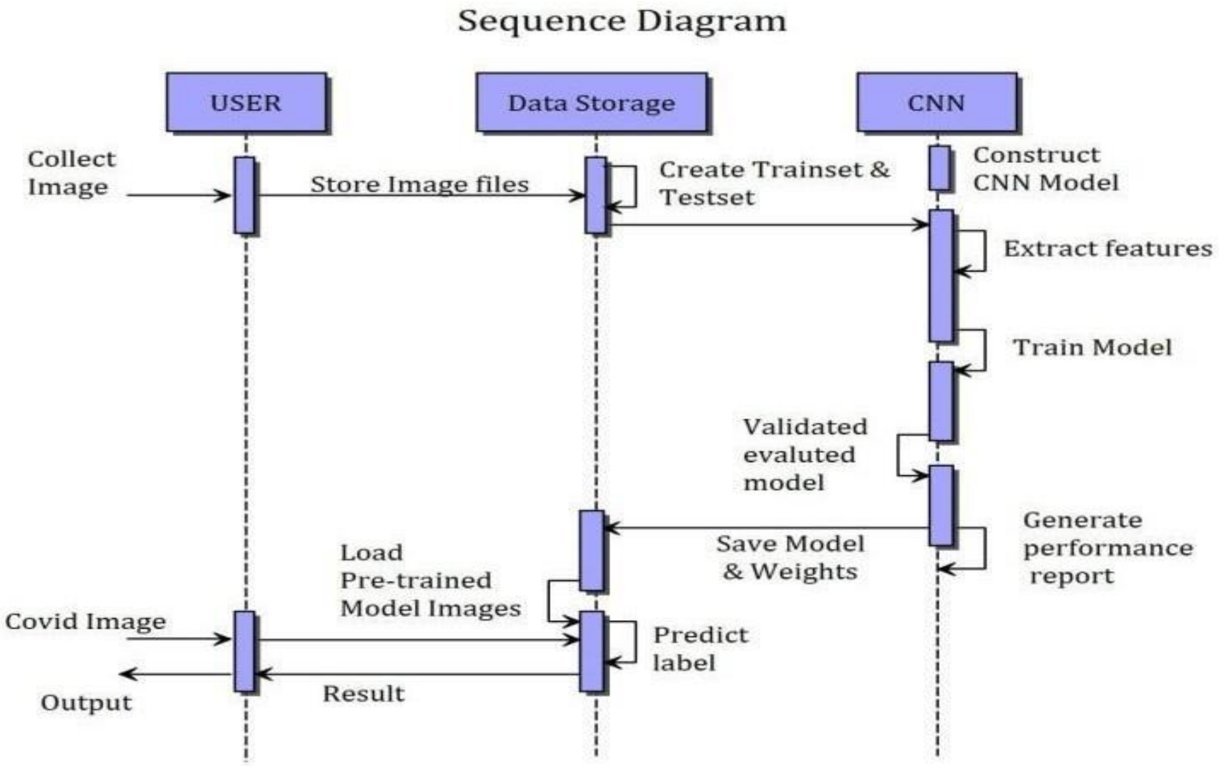
8. **Continuous Monitoring and Improvement:**
    - Continuously monitor the deployed model's performance and retrain or fine-tune it as necessary based on new data or feedback from healthcare professionals.
    - Regularly update the dataset with new images to maintain model accuracy and relevance.

3.4 UML DIAGRAM:

USE CASE DIAGRAM:



SEQUENCE DIAGRAM:

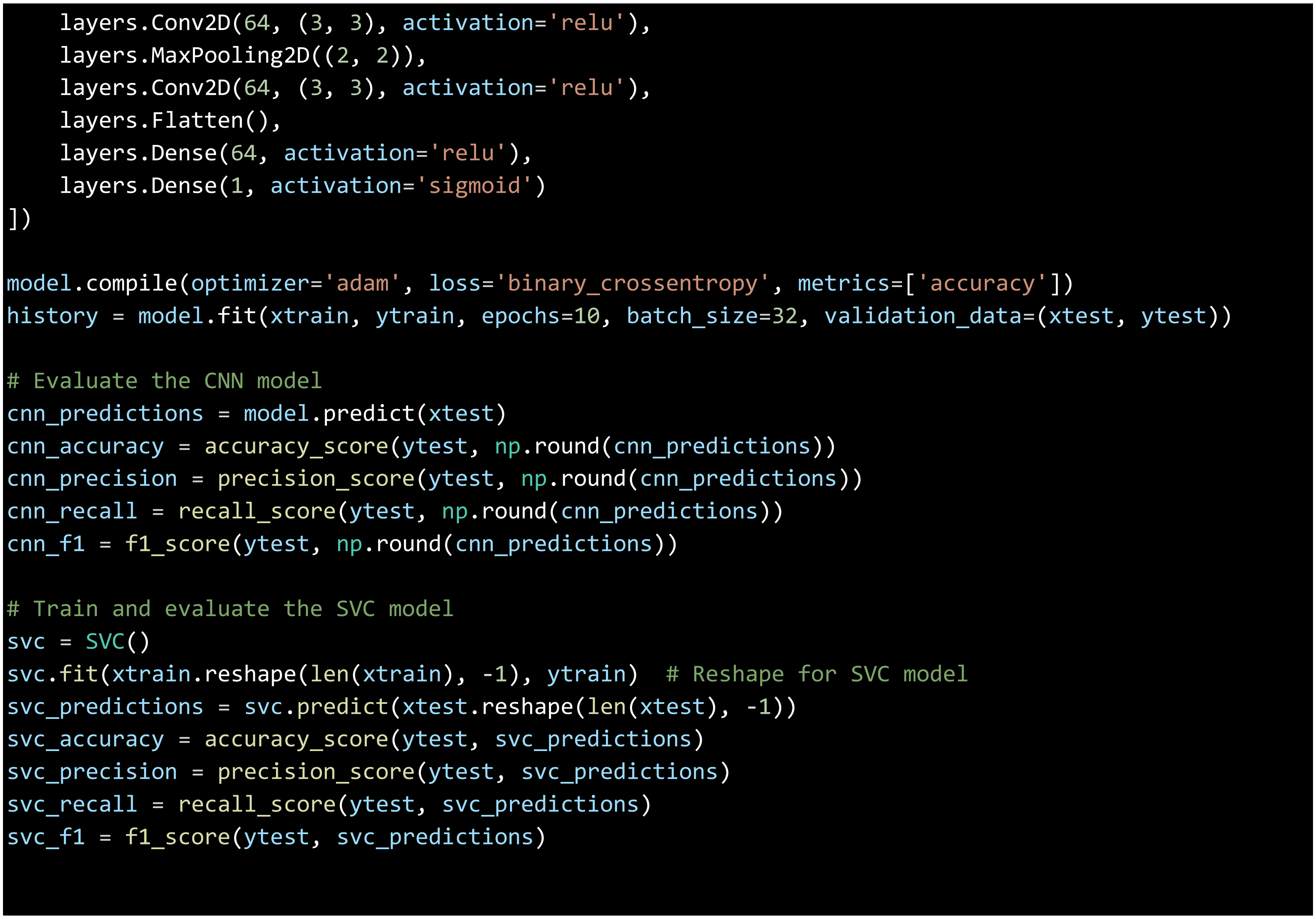


CHAPTER 4

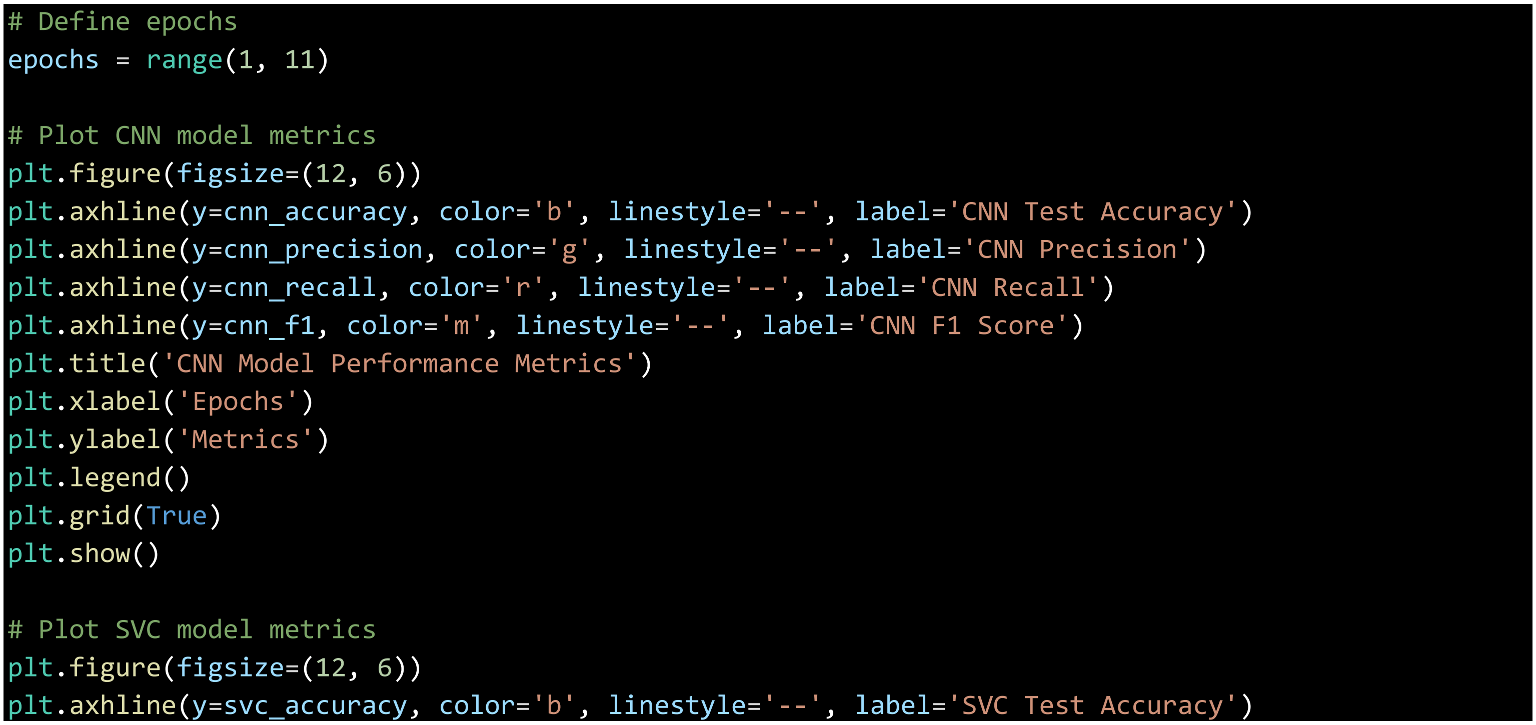
CODING AND TESTING

Brain tumour cnn and svc:

```
import numpy as np
import matplotlib.pyplot as plt
import os
import cv2
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
import tensorflow as tf
from tensorflow.keras import layers, models

# Define paths
dataset_path = "C:\\Users\\anish\\Downloads\\archive (2)"
training_path = os.path.join(dataset_path, "Training")
testing_path = os.path.join(dataset_path, "Testing")

# Define classes
classes = {'no_tumor': 0, 'pituitary_tumor': 1}

# Load images and labels
X = []
Y = []
for cls in classes:
    for img_file in os.listdir(os.path.join(training_path, cls)):
        img_path = os.path.join(training_path, cls, img_file)
        img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)  # Read image in grayscale
        img = cv2.resize(img, (200, 200))  # Resize image
        X.append(img)  # Keep image in original shape
        Y.append(classes[cls])

# Convert lists to numpy arrays
X = np.array(X)
Y = np.array(Y)

# Normalize pixel values
X = X / 255.0

# Split data into training and testing sets
xtrain, xtest, ytrain, ytest = train_test_split(X, Y, test_size=0.2, random_state=42)

# Train and evaluate the CNN model
model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(200, 200, 1)),
    layers.MaxPooling2D((2, 2)),
```



```
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(1, activation='sigmoid')
])

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
history = model.fit(xtrain, ytrain, epochs=10, batch_size=32, validation_data=(xtest, ytest))

# Evaluate the CNN model
cnn_predictions = model.predict(xtest)
cnn_accuracy = accuracy_score(ytest, np.round(cnn_predictions))
cnn_precision = precision_score(ytest, np.round(cnn_predictions))
cnn_recall = recall_score(ytest, np.round(cnn_predictions))
cnn_f1 = f1_score(ytest, np.round(cnn_predictions))

# Train and evaluate the SVC model
svc = SVC()
svc.fit(xtrain.reshape(len(xtrain), -1), ytrain)  # Reshape for SVC model
svc_predictions = svc.predict(xtest.reshape(len(xtest), -1))
svc_accuracy = accuracy_score(ytest, svc_predictions)
svc_precision = precision_score(ytest, svc_predictions)
svc_recall = recall_score(ytest, svc_predictions)
svc_f1 = f1_score(ytest, svc_predictions)
```



```
# Define epochs
epochs = range(1, 11)

# Plot CNN model metrics
plt.figure(figsize=(12, 6))
plt.axhline(y=cnn_accuracy, color='b', linestyle='--', label='CNN Test Accuracy')
plt.axhline(y=cnn_precision, color='g', linestyle='--', label='CNN Precision')
plt.axhline(y=cnn_recall, color='r', linestyle='--', label='CNN Recall')
plt.axhline(y=cnn_f1, color='m', linestyle='--', label='CNN F1 Score')
plt.title('CNN Model Performance Metrics')
plt.xlabel('Epochs')
plt.ylabel('Metrics')
plt.legend()
plt.grid(True)
plt.show()

# Plot SVC model metrics
plt.figure(figsize=(12, 6))
plt.axhline(y=svc_accuracy, color='b', linestyle='--', label='SVC Test Accuracy')
plt.axhline(y=svc_precision, color='g', linestyle='--', label='SVC Precision')
plt.axhline(y=svc_recall, color='r', linestyle='--', label='SVC Recall')
plt.axhline(y=svc_f1, color='m', linestyle='--', label='SVC F1 Score')
plt.title('SVC Model Performance Metrics')
plt.xlabel('Epochs')
plt.ylabel('Metrics')
plt.legend()
plt.grid(True)
plt.show()
```

```
# Print the metrics
print("CNN Model Metrics:")
print("Accuracy:", cnn_accuracy)
print("Precision:", cnn_precision)
print("Recall:", cnn_recall)
print("F1 Score:", cnn_f1)
print("\nSVC Model Metrics:")
print("Accuracy:", svc_accuracy)
print("Precision:", svc_precision)
print("Recall:", svc_recall)
print("F1 Score:", svc_f1)

# Determine the best model
if cnn_accuracy > svc_accuracy:
    print("\nCNN Model is the best model based on Accuracy.")
elif cnn_accuracy < svc_accuracy:
    print("\nSVC Model is the best model based on Accuracy.")
else:
    print("\nBoth models have the same Accuracy.")
```

```
epochs = range(1, 11)

plt.figure(figsize=(12, 6))

# Plot CNN model metrics
plt.plot(epochs, history.history['accuracy'], label='CNN Accuracy', marker='o')
plt.plot(epochs, history.history['val_accuracy'], label='CNN Validation Accuracy', marker='o')

# Plot SVC model metrics
plt.axhline(y=svc_accuracy, color='r', linestyle='--', label='SVC Accuracy')

plt.title('Model Performance Comparison')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.grid(True)
plt.show()
```

```
import random

# Select 9 random indices
random_indices = random.sample(range(len(xtest)), 9)

# Print 9 random sample test results using the CNN model
plt.figure(figsize=(12, 8))
for i, idx in enumerate(random_indices):
    prediction = cnn_predictions[idx]
    image = xtest[idx].reshape(200, 200)  # Reshape image to original dimensions
    plt.subplot(3, 3, i+1)
    plt.imshow(image, cmap='gray')
    plt.axis('off')
    if prediction >= 0.5:
        plt.title("Predicted: Tumor")
    else:
        plt.title("Predicted: No Tumor")
plt.show()
```

Brain tumour SVC and logistic regression:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

```
import zipfile

zip_path = "C:\\Users\\anish\\Downloads\\archive (2).zip"
extract_path = "C:\\Users\\anish\\Downloads\\archive (2)"  # Specify the directory to extract the files

# Extract the ZIP file
with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    zip_ref.extractall(extract_path)

print("Dataset extracted successfully.")
```

```
import cv2
import os
import zipfile

dataset_zip_path = r"C:\\Users\\anish\\Downloads\\archive (2).zip"
extracted_folder_path = r"C:\\Users\\anish\\Downloads\\archive (2)\\Training"

# Extract the contents of the ZIP file to the specified folder
with zipfile.ZipFile(dataset_zip_path, 'r') as zip_ref:
    zip_ref.extractall(extracted_folder_path)

classes = {'no_tumor': 0, 'pituitary_tumor': 1}

X = []  # List to store image data
Y = []  # List to store corresponding labels

for cls in classes:
    pth = os.path.join(extracted_folder_path, cls)
    for j in os.listdir(pth):
        img_path = os.path.join(pth, j)
        img = cv2.imread(img_path, 0)  # Read the image in grayscale
        img = cv2.resize(img, (200, 200))  # Resize the image to (200, 200)
        X.append(img)  # Append the image to X list
        Y.append(classes[cls])  # Append the corresponding label to Y list
```

```
X = np.array(X)
Y = np.array(Y)

X_updated = X.reshape(len(X), -1)
```

```
np.unique(Y)
```

```

pd.Series(Y).value_counts()
```

```
X.shape, X_updated.shape
```

```
plt.imshow(X[0], cmap='gray')
```

```
X_updated = X.reshape(len(X), -1)
X_updated.shape
```

```
xtrain, xtest, ytrain, ytest = train_test_split(X_updated, Y, random_state=10,
                                               test_size=.20)
```

```
xtrain.shape, xtest.shape
```

```
print(xtrain.max(), xtrain.min())
print(xtest.max(), xtest.min())
xtrain = xtrain/255
xtest = xtest/255
print(xtrain.max(), xtrain.min())
print(xtest.max(), xtest.min())
```

```
from sklearn.decomposition import PCA
```

```
print(xtrain.shape, xtest.shape)

pca = PCA(.98)
# pca_train = pca.fit_transform(xtrain)
# pca_test = pca.transform(xtest)
pca_train = xtrain
pca_test = xtest
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
```

```
import warnings
warnings.filterwarnings('ignore')

lg = LogisticRegression(C=0.1)
lg.fit(xtrain, ytrain)
```

```
sv = SVC()
sv.fit(xtrain, ytrain)
```

```
print("Training Score:", lg.score(xtrain, ytrain))
print("Testing Score:", lg.score(xtest, ytest))
```

```
print("Training Score:", sv.score(xtrain, ytrain))
print("Testing Score:", sv.score(xtest, ytest))
```

```
pred = sv.predict(xtest)
```

```
misclassified=np.where(ytest!=pred)
misclassified
```

```
print("Total Misclassified Samples: ",len(misclassified[0]))
print(pred[36],ytest[36])
```

```
dec = {0:'No Tumor', 1:'Positive Tumor'}
```

```
import cv2
import matplotlib.pyplot as plt
import os

plt.figure(figsize=(12, 8))
c = 1

for i in os.listdir("C:\\Users\\anish\\Downloads\\archive (2)\\Testing\\pituitary_tumor")[:9]:
    plt.subplot(3, 3, c)

    img = cv2.imread("C:\\Users\\anish\\Downloads\\archive (2)\\Testing\\pituitary_tumor\\" + i, 0)
    img1 = cv2.resize(img, (200, 200))
    img1 = img1.reshape(1, -1) / 255

    # Replace sv.predict with your prediction code
    p = sv.predict(img1)

    plt.title(dec[p[0]])
    plt.imshow(img, cmap='gray')
    plt.axis('off')
    c += 1

dataset_train_path = r"C:\\Users\\anish\\Downloads\\archive (2)\\Training"
dataset_test_path = r"C:\\Users\\anish\\Downloads\\archive (2)\\Testing"
```

```
import matplotlib.pyplot as plt

# Accuracy scores
lg_train_accuracy = lg.score(xtrain, ytrain)
lg_test_accuracy = lg.score(xtest, ytest)
sv_train_accuracy = sv.score(xtrain, ytrain)
sv_test_accuracy = sv.score(xtest, ytest)

# Plotting
```

```
labels = ['Logistic Regression (Train)', 'Logistic Regression (Test)', 'SVM (Train)', 'SVM (Test)']
scores = [lg_train_accuracy, lg_test_accuracy, sv_train_accuracy, sv_test_accuracy]

plt.figure(figsize=(10, 6))
plt.bar(labels, scores, color=['blue', 'orange', 'green', 'red'])
plt.title('Accuracy of Logistic Regression and SVM')
plt.ylabel('Accuracy')
plt.ylim(0, 1)  # Set y-axis limits from 0 to 1
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.show()
```

```
from sklearn.metrics import precision_score, recall_score, f1_score

# Predictions
lg_pred_train = lg.predict(xtrain)
lg_pred_test = lg.predict(xtest)
sv_pred_train = sv.predict(xtrain)
sv_pred_test = sv.predict(xtest)

# Metrics
lg_train_precision = precision_score(ytrain, lg_pred_train)
lg_test_precision = precision_score(ytest, lg_pred_test)
sv_train_precision = precision_score(ytrain, sv_pred_train)
sv_test_precision = precision_score(ytest, sv_pred_test)

lg_train_recall = recall_score(ytrain, lg_pred_train)
lg_test_recall = recall_score(ytest, lg_pred_test)
sv_train_recall = recall_score(ytrain, sv_pred_train)
sv_test_recall = recall_score(ytest, sv_pred_test)

lg_train_f1 = f1_score(ytrain, lg_pred_train)
lg_test_f1 = f1_score(ytest, lg_pred_test)
sv_train_f1 = f1_score(ytrain, sv_pred_train)
sv_test_f1 = f1_score(ytest, sv_pred_test)

# Displaying metrics
print("Logistic Regression Metrics:")
print("Training Precision:", lg_train_precision)
print("Testing Precision:", lg_test_precision)
print("Training Recall:", lg_train_recall)
print("Testing Recall:", lg_test_recall)
print("Training F1 Score:", lg_train_f1)
print("Testing F1 Score:", lg_test_f1)
print("\n")

print("SVM Metrics:")
print("Training Precision:", sv_train_precision)
print("Testing Precision:", sv_test_precision)
print("Training Recall:", sv_train_recall)
print("Testing Recall:", sv_test_recall)
print("Training F1 Score:", sv_train_f1)
print("Testing F1 Score:", sv_test_f1)
```

```
import matplotlib.pyplot as plt

# Define metrics and their scores
metrics = ['Precision', 'Recall', 'F1-score']
lg_train_scores = [lg_train_precision, lg_train_recall, lg_train_f1]
lg_test_scores = [lg_test_precision, lg_test_recall, lg_test_f1]
sv_train_scores = [sv_train_precision, sv_train_recall, sv_train_f1]
sv_test_scores = [sv_test_precision, sv_test_recall, sv_test_f1]

# Plotting
plt.figure(figsize=(10, 6))

# Plotting lines for Logistic Regression
plt.plot(metrics, lg_train_scores, marker='o', color='blue', label='Logistic Regression (Train)')
plt.plot(metrics, lg_test_scores, marker='o', color='lightblue', label='Logistic Regression
(Test)')

# Plotting lines for SVM
plt.plot(metrics, sv_train_scores, marker='o', color='green', label='SVM (Train)')
plt.plot(metrics, sv_test_scores, marker='o', color='lightgreen', label='SVM (Test)')

# Adding labels and title
plt.xlabel('Metrics')
plt.ylabel('Score')
plt.title('Performance Metrics Comparison')
plt.legend()
plt.grid(True)

# Show plot
plt.show()
```

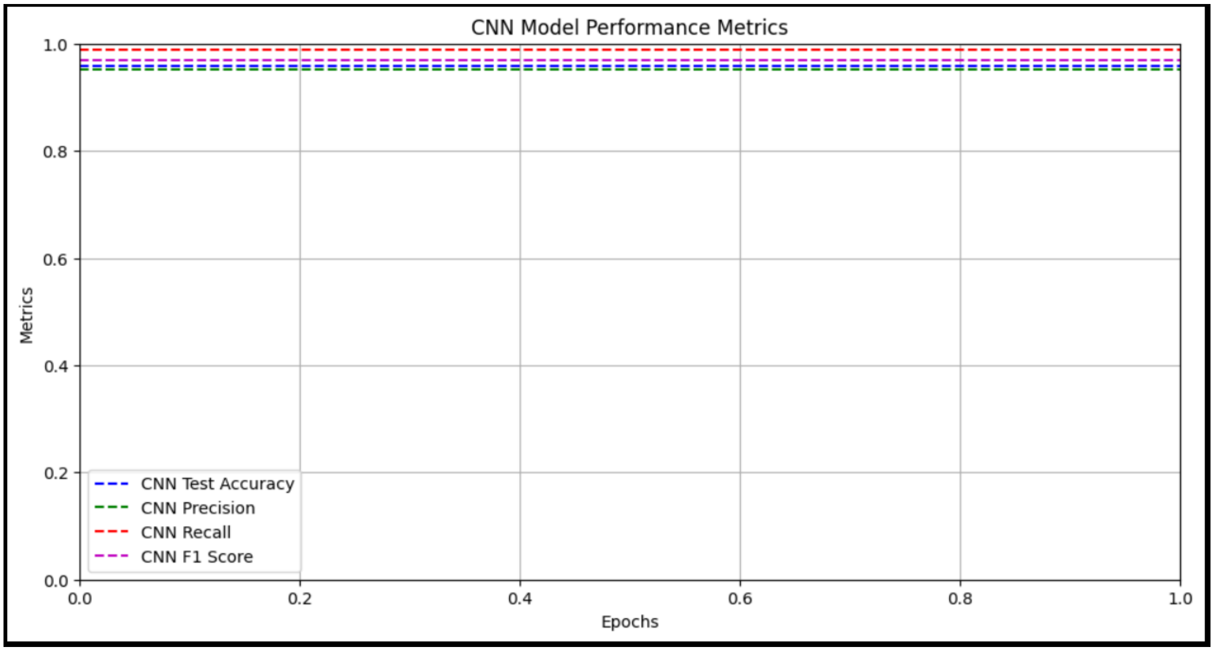
```
# Calculate average scores for each algorithm on the testing set
lg_avg_test_score = (lg_test_precision + lg_test_recall + lg_test_f1) / 3
sv_avg_test_score = (sv_test_precision + sv_test_recall + sv_test_f1) / 3

# Compare average scores
if lg_avg_test_score > sv_avg_test_score:
    print("Logistic Regression performs better on the testing set.")
    best_algorithm = "Logistic Regression"
    best_scores = [lg_test_precision, lg_test_recall, lg_test_f1]
else:
    print("SVM performs better on the testing set.")
    best_algorithm = "SVM"
    best_scores = [sv_test_precision, sv_test_recall, sv_test_f1]

# Print the average scores for the best algorithm
print("Average Precision for", best_algorithm + ":", best_scores[0])
print("Average Recall for", best_algorithm + ":", best_scores[1])
print("Average F1-score for", best_algorithm + ":", best_scores[2])
```

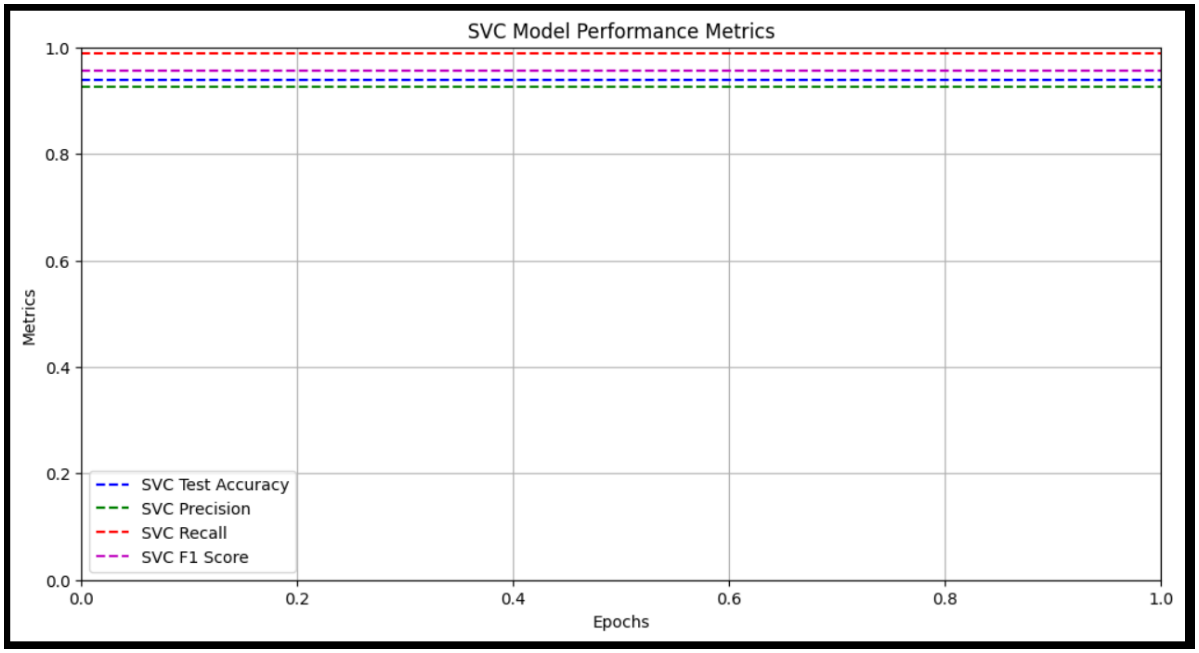
CHAPTER 5

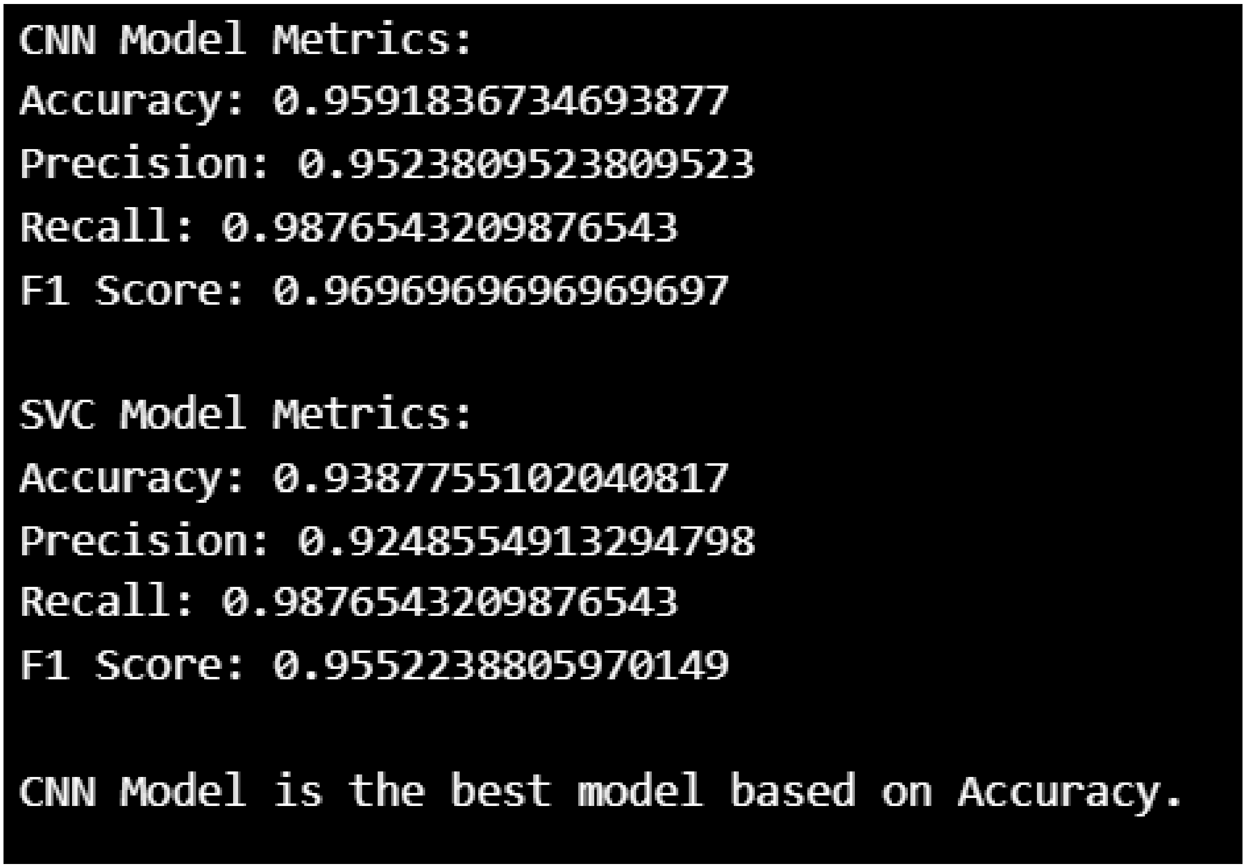
RESULTS

Screenshots:

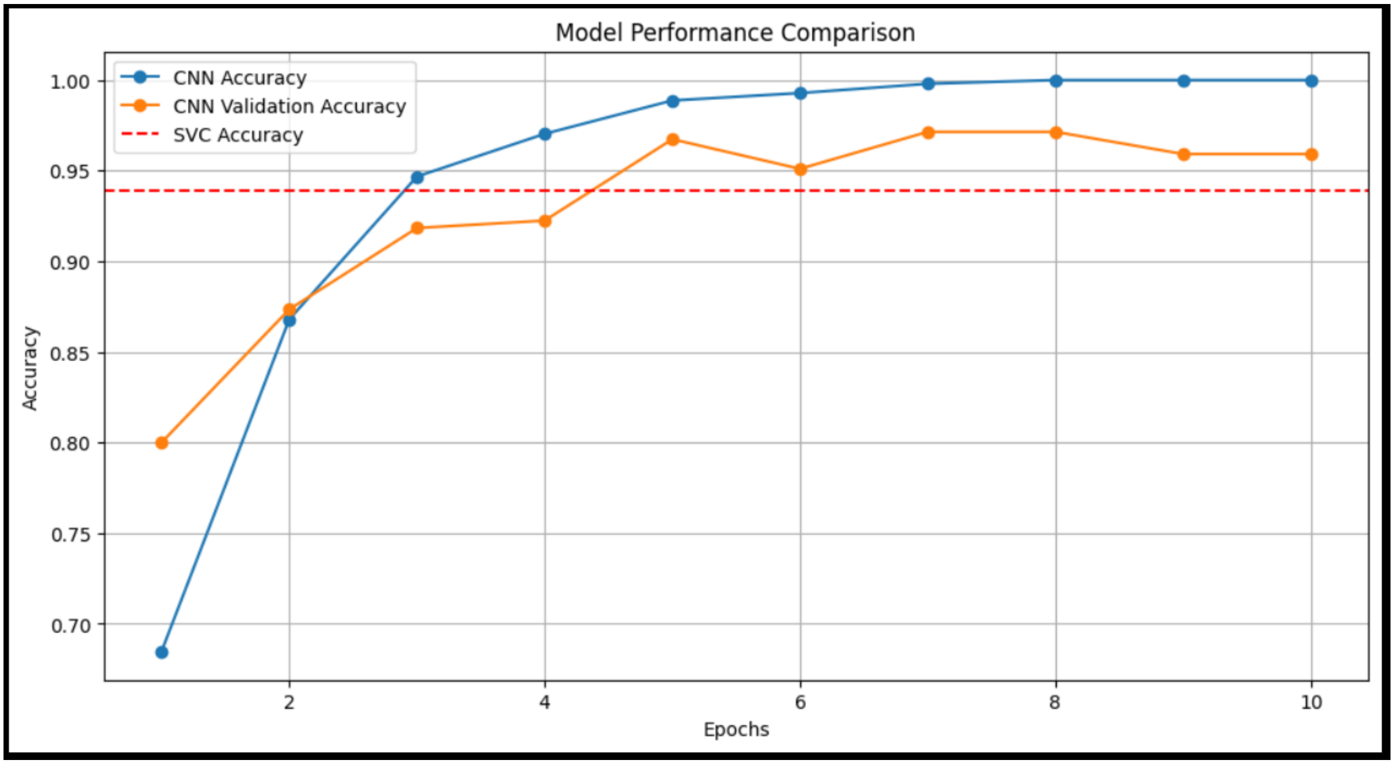
Brain tumour cnn and svc:

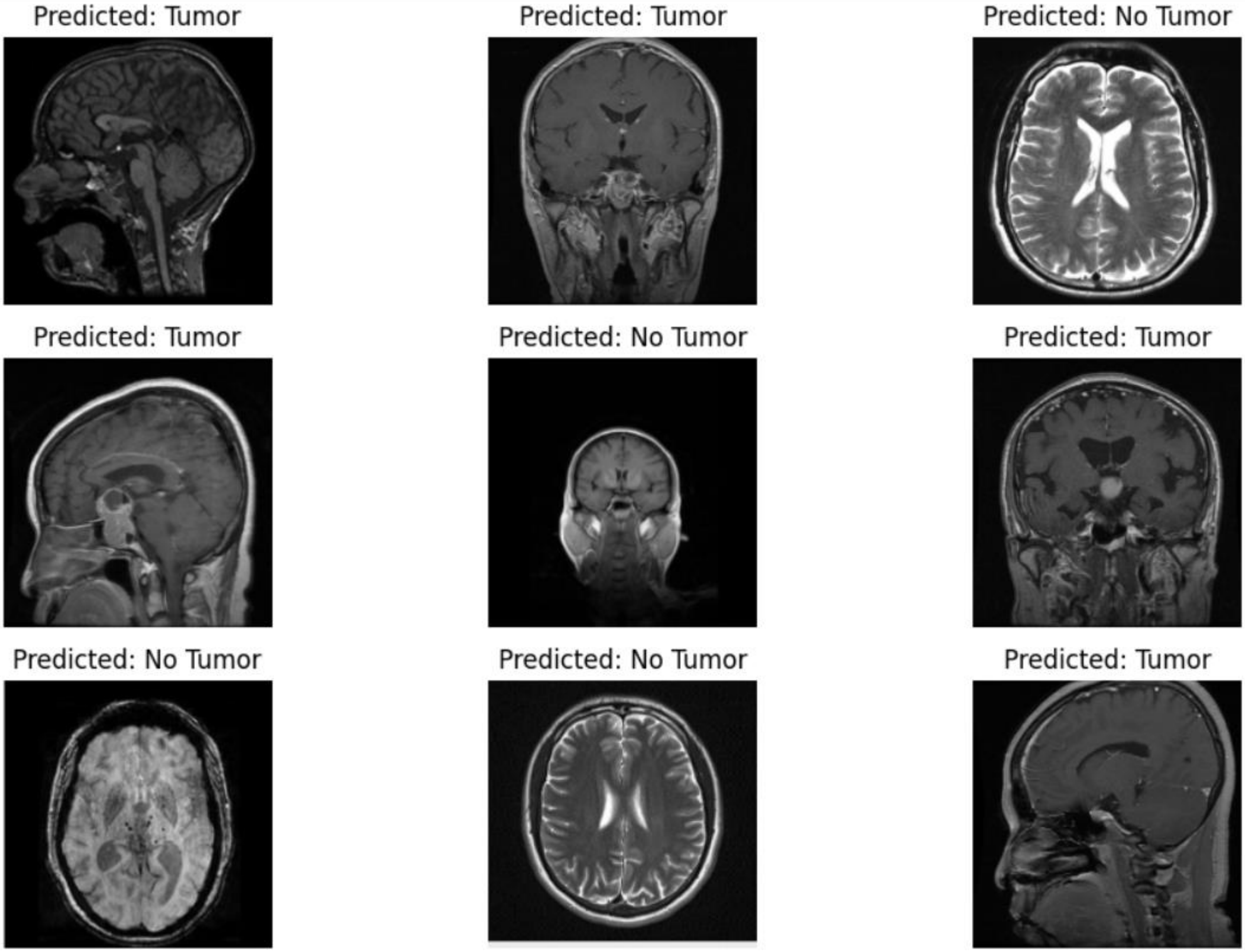




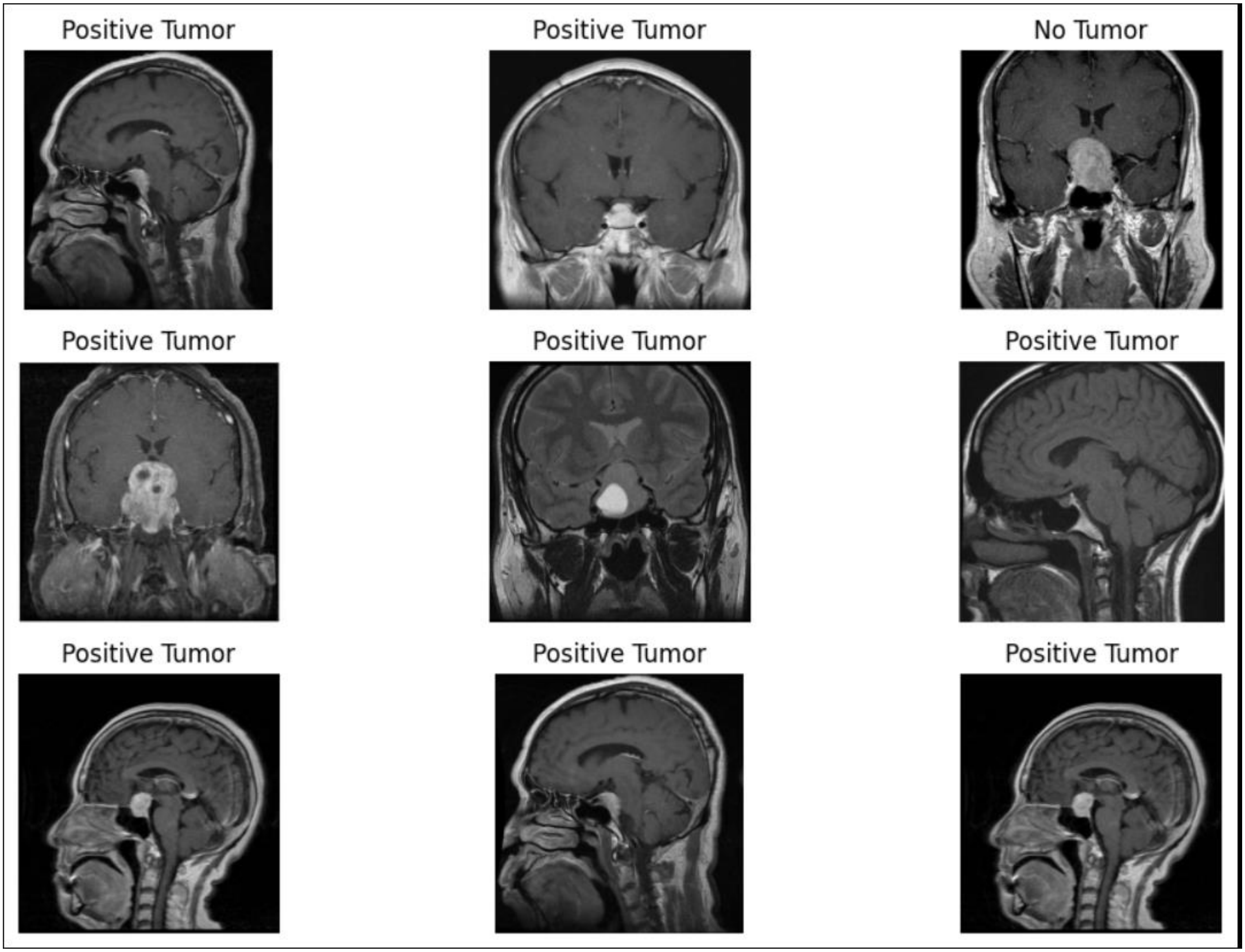


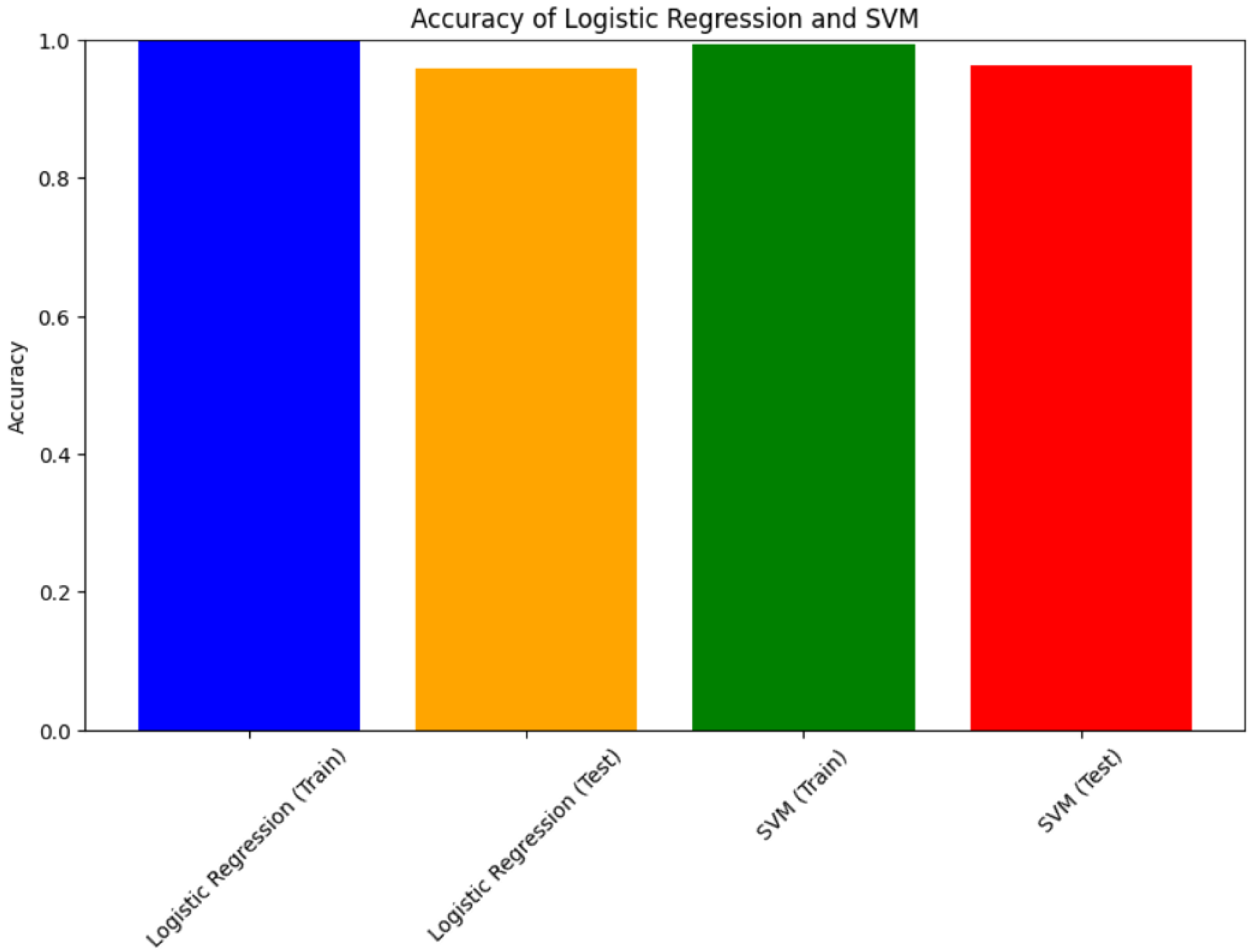
```
CNN Model Metrics:
Accuracy: 0.9591836734693877
Precision: 0.9523809523809523
Recall: 0.9876543209876543
F1 Score: 0.9696969696969697

SVC Model Metrics:
Accuracy: 0.9387755102040817
Precision: 0.9248554913294798
Recall: 0.9876543209876543
F1 Score: 0.9552238805970149

CNN Model is the best model based on Accuracy.
```

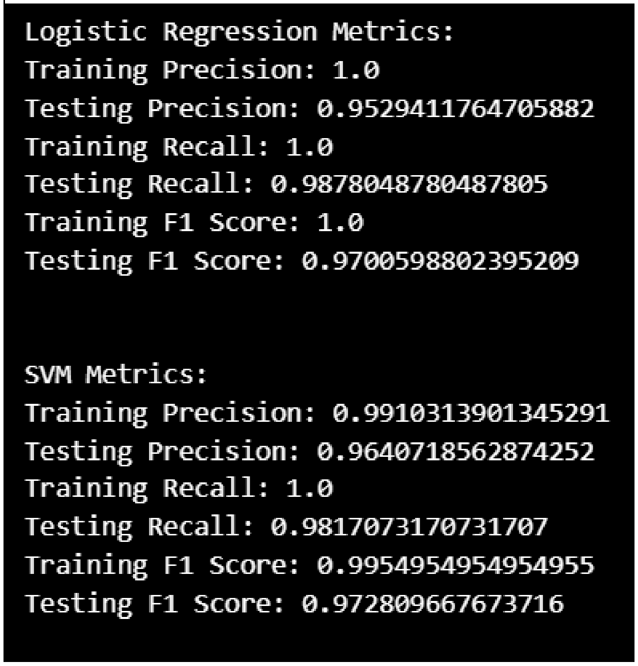



Predicted: Tumor
Predicted: Tumor
Predicted: No Tumor
Predicted: Tumor
Predicted: No Tumor
Predicted: Tumor
Predicted: No Tumor
Predicted: No Tumor
Predicted: Tumor

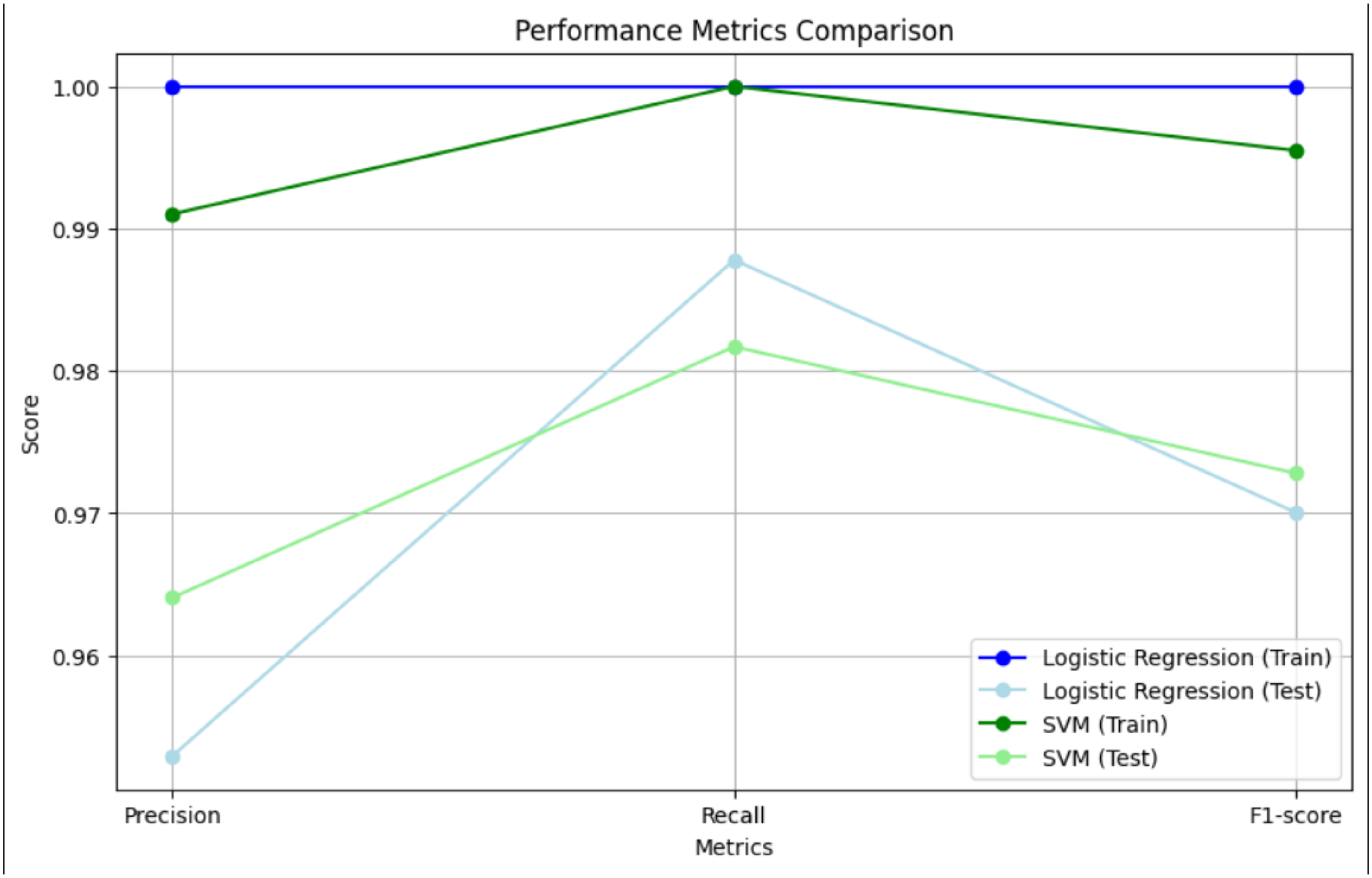
Brain tumour SVC and logistic regression:

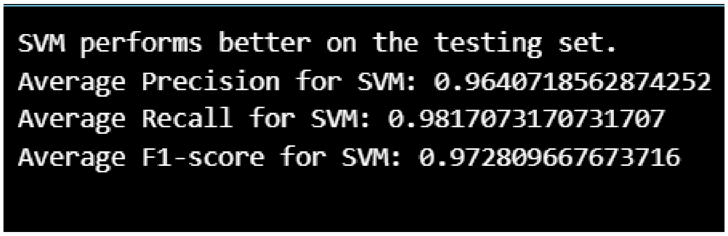






```
Logistic Regression Metrics:
Training Precision: 1.0
Testing Precision: 0.9529411764705882
Training Recall: 1.0
Testing Recall: 0.9878048780487805
Training F1 Score: 1.0
Testing F1 Score: 0.9700598802395209


SVM Metrics:
Training Precision: 0.9910313901345291
Testing Precision: 0.9640718562874252
Training Recall: 1.0
Testing Recall: 0.9817073170731707
Training F1 Score: 0.9954954954954955
Testing F1 Score: 0.972809667673716
```





```
SVM performs better on the testing set.
Average Precision for SVM: 0.9640718562874252
Average Recall for SVM: 0.9817073170731707
Average F1-score for SVM: 0.972809667673716
```

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENTS

The application of deep learning to predict BRAIN TUMOR from Brain X-ray images and assess disease severity has shown significant promise in providing rapid and accurate diagnosis. By leveraging pre-trained convolutional neural networks (CNNs) such as CNN, SVC and logistic regression as well as exploring SIAMESE networks, deep learning models have demonstrated high performance in distinguishing between BRAIN TUMOR positive and healthy Brain X-rays.
CNN, in particular, has shown notable accuracy and reliability in this task, making it a strong choice for medical imaging applications. The potential to predict the severity of the disease further enables healthcare professionals to make more informed decisions about patient management and treatment strategies.

The use of these advanced technologies can support healthcare systems in effectively managing the ongoing pandemic by streamlining diagnosis, prioritizing care, and optimizing resource allocation. Additionally, ethical and legal considerations, including patient privacy and data protection, must be maintained throughout the process to ensure responsible use of AI in healthcare.

Future Enhancements:
While deep learning models have shown promising results, there is still room for further research and improvement in this area:

Integration with Other Imaging Modalities: Combining Brain X-rays with other imaging modalities such as computed tomography (CT) scans and magnetic resonance imaging (MRI) could improve the accuracy and robustness of BRAIN TUMOR detection and severity prediction.
Domain Adaptation: Future work should focus on improving the generalization of models across different populations, imaging equipment, and healthcare settings. Domain adaptation techniques can help models perform consistently across diverse datasets.

Explainability and Interpretability: Enhancing model explainability can provide insights into how models make predictions, aiding healthcare professionals in understanding the reasoning behind the model's decisions and building trust in the technology.

Real-time Monitoring and Feedback: Integrating deep learning models into clinical workflows can provide real-time monitoring and feedback for healthcare professionals, enabling them to make timely and informed decisions.

Continuous Learning: Developing systems that continuously learn from new data and adapt over time can help maintain model accuracy and relevancy, ensuring sustained performance as the pandemic evolves.

Collaboration and Data Sharing: Encouraging collaboration and data sharing across research institutions and healthcare organizations can facilitate the development of more robust and generalizable models.

Ethical Considerations: Ensuring ethical AI practices, including transparency, fairness, and accountability, is crucial for the responsible deployment of deep learning models in healthcare. By addressing these future enhancements, the potential impact of deep learning on BRAIN TUMOR diagnosis and severity prediction from Brain X-ray images can be maximized, ultimately leading to improved patient outcomes and more efficient healthcare delivery.

CHAPTER 7

REFERENCES

1. Accurate brain tumor detection using deep convolutional neural network Md. Saikat Islam Khan,a Anichur Rahman,a,b,⁎ Tanoy Debnath,a,c Md. Razaul Karim,a Mostofa Kamal Nasir,a Shahab S. Band,d,⁎ Amir Mosavi,e,f,⁎ and Iman Dehzangig,h

2. Gorbalenya A.E., Baker S.C., Baric R., Groot R.J.d., Drosten C., Gulyaeva A.A., Haagmans B.L., Lauber C., Leontovich A.M., Neuman B.W., et al. Severe acute respiratory syndrome-related tumor: The species and its viruses—A statement of the Tumor Study Group. bioRxiv. 2020

3. Stoecklin S.B., Rolland P., Silue Y., Mailles A., Campese C., Simondon A., Mechain M., Meurice L., Nguyen M., Bassi C., et al. First cases of tumor disease 2019 (BRAIN TUMOR) in France: Surveillance, investigations and control measures, January 2020. Eurosurveillance. 2020;25:2000094.

4. MRI-based brain tumour image detection using CNN based deep learning method Author links open overlay panelArkapravo Chattopadhyay, Mausumi Maitra

5. Chavez S., Long B., Koyfman A., Liang S.Y. Tumor Disease (BRAIN TUMOR): A primer for emergency physicians. Am. J. Emerg. Med. 2021;44:220–229. doi: 10.1016/j.ajem.2020.03.036.

6. Al-Waisy A., Mohammed M.A., Al-Fahdawi S., Maashi M., Garcia-Zapirain B., Abdulkareem K.H., Mostafa S., Kumar N.M., Le D.N. Covid-deepnet: Hybrid multimodal deep learning system for improving Brain tumor pneumonia detection in Brain X-ray images. Comput. Mater. Contin. 2021;67:2409–2429. doi: 10.32604/cmc.2021.012955.

7. Elghamrawy S.M., Hassnien A.E., Snasel V. Optimized Deep Learning-Inspired Model for the Diagnosis and Prediction of BRAIN TUMOR. Comput. Mater. Contin. 2021;67:2353–2371. doi: 10.32604/cmc.2021.014767.

8. Armstrong M. The Countries with the Most BRAIN TUMOR Cases. 2021. [(accessed on 13 December 2021)].

9. Asai T. BRAIN TUMOR: Accurate Interpretation of Diagnostic Tests—A Statistical Point of View. [(accessed on 13 December 2021)];2021

10. Gaál G., Maga B., Lukács A. Attention u-net based adversarial architectures for Brain X-ray lung segmentation. arXiv. 20202003.10304

11. Narin A., Kaya C., Pamuk Z. Automatic detection of tumor disease (Brain tumor) using X-ray images and deep convolutional neural networks. Pattern Anal. Appl. 2021;24:1207–1220. doi: 10.1007/s10044-021-00984-y.

12. Das N.N., Kumar N., Kaur M., Kumar V., Singh D. Automated deep transfer learning-based approach for detection of BRAIN TUMOR infection in Brain X-rays. Irbm. 2020 in press.

13. Chatfield K., Simonyan K., Vedaldi A., Zisserman A. Return of the devil in the details: Delving deep into convolutional nets. arXiv. 20141405.3531

14. Simonyan K., Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv. 20141409.1556

15. Shen D., Wu G., Suk H.I. Deep learning in medical image analysis. Annu. Rev. Biomed. Eng. 2017;19:221–248. doi: 10.1146/annurev-bioeng-071516-044442.

16. Litjens G., Kooi T., Bejnordi B.E., Setio A.A.A., Ciompi F., Ghafoorian M., Van Der Laak J.A., Van Ginneken B., Sánchez C.I. A survey on deep learning in medical image analysis. Med. Image Anal. 2017;42:60–88. doi: 10.1016/j.media.2017.07.005

17. Wang L., Lin Z.Q., Wong A. Covid-net: A tailored deep convolutional neural network design for detection of Brain tumor cases from Brain X-ray images. Sci. Rep. 2020;10:1–12. doi: 10.1038/s41598-020-76550-z.

18. Hemdan E.E.-D., Shouman M.A., Karar M.E. COVIDXnet: A framework of deep learning classifiers to diagnose BRAIN TUMOR in X-ray images. arXiv. 20202003.11055

19. Zhang J., Xie Y., Li Y., Shen C., Xia Y. Brain tumor screening on Brain X-ray images using deep learning based anomaly detection. arXiv. 20202003.12338

20. Ozturk T., Talo M., Yildirim E.A., Baloglu U.B., Yildirim O., Acharya U.R. Automated detection of BRAIN TUMOR cases using deep neural networks with X-ray images. Comput. Biol. Med. 2020;121:103792. doi: 10.1016/j.compbiomed.2020.103792.