UNIVERSITY OF CALIFORNIA

Los Angeles

Brain Tumor Detection through Machine Learning Classification

A thesis submitted in partial satisfaction

of the requirements for the degree

Master of Applied Statistics and Data Science

by

Long Ling

2024

ABSTRACT OF THE THESIS

Brain Tumor Detection through Machine Learning Classification

by

Long Ling
Master of Applied Statistics and Data Science
University of California, Los Angeles, 2024
Professor Yingnian Wu, Chair

This article examines the efficacy of machine learning techniques in analyzing brain tumor MRI images with the aim of reducing the workload of medical professionals. The study compared various machine learning methods for processing MRI data and their accuracy. Results show that convolutional neural networks (CNN), including Custom CNN, ResNet v2, and VGG 16, outperform traditional machine learning algorithms such as random forest, K-NN, and SVM in tumor classification accuracy. VGG 16 shows the highest accuracy, reaching 98.73%, and has the smallest loss compared to other CNN models. These data results provide insights into the comparative performance of machine learning models, revealing their strengths and limitations in processing different brain tumor images.

The thesis of Long Ling is approved.

Michael Tsiang

Frederic R. Paik Schoenberg

Yingnian Wu, Committee Chair

University of California, Los Angeles

2024

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

Brain tumor is a serious disease that endangers health and seriously affects patients' quality of daily life. There are different types of brain tumors, the common ones are meningioma tumors, glioma tumors, and pituitary tumors. Regardless of whether the brain tumor is benign or malignant, when the tumor size increases to a certain extent, patients usually experience severe headache, dizziness, nausea, retching and other uncomfortable symptoms. In addition, brain tumor expansion may compress surrounding important blood vessels and nerves, causing patients to experience epileptic seizures or significant limb movement dysfunction. Prompt detection and treatment of brain tumors is crucial. Early diagnosis and intervention can improve patient cure and survival rates and reduce the severity of symptoms and complications.

Despite many advances in advanced imaging technology over the past few decades, traditional structural magnetic resonance imaging (MRI) remains the standard method of care imaging in neuro-oncology practice. [14] Given the large scale of MRI image data, machine learning algorithms are often used as tools to assist in the imaging assessment of brain tumors. The current classification method for brain tumor MRI images is mainly the traditional classification method of manual identification. Clinicians visually inspect MRI images to identify any abnormal features, and rely on experience combined with lesion characteristics observed in MRI images, such as tumor size, shape, and location, to determine the type of tumor. Machine learning algorithms can serve as aids in detection and diagnosis, aiding doctors in interpreting medical imaging results and decreasing interpretation times. [4]

Compared with traditional manual judgment, machine learning technology can help improve efficiency and reduce the workload of medical staff. Different machine learning methods use different data processing methods when processing MRI images, and the classification results obtained will also be different.

Kowshika, A., et al. [8] describe in the journal NeuroQuantology a protocol for detailed information on stroke and tumor cells. In their study, they first denoised and decolorized the images. They then used a logistic regression model to determine the type of test image and achieved a 98% accuracy.

Ravikumar Gurusamy et al. [6] performed preprocessing and feature extraction of MRI images. They employ extensive pre-processing techniques to remove unnecessary noise. Finally, they extracted the best set of features from these images and the performance of neural network, KNN and the proposed method exceeded 96% in both positive and negative predictive values.

CNN (Convolutional Neural Network) is a machine learning method widely used in image classification. DC Febrianto et al. [5] believe that convolutional neural networks are sufficient to diagnose brain tumors from MRI images. They made 2 CNN models as comparison materials and then compared them using standard deviation, mean and mean of loss, accuracy and F1 score. The model using 2 convolutions gave better results, with an accuracy of 93% and a loss of 0.23264. This study concluded that the number of convolutional layers impacts the classification quality. Adding more training layers can improve the result accuracy, but it takes longer to train the model.

S. Deepak et al. [3] combined CNN and SVM to classify medical images. It was evaluated using the Figshare open data set, and the overall classification accuracy reached 95.82%. It is concluded that when there is less available training data, the performance of CNN features and SVM classifier is better, and the CNN-SVM combination has lower computational complexity and memory requirements.

The main goal of this study is to conduct a comprehensive comparison of different machine learning models for classifying brain tumor MRI images to evaluate their performance in brain tumor classification tasks. By analyzing the various machine learning methods involved in the above research literature, we can understand the advantages and limitations of these models when processing different types of brain tumor images.

CHAPTER 2

Data Introduction

The project used data from Kaggle [1]. The dataset contains multiple types of brain tumors as well as MRI images of healthy brains. It contains four sub-folders, namely glioma tumor, meningioma tumor, no tumor and pituitary tumor. Table 2.1 shows the count for each class.

Class	**Count**
glioma tumor	926
meningioma tumor	937
no tumor	500
pituitary tumor	901

Table 2.1: Class Distribution

2.1 Data Preprocessing

In this project, OpenCV was used to read the image, convert it to grayscale, and resize it to a fixed size of 200*200 pixels. When using the CNN model, the data is re-read without conversion to grayscale and has a fixed size of 150*150. The code initializes empty lists X and Y to store image data and their respective labels. The image data is then converted into a one-dimensional vector form by reshaping the X array so that it can be fed into the machine learning model. The data set was then randomly divided into training and test sets with 80% training data and 20% test data. The data set has a total of 3264 images, : 2,611 for training and 653 for testing. Figure 2.1 shows the counts and proportions of each class

in the training set and test set.

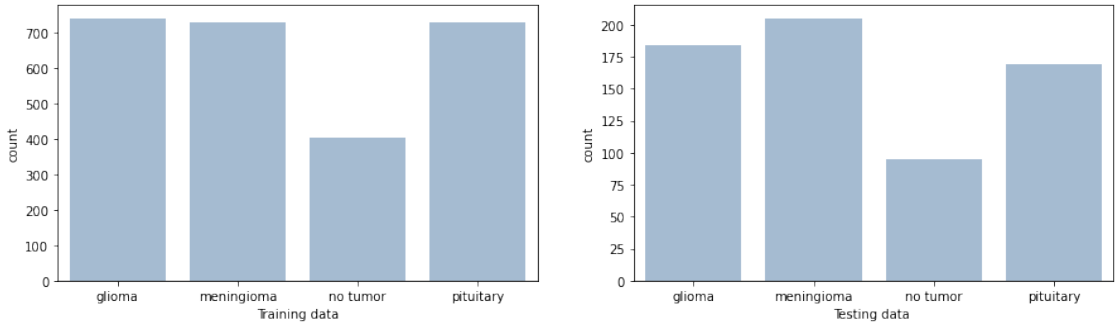

Figure 2.1: Class Distribution in Training and Test Sets

The following are sample images of each class in the training set. We can see that there are MRI images from different angles in each class.

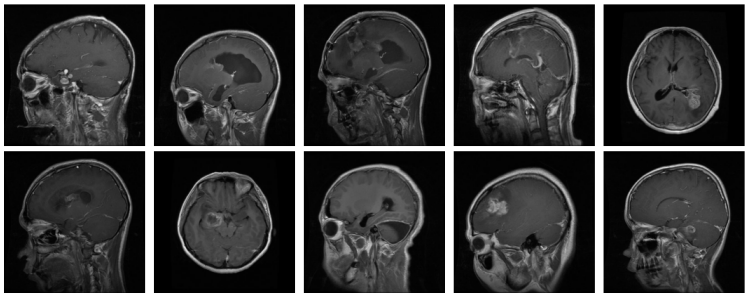

Figure 2.2: Glioma Tumor Sample

The next step is to normalize the data. We understand that the pixel value of image data ranges from 0 to 255. The code then divides the pixel values in the training and test sets by 255, which scales them to a range between 0 and 1. This helps improve the training efficiency and stability of the model. Then we can use machine learning methods to make classification for these images.

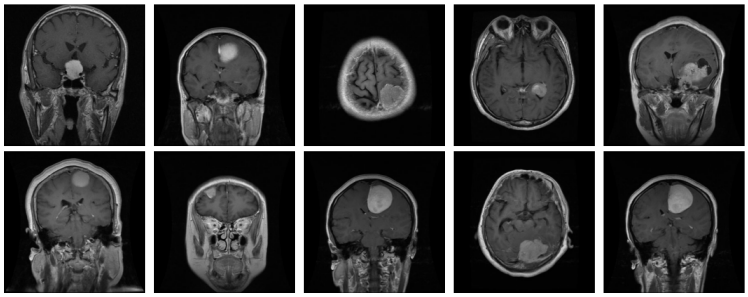

Figure 2.3: Meningioma Tumor Sample

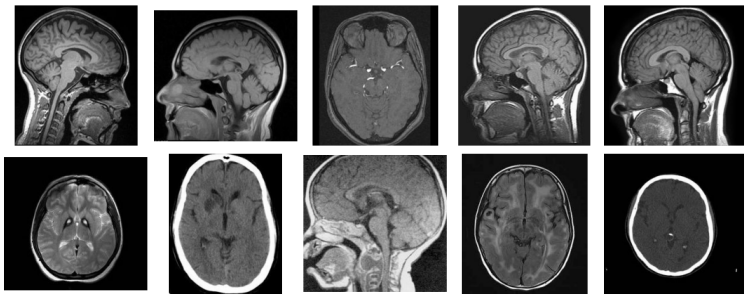

Figure 2.4: No Tumor Sample

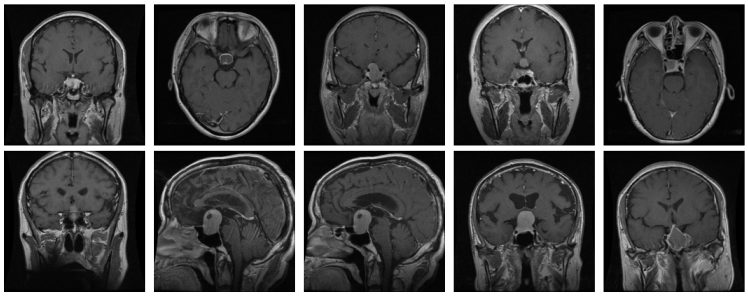

Figure 2.5: Pituitary Tumor Sample

CHAPTER 3

Methodology and Models

3.1 Random Forest

As a supervised learning model, random forests operate by building a large number of decision trees while training. For classification tasks, the output of a random forest is the class selected by the majority of trees. [16] Specific steps are as follows:

- Use bootstrap sampling to create samples from the original dataset. Build a decision tree for each sample.
- Recursive binary splitting is performed to produce two subsets at each split.
- Predict the response variable with majority voting.

Figure 3.1 shows the process of using a random forest classifier to perform a classification task. In this study, a random forest classifier was used to train on the given training data. Predict the test data, calculate the prediction accuracy and then output the accuracy and classification report.

The classification report provides precision, recall, F1-score, and support for each category, along with the overall accuracy.

The formulas for calculating precision, recall, and F1-score are as follows [9]:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Postive}}$$

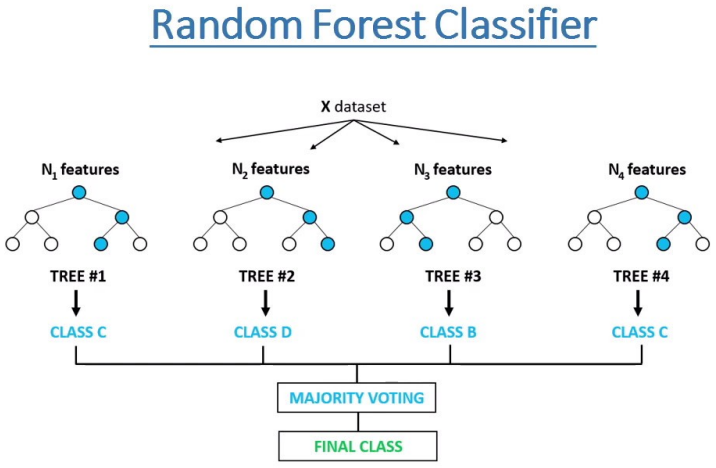

Figure 3.1: Random Forest Classifier [7]

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

$$\text{F1 Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

F1 Score is the weighted harmonic average of precision and recall, offers a comprehensive evaluation of the model's performance. Table 3.1 is the classification report we got for the test set under random forest.

According to the Table 3.1 , we can see that the precision, recall, and F1-score of each category are relatively high, showing that the model performs well on each category. In particular, the F1 score for glioma tumor reaches 0.93, which means that the model performs best in predicting pituitary tumor, while meningioma tumor and no tumor also have high F1 scores. Overall Accuracy is 0.87.

Next is the confusion matrix of this model. As can be seen from Figure 3.2, the overall prediction accuracy is relatively high. For a category, the number predicted to be of some other category is generally within ten, and we can see that the only number that exceeds 10 is that thirty MRI images of glioma tumors were identified as meningioma tumor images.

Class	Precision	Recall	F1 score	Support
glioma tumor	0.93	0.77	0.84	184
meningioma tumor	0.82	0.90	0.86	205
no tumor	0.81	0.91	0.86	95
pituitary tumor	0.92	0.93	0.93	169
Accuracy			0.87	653

Table 3.1: Random Forest Classification Report

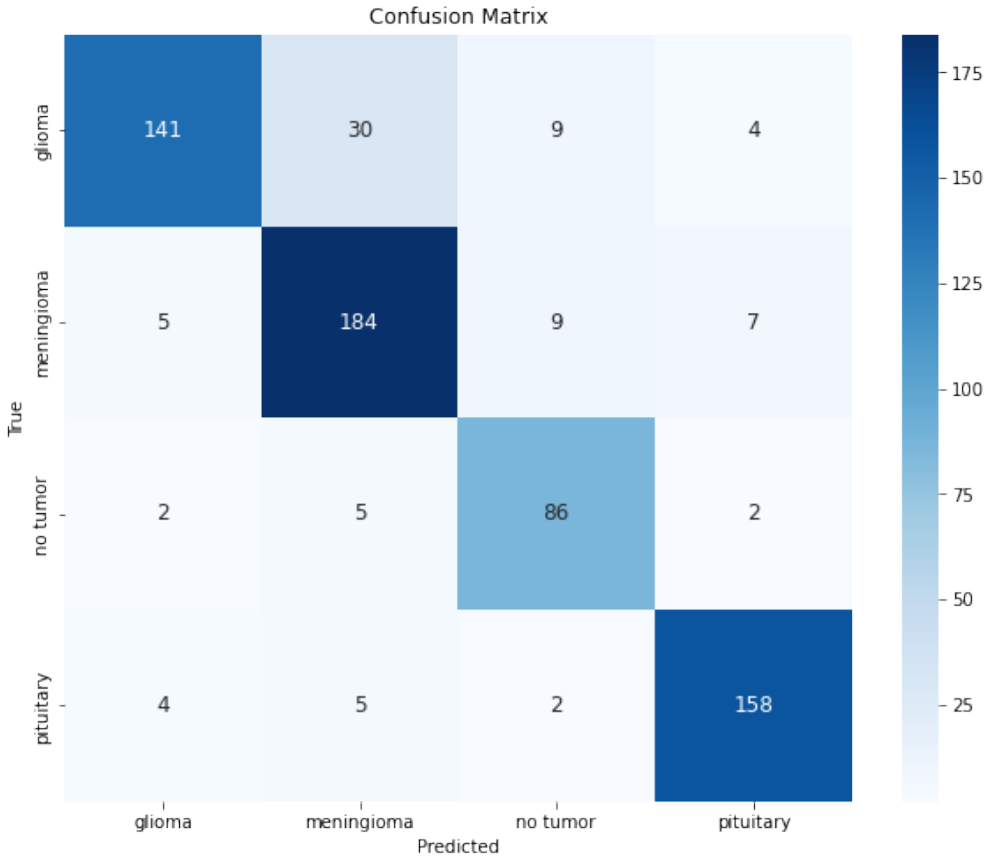

Figure 3.2: Random Forest Confusion Matrix

3.2 KNN

In statistics, k-nearest neighbor algorithm (k-NN) is classified as a non-parametric supervised learning method. In the k-NN classification process, the output result is determined based on the neighbors of the object. Specifically, the classification of an object is determined by a majority vote of the most common class among its k nearest neighbors as shown in Figure 3.3. [15]

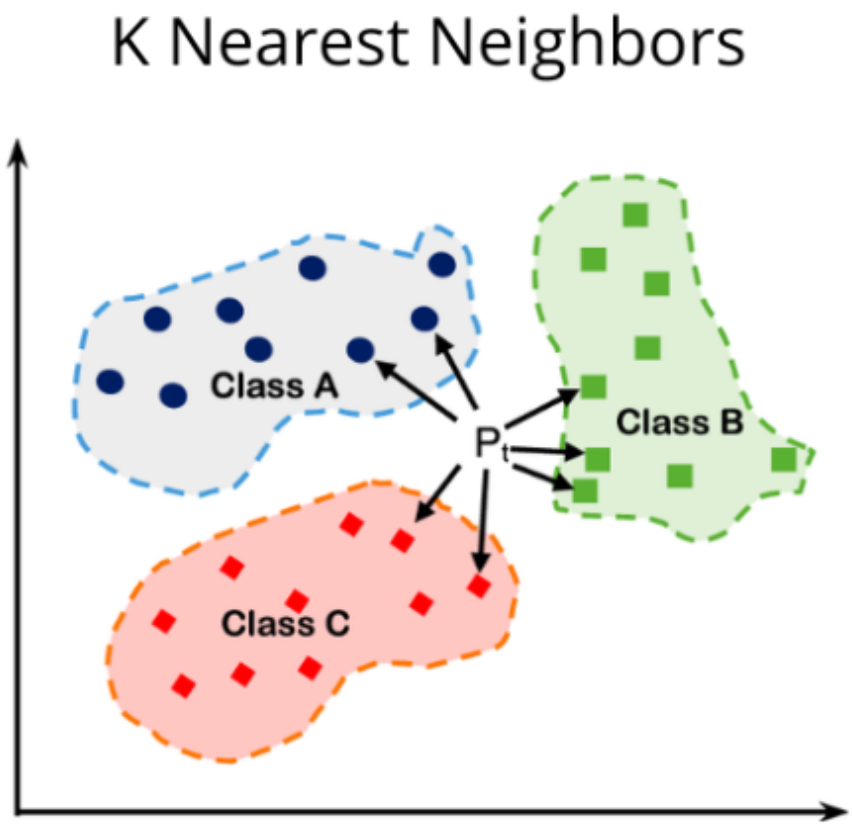

Figure 3.3: K-Nearest Neighbors Classifier [11]

In this project, cross-validation is used to find the best k value, and fold is set to 5. First, a K Neighbors Classifier is defined, and then candidate parameters for the k value are defined through GridSearchCV. The candidate parameters here are 1 to 10. Next, cross-validation

was used to select the best k value, and then a kNN model was built using the best k value. The result was that the best k value was 1. Then the object is simply classified into the category of its single nearest neighbor.

When k=1, k-NN may cause overfitting, especially when there are noise or outliers in the training data. This is because when k=1, the model is very sensitive to the training data and may use noise or outliers as the basis for decision-making. But considering that the model performs well on the test data, k=1 is still considered.

The prediction results are shown in Table 3.2. We can see that the f1 score of all categories is above 0.8, and the accuracy reaches 0.89. This shows that the k-NN model can effectively classify each class, and the overall performance is good. Among them, it is worth noting that the recall of the class Pituitary Tumor is 1.0. This shows that the model did not miss any real "pituitary tumor" samples in this class, that is, all samples in the "pituitary tumor" category were successfully detected, and no detections were missed. In areas such as medical diagnostics, ensuring that diseases are not missed is crucial. The k-NN model showed very high accuracy in identifying the pituitary tumor category.

Class	**Precision**	**Recall**	**F1 score**	**Support**
glioma tumor	0.86	0.86	0.86	184
meningioma tumor	0.89	0.87	0.88	205
no tumor	0.87	0.82	0.84	95
pituitary tumor	0.94	1.00	0.97	169
Accuracy			0.89	653

Table 3.2: k-NN Classification Report

The fact that the k-NN model does not identify the pituitary tumor into other classes can also be clearly seen in the confusion matrix. In Figure 3.4, the number of pictures in each category that are misidentified as another type is below 20. We can also intuitively see that, like the random forest model, glioma tumor and meningioma tumor are still two

categories that are easily confused.

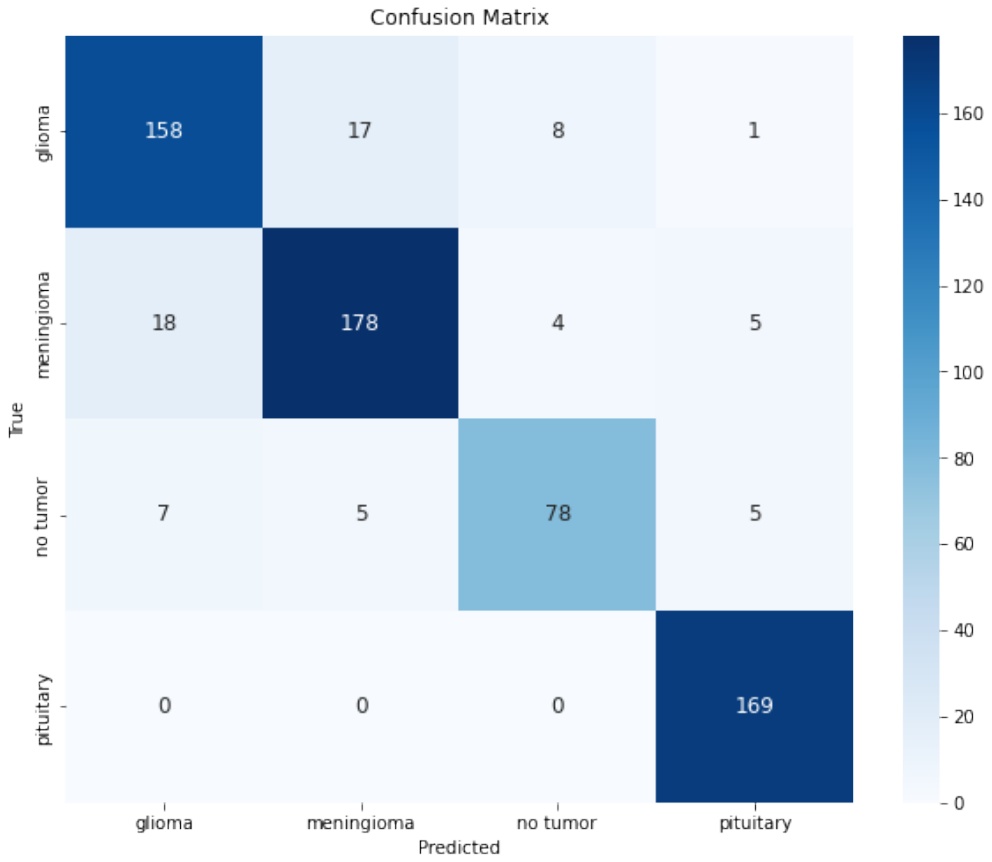

Figure 3.4: k-NN Confusion Matrix

3.3 SVM

Support Vector Machine (SVM) uses a subset of the training data set to define a decision boundary, which is used to classify samples. This subset is called a support vector. When a data set contains samples from two different categories, it can be determined by observing the distribution of the samples whether the data set is linearly separable, that is, whether there is a hyperplane such that all samples of the same category are located on the same side of the hyperplane. Although there may be many hyperplanes that all have zero classification

error on the training data, there is no guarantee that they will perform equally well on the test data set.

When selecting the optimal hyperplane, the goal of SVM is to make the edge of the decision boundary as large as possible. Decision boundaries with larger margins generalize better than decision boundaries with smaller margins. Larger edges can tolerate a greater degree of perturbation and therefore classify unknown samples more robustly. In contrast, a decision boundary with a small margin overfits the training data and therefore has poor generalization ability to unknown samples.

Assume that the training set contains m samples, each sample has n input features, where the category y_i of each sample is in {-1,1}.

The decision interface B can be represented as a hyperplane uniquely determined by the parameters (w, b), where w is the normal vector of the plane and b is the displacement of the plane. The two parallel edge interfaces B1 and B2 are parallel to the decision-making interface, and their displacements (also called function intervals) to the decision-making interface are both 1, that is, $|b - b_1| = |b - b_2| = 1$.

Then the edge of the decision interface is given by the distance between the two hyperplanes. The distance from a point on one plane to another plane is the distance between two parallel planes. If a sample point x is located above the decision interface, then $w^T x + b > 0$, the sample is marked as +1; if a sample point x is located below the decision interface, then $w^T x + b < 0$, the sample is marked as -1. The SVM classifier not only ensures accurate classification, but also maximizes the edge distance between the two class.

The main goal of the SVM classifier is to ensure the accuracy of classification and to increase the differences between categories as much as possible, making them easier to distinguish.

In this task of distinguishing images, an SVM classification is created through SVC (kernel='rbf'), where the parameter kernel='rbf' specifies the use of the Radial Basis Function

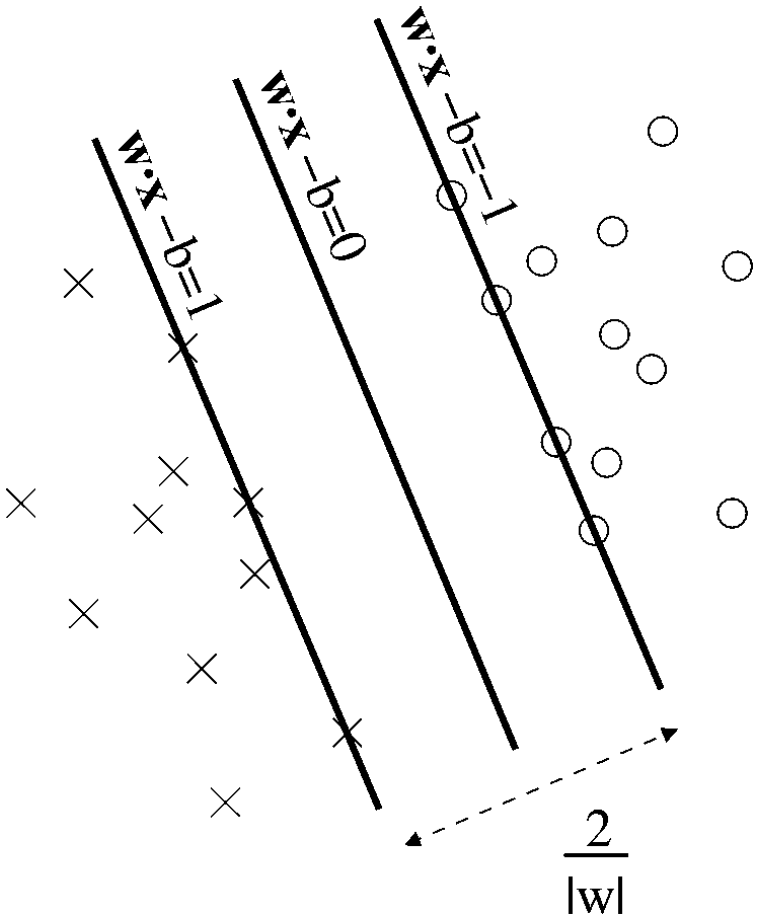

Figure 3.5: SVM Classifier

(RBF) as the kernel function.

According to the classification report of SVM, Table 3.3, the model performs best on the pituitary tumor category, with high precision, recall and F1 score. For the glioma tumor and meningioma tumor categories, the model's performance is average, with slightly lower precision and recall. This is consistent with the previous model results.

It can also be seen intuitively from the confusion matrix, Figure 3.6 that although the number of confusions for other types is very low, the number of confusions for glioma tumor and meningioma tumors is large, and the number of misclassifications is around 30.

Class	Precision	Recall	F1 score	Support
glioma tumor	0.79	0.73	0.76	184
meningioma tumor	0.77	0.78	0.77	205
no tumor	0.84	0.80	0.82	95
pituitary tumor	0.86	0.95	0.90	169
Accuracy			0.81	653

Table 3.3: SVM Classification Report

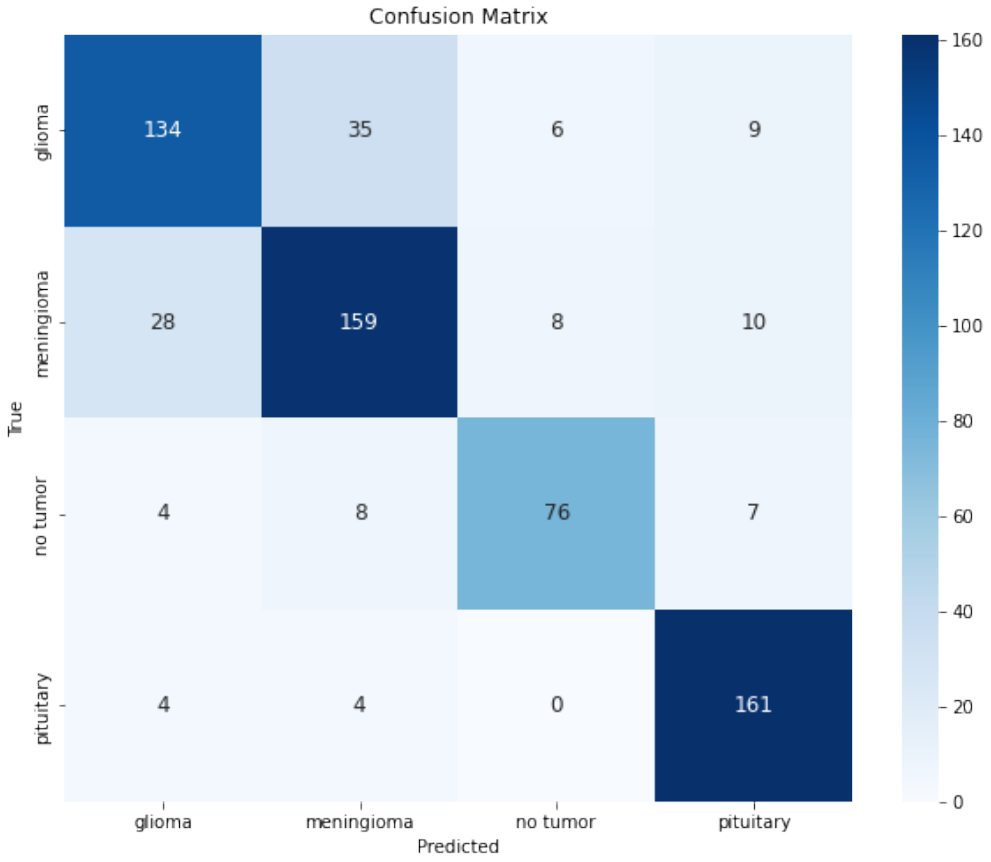

Figure 3.6: SVM Confusion Matrix

3.4 CNN

As a powerful tool, neural network is widely used image classification and recognition tasks. The neural network algorithm can automatically identify the characteristics of the image through training of image data, thereby classifying and identifying the image. Image recognition technology based on neural network algorithms has broad application prospects in image recognition, face recognition, object detection and other fields.

Among neural network algorithms, convolutional neural network (CNN) is the most widely used model. The CNN model extracts features in the image by using convolution kernels to perform convolution operations on the image. Convolutional neural network consists of input layer, hidden layer and output layer. The neural network's input layer receives image data, while the hidden layer extracts image features using operations such as convolution, pooling and activation functions, and the output layer classifies based on these features.

In the convolutional neural network, the convolution layer and the pooling layer are two very important parts. The convolutional layer can extract image features by using convolution kernels. The pooling layer can reduce the size of the feature map, thereby increasing the computing speed of the network.

The rectified linear activation function (ReLU) is a piecewise linear function characterized by the output being equal to the input when the input is not negative and zero when the input is negative. [10] The formula is :

$$f(x) = \max(0, x)$$

Due to its many advantages, ReLU has become the default activation function for many types of neural networks. Models using ReLU are easier to train and generally achieve better performance. [2]

3.4.1 Custom CNN

I used the convolutional neural network model, commonly used for image classification tasks. This model gradually extracts the features of the image through a series of convolution and pooling layers and performs classification through a fully connected layer. First, the convolutional layer extracts local features in the image, and each convolutional layer adds nonlinearity through the ReLU activation function. The subsequent pooling layer helps reduce the amount of data and retain important features. After multiple convolutions and pooling, the resulting feature map is flattened into a one-dimensional vector and passed to the fully connected layer. The fully connected layer performs classification tasks by learning complex relationships between features, and the final output layer outputs the probability distribution of each category through the Softmax activation function.

Layer	**Output Shape**	**Operation**
Conv2D	(None, 148, 148, 32)	3x3 convolution, ReLU activation
MaxPooling2D	(None, 74, 74, 32)	2x2 max pooling
Conv2D	(None, 72, 72, 64)	3x3 convolution, ReLU activation
MaxPooling2D	(None, 36, 36, 64)	2x2 max pooling
Conv2D	(None, 34, 34, 128)	3x3 convolution, ReLU activation
MaxPooling2D	(None, 17, 17, 128)	2x2 max pooling
Flatten	(None, 36992)	Flatten to 1D array
Dense	(None, 512)	Fully connected layer, ReLU activation
Dense	(None, 256)	Fully connected layer, ReLU activation
Dense	(None, 4)	Output layer, Softmax activation

Table 3.4: Summary of the layers in the Custom CNN model

We can see from Table 3.4 that since the convolution kernel can only move 1 pixel in each direction, the convolution kernel cannot completely cover the boundary pixels of the input image during the convolution operation. In this case, both the height and width of

the output feature map decrease by 1 pixel, resulting in a reduction of the output feature map size by 2 pixels.

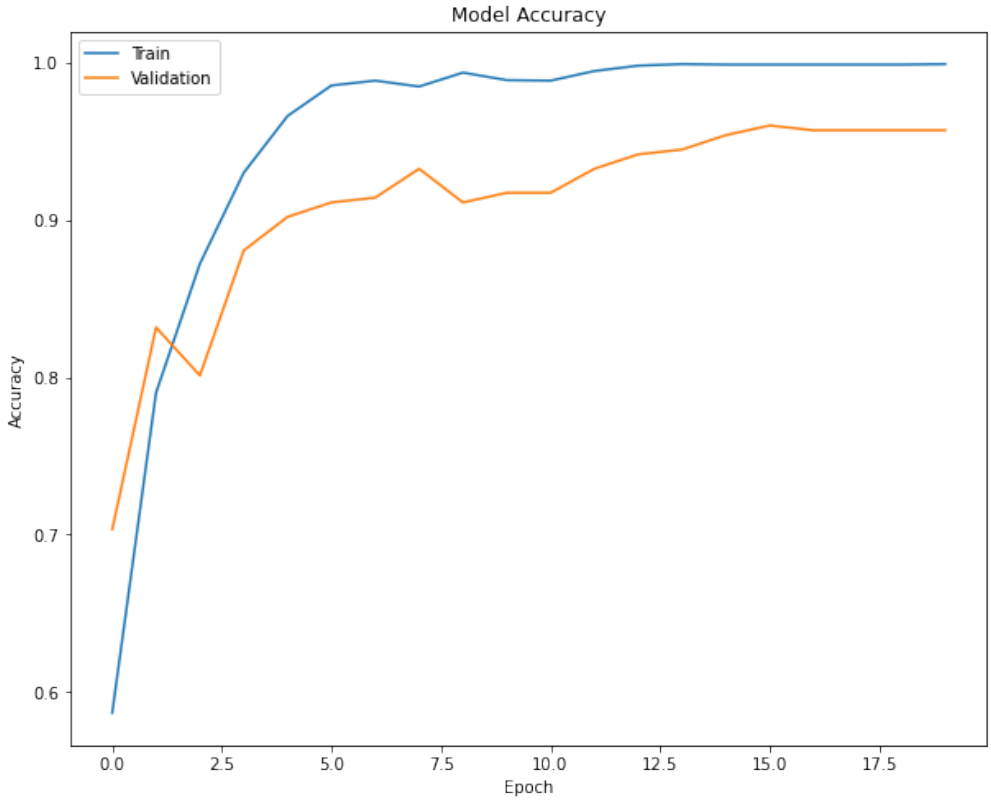

Figure 3.7: Custom CNN Accuracy

Figure 3.7 shows the model accuracy on training set and validation set. After the model was trained for several rounds of epochs, the training accuracy increased rapidly and stabilized at epoch 5, reaching about 0.98. This shows that the model learned a lot of information on the training data and fit the training data almost perfectly. The verification accuracy fluctuates in the early stage, and stabilizes when epoch is about 3, reaching about 0.9. The gap between validation accuracy and training accuracy indicates that the model is overfitting. It means that the model performs well on the training data, but its performance on unseen validation data may degrade. Therefore, although the training accuracy is high, the generalization ability of the model may be affected. The final test accuracy is 0.9848, which

is close to the training accuracy. This shows that the model also performs well on unknown test data, but it is still necessary to be wary of the reduction in generalization ability that may be caused by overfitting.

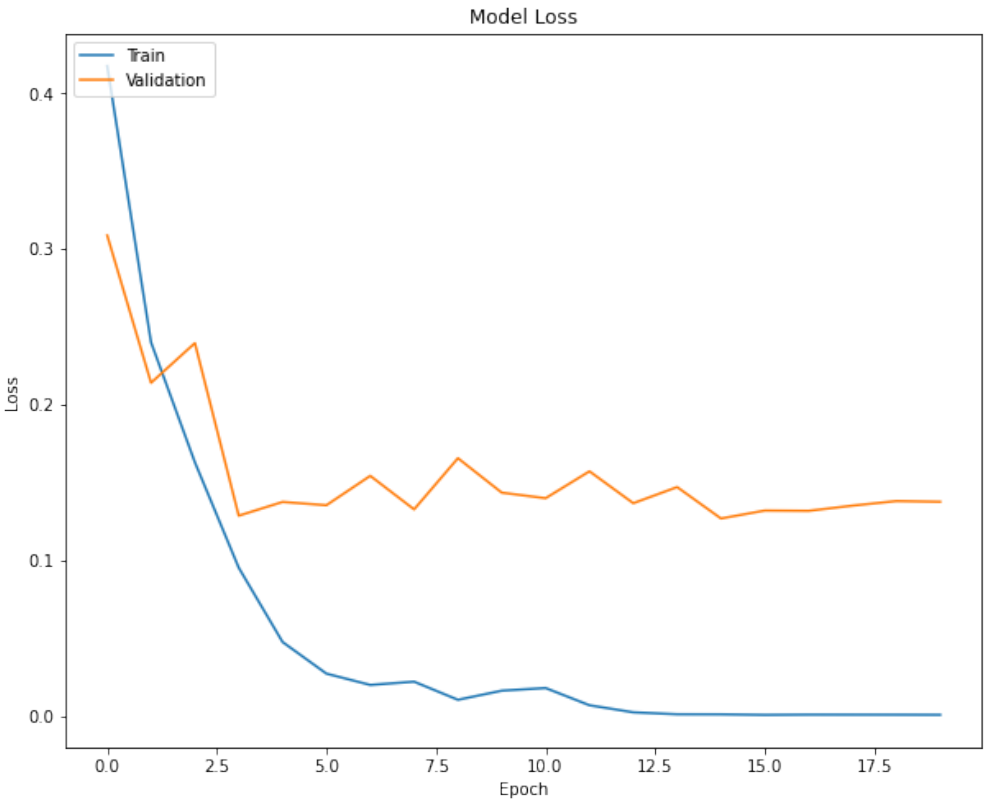

Figure 3.8: Custom CNN Loss

Loss is a measure of the difference between the model's predicted value and the true value. It represents the performance of the model during the training process and is usually used to measure the accuracy or error rate of model predictions. In this model, cross-entropy loss is used as the loss function to calculate the loss value of each sample and update the model parameters based on the average loss of all samples.

From Figure 3.8, we can see that the training loss drops rapidly in the first few epochs and stabilizes at around 0.02 when epoch equals 5. This indicates that the model fits the training data well. The validation loss also decreases in the first few epochs and levels off

when epoch equals 5, but the value fluctuates widely between 0.1 and 0.2. This fluctuation may indicate that the model's performance on the validation data is unstable and may be affected by noise in the data. The model's loss on the test data is 0.0288. Lower loss values indicate that the model generalizes well to unknown data. The test set results are shown in Table 3.5.

Test Result	
Accuracy	0.9848
Loss	0.0288

Table 3.5: Test Result for Custom CNN

3.4.2 CNN ResNet v2

Inception-ResNet-v2 is a deep neural network architecture that combines the Inception architecture and residual connections. The Inception architecture has been proven to show good performance with minimal computational resources. Combining residual connections with a conventional architecture allows the model to reach state-of-the-art performance levels, comparable to the latest generation Inception-v3 network. [13] Through research on the combination of Inception architecture and residual connections, the research results of Christian Szegedy et al. [13] have demonstrated that employing residual connections during training can notably hasten the training process of the Inception network. Compared to the equally expensive Inception network without residual connections, the residual Inception network shows better performance.

In this project, the Inception-ResNet-v2 model was first loaded, and a global average pooling layer and a fully connected layer were incorporated. Finally, an output layer is added to perform predictions on multi-classification tasks using a softmax activation function. The entire model is compiled through the Adam optimizer and cross-entropy loss function to optimize model parameters during training. In Table 3.6, we can see the operations in addition to the ResNet v2 model.

Layer	**Output Shape**	**Operation**
Input (ResNet)	(None, 150, 150, 3)	Input Image
GlobalAveragePooling2D	(None, 2048)	Global average pooling
Dense	(None, 60)	Dense layer, ReLU activation
Dense	(None, 60)	Dense layer, ReLU activation
Dense	(None, 4)	Output layer, Softmax activation

Table 3.6: Summary of the layers in the ResNet model

The training accuracy of ResNet-v2 CNN stabilizes slowly. After epoch is greater than 2, the accuracy slowly rises from 0.95 to about 0.99. Although the speed is slower, the final

training accuracy reaches a very high level, indicating that the model learns well on the training data. The validation accuracy fluctuates greatly. When epoch is less than 10, the validation accuracy basically does not exceed 0.9, and there are two obvious troughs, and the accuracy drops below 0.75. This indicates that the model generalizes poorly in the early stages of training. When epoch is greater than 10, the validation accuracy is between 0.9 and 0.95, but the fluctuation is still large. We can see that as epochs increase, the fluctuations in verification accuracy decrease. This shows that although the model's performance improves slightly in the later training stages, there is still a certain degree of instability.

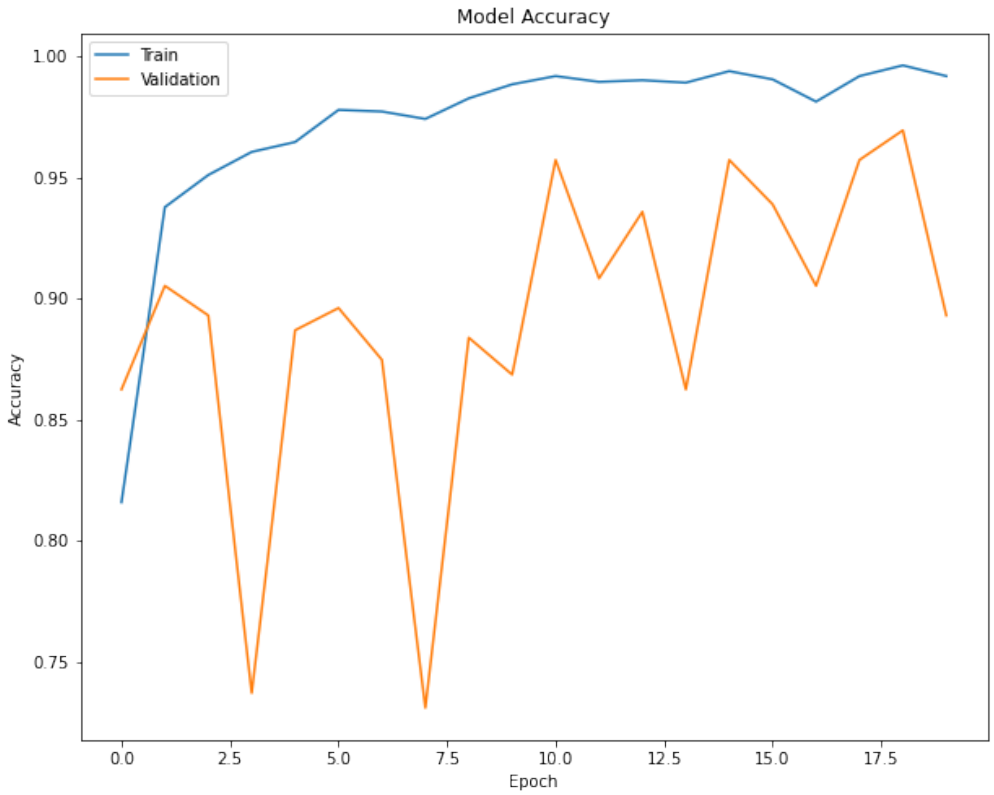

Figure 3.9: ResNet v2 Accuracy

From Figure 3.10, we can see that the train loss of ResNet CNN has always been low, always below 0.1, and the validation loss fluctuates greatly, generally between 0.1 and 0.4. When epoch is equal to 4, the validation loss is the highest, 0.3138. The training loss

remains consistently low, indicating that the model learns well on the training data and is able to fit the data effectively. However, the validation loss fluctuates greatly, especially reaching the highest value when epoch is equal to 4, which may mean that the model has poor generalization ability to the validation data at certain stages. Nonetheless, the final test loss is similar to the training loss, proving the robustness and effectiveness of the model. The test set results are shown in Table 3.7.

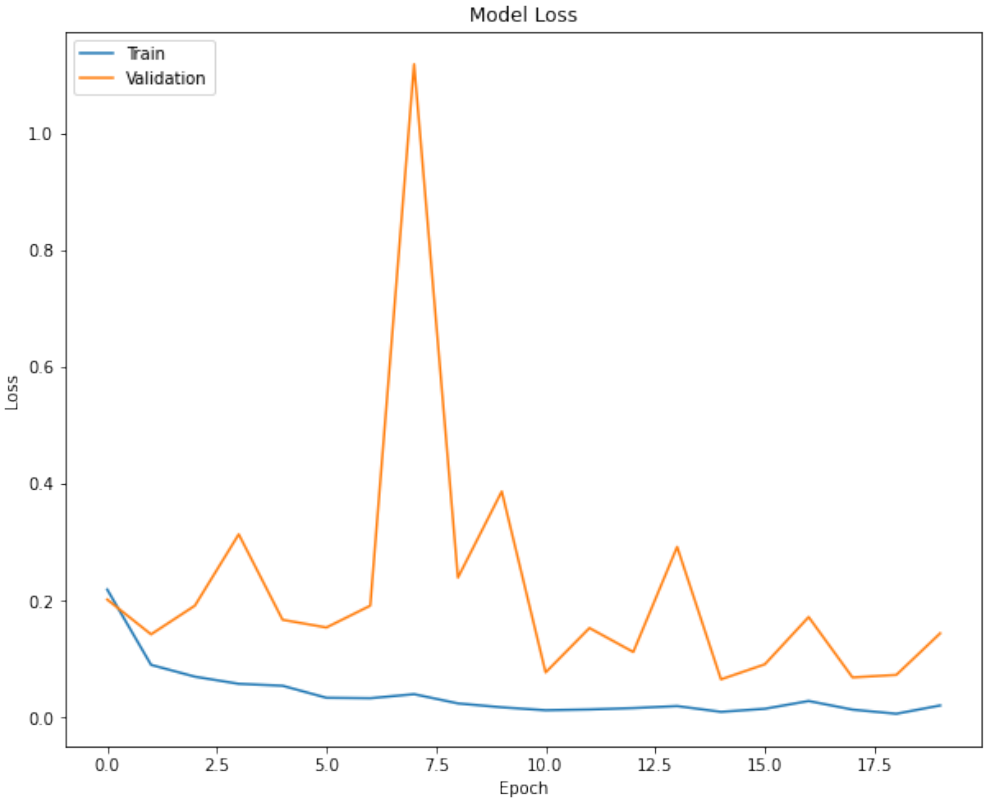

Figure 3.10: ResNet v2 Loss

Test Result	
Accuracy	0.9391
Loss	0.1311

Table 3.7: Test Result for ResNet v2

3.4.3 CNN VGG 16

The VGG network was originally proposed in 2014 by Karen Simonyan and Andrew Zisserman of the Visual Geometry Group of the Department of Engineering Science at the University of Oxford. They published a paper entitled "Very Deep Convolutional Networks for Large-Scale Image Recognition", demonstrating the performance of their model in object detection and classification, and won the first and second place in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). [12]

This project uses a neural network model based on the VGG 16 architecture. It first receives as input an image of size 150x150 pixels with 3 channels. Table 3.8 shows the overall layers. Then through the global average pooling layer and two fully connected layers, the ReLU activation function is used to gradually learn the complex features in the image. At the output layer, the Softmax activation function is applied to transform the network's raw output into a probability distribution for classification predictions.

The VGG 16 network is a relatively deep neural network with 16 convolutional and fully connected layers. Therefore, compared with the first two CNN models, the VGG 16 network may require more epochs to train. This model performs poorly when epoch is equal to 20, so here we set epoch equal to 50.

Layer	**Output Shape**	**Operation**
Input (VGG16)	(None, 150, 150, 3)	Input Image
GlobalAveragePooling2D	(None, 512)	Global average pooling
Dense	(None, 60)	Dense layer, ReLU activation
Dense	(None, 60)	Dense layer, ReLU activation
Dense	(None, 4)	Output layer, Softmax activation

Table 3.8: Summary of the layers in the VGG16 model

From Figure 3.11, as the epoch increases, the training accuracy gradually stabilizes at

a high level after epoch exceeds 20, floating from 0.9 to 0.99, suggesting that the model effectively learns from the training data. However, there are still large fluctuations in the verification accuracy. Although the verification accuracy fluctuates between 0.88 and 0.93, it shows a steady upward trend overall. The final test accuracy is 0.9873, which is very close to the highest value of training accuracy, indicating that the model performs well on unknown data and has good generalization ability. Although the model fluctuates greatly on the validation data, its performance on the test data proves its effectiveness and robustness.

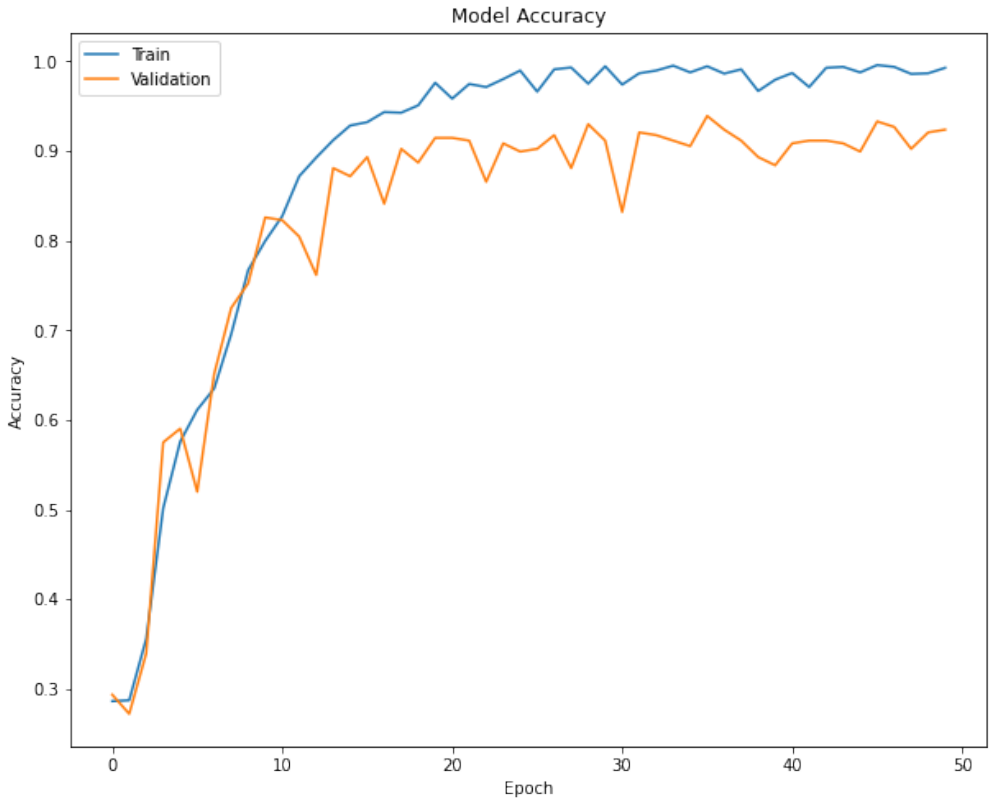

Figure 3.11: VGG 16 Accuracy

The train loss of VGG 16 has been below 0.1 after epoch is greater than 20. Finally, it reached a very low value of 0.0097 when epoch equals 50. Validation loss fluctuates greatly, and there is still a large fluctuation after epoch is greater than 20, generally between 0.1 and 0.4. The final test loss is 0.0215, which is similar to the training loss. The test set results

are shown in Table 3.9.

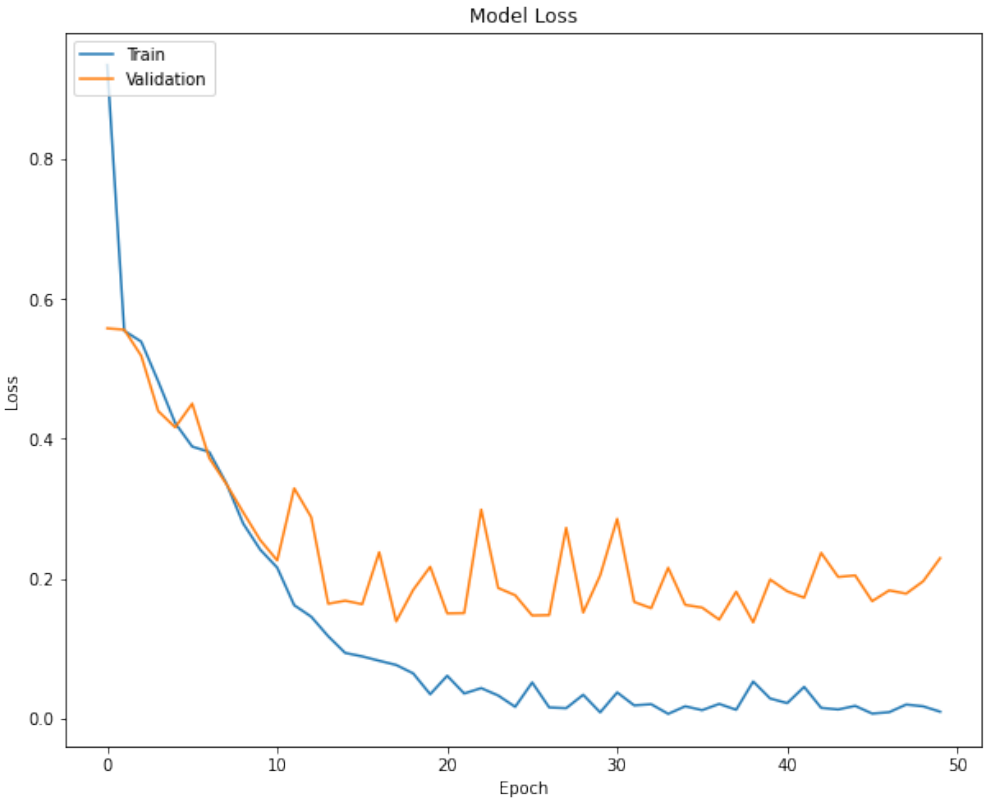

Figure 3.12: VGG 16 Loss

Test Result	
Accuracy	0.9873
Loss	0.0215

Table 3.9: Test Result for VGG 16

Evaluating the quality of a CNN model usually requires a comprehensive consideration of accuracy and loss. From the above three CNN models, the worst performance is ResNet v2. The accuracy of the test set is lower than 095, and the loss is greater than 0.1. In addition, it can be seen from the above pictures that for the ResNet v2 model, the training set performs better, but the performance on the validation set is unstable, which may indicate that the

model has poor generalization ability at this stage and cannot adapt to validation data. For the remaining two models, the test set accuracy is almost the same. By comparing losses, we can see that VGG 16 fits the test data better. However, the VGG 16 model has a deeper network structure, and it takes a longer time to train a VGG model. The VGG model performs well in image recognition tasks, but its long training time and high number of times have become an inconvenience in use.

CHAPTER 4

Discussion

Table 4.1 shows the accuracy of different models in classifying brain tumor MRI images.

Model	**Accuracy**
Random Forest	0.8716
K-NN	0.8928
SVM	0.8116
Custom CNN	0.9848
ResNet v2	0.9391
VGG 16	0.9873

Table 4.1: Result Comparison

This projext explores how machine learning can help analyze MRI images of brain tumors, thereby reducing the workload of medical staff. It compares several machine learning methods for processing MRI data and their ability to accurately classify tumors. Results show that convolutional neural networks (CNN), including Custom CNN, ResNet v2, and VGG 16, outperform other machine learning models in accurately classifying brain tumors based on MRI images. The accuracy of these CNN-based models surpasses traditional machine learning algorithms such as Random Forest, K-NN, and SVM. This highlights the superior ability of CNNs to extract meaningful features from MRI data and perform precise tumor classification. Custom CNN and VGG 16 achieved the highest accuracy (98.48% and 98.73%, respectively). Among them, VGG 16 successfully achieved good accuracy and minimal loss, highlighting its robustness and reliability in accurately classifying brain tumors

from MRI images. This suggests that deep learning models may provide promising improvements in brain tumor classification from MRI images. We can also see that the accuracy of convolutional neural network is high, which to a certain extent can assist medical staff in improving work efficiency and reducing interpretation times.

There is a lot of future work for this project. To further enhance machine learning in assisting MRI images in the assessment of brain tumors, more preprocessing methods can be explored to improve the efficiency of the model. The data used in this project is small. If a large amount of data needs to be processed, transfer learning technology can be used to improve model performance. By exploring these pathways, more accurate and reliable machine learning models can be developed to assist in the diagnosis of brain tumors using MRI images.

REFERENCES

[1] Sartaj Bhuvaji, Ankita Kadam, Prajakta Bhumkar, Sameer Dedge, and Swati Kanchan. Brain tumor classification (mri), 2020.

[2] Jason Brownlee. A gentle introduction to the rectified linear unit (relu), August 2020.

[3] S Deepak and PM Ameer. Automated categorization of brain tumor from mri using cnn features and svm. *Journal of Ambient Intelligence and Humanized Computing*, 12:8357–8369, 2021.

[4] Bradley J Erickson, Panagiotis Korfiatis, Zeynettin Akkus, and Timothy L Kline. Machine learning for medical imaging. *Radiographics : a review publication of the Radiological Society of North America, Inc*, 37(2):505–515, 2017.

[5] DC Febrianto, I Soesanti, and HA Nugroho. Convolutional neural network for brain tumor detection. In *IOP Conference Series: Materials Science and Engineering*, volume 771, page 012031. IOP Publishing, 2020.

[6] Ravikumar Gurusamy and Vijayan Subramaniam. A machine learning approach for mri brain tumor classification. *Computers, Materials and Continua*, 53(2):91–109, 2017.

[7] Farkhod Khushaktov. Introduction random forest classification by example. https://medium.com/@mrmaster907/introductionrandomforestclassificationbyexample-6983d95c7b91.

[8] A KOWSHIKA, M SHOBANA, and MS Kavitha. Advanced logistic regression for detecting the brain tumour cells. *NeuroQuantology*, 20(8):9090, 2022.

[9] Rohit Kundu. F1 score in machine learning: Intro & calculation, December 2022.

[10] Thi-Lam-Thuy LE. How does relu enable neural networks to approximate continuous nonlinear functions?, January 2024.

[11] Sachinsoni. K nearest neighbours — introduction to machine learning algorithms. https://medium.com/@sachinsoni600517/k-nearest-neighbours-introduction-to-machine-learning-algorithms-9dbc9d9fb3b2.

[12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[13] Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261, 2016.

[14] Javier E Villanueva-Meyer, Marc C Mabray, and Soonmee Cha. Current clinical brain tumor imaging. *Neurosurgery*, 81(3):397, 2017.

[15] Wikipedia. k-nearest neighbors algorithm. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm.

[16] Wikipedia. Random forest. https://en.wikipedia.org/wiki/Random_forest.