Brain Tumor Detection and Classification with CNN

Ruining Li

[1]New York University
rl4895@nyu.edu
the link to the code base is https://github.com/leeeexi/7123-final-brain-tumer-classification-CNN

Abstract

Brain tumor classification plays a crucial role in fthe diagnosis and treatment planning of brain cancer patients. In this project, we explore the application of convolutional neural networks(CNNs) for the automated classification of brain tumors using medical imaging data. The objectie is to develop a deep learning model capable of accurately distinguishing between differen types of brain tumors, thereby aiding radiologists in making timely and accurate diagnoses. The dataset used in this study consists of MRI images of the brain, and the dataset is divided into multiple categories, including glioma, meningioma, pituitary and notumor. We propse a CNN architecture and the experimental accuray of the trained model on test dataset is 92.91%

Introduction

Convolutional Neural Networks(CNNs) are a class of deep learning models primarily used for image recognition and computer vision tasks. CNNs are inspired by the organization and functioning of the human visual cortex, making them well-suited for tasks that involve processing visual data. At the core of CNNs are convolutional layers, which apply convolution operations to the input data. There layers consist of filters that slides across the input image, performing element-wise multiplications and summations to produce feature maps. The convolution operation helps the network learn hierarchical representations of the input data, capturing local patterns and spatial relationships. One of the key advantages of CNNs is their ability to automatically learn features from raw input data, eliminating the need for handcrafted feature engineering. In the brain tumor detection and classification project, CNN is used for both detection and classification of brain tumor MRI images. The basic model of a CNN is shown in Figure 1(upGrad, n.d.). The datset used in the project is a modification of BraTS. Instead of containing four stages of brain tumor MRI images, the dataset I used contain one stage of brain tumors which are divided into four main types: Glioma, Meningioma, Pituitary, and Notumor.

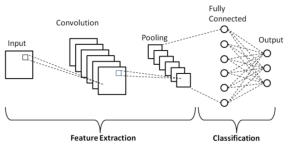

Figure 1: Basic model of CNN

Methodology

In this project, I made use of the combination of three datasets from Kaggle(Nickparvar,2021). The dataset consists of 7032 brain tumor MRI images divided into four classes: Giloma, Meningioma, Notumor, and Pituitary. 5712 images are used for training and 1311 images are used for testing. The dataset is preprocessed through OpenCV. Then data augmentation is done through ImageDataGenerator of Keras.

Image preprocessing

The input images are first downloaded with OpenCV imread function with their folder name as their labels. The labels are recorded in one list.The example image after preprocessing is shown in Figure 2. After downloading the image from their file path, some OpenCV functions are used to process the images, including:

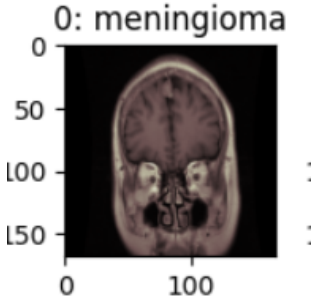

Figure 2: Eample

- BilateralFilter: The bilateralFilter is a function in OpenCV used for noise reduction while preserving edges in an image. It applies a bilateral filter to the input image, which means it smoothens the image by averaging the pixels in the neighborhood while considering both spatial distance and intensity difference.
- ApplyColorMap:The applyColorMap function is used to apply a colormap to a given grayscale image. It maps the single-channel input image to a 3-channel color image using a specified colormap
- Resize: The resize function in OpenCV is used to resize images to a specified width and height, or to scale images by a given factor. It is commonly used for image preprocessing, data augmentation, and preparing images for further processing or display.

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 168, 168, 32)	1568
max_pooling2d_3 (MaxPooling2D)	(None, 84, 84, 32)	0
dropout_4 (Dropout)	(None, 84, 84, 32)	0
conv2d_4 (Conv2D)	(None, 84, 84, 32)	16416
max_pooling2d_4 (MaxPooling2D)	(None, 42, 42, 32)	0
dropout_5 (Dropout)	(None, 42, 42, 32)	0
conv2d_5 (Conv2D)	(None, 42, 42, 64)	32832
max_pooling2d_5 (MaxPooling2D)	(None, 21, 21, 64)	0
dropout_6 (Dropout)	(None, 21, 21, 64)	0
flatten_1 (Flatten)	(None, 28224)	0
dense_2 (Dense)	(None, 128)	3612800
dropout_7 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 4)	516

Total params: 3664132 (13.98 MB)
Trainable params: 3664132 (13.98 MB)
Non-trainable params: 0 (0.00 Byte)

Figure 3: Model summary

Data Augmentation

I chose ImageDataGenerator module for data augmentation. The ImageDataGenerator is a utility in Keras, often used with TensorFlow, for generating batches of augmented image data. It allows for real-time data augmentation during model training, which can help improve the generalization and performance of deep learning models, particularly for tasks like image classification, object detection, and segmentation. It provides various options for data augmentation, including rotation, shifting, flipping, zooming, shearing, and more. This helps increase the diversity of the training dataset, reducing overfitting and improving model generalization. The transform chosen for data augmentation includes:

- Rotation: The rotation transformation in ImageDataGenerator rotates images by a specified angle. The parameter is:rotation_range: Specifies the range of random rotation applied to the image. It can be a single integer or a tuple (min_angle, max_angle), indicating the minimum and maximum rotation angles in degrees.
- Width/Height shift: The width/height shift transformation in ImageDataGenerator shifts images horizontally and vertically by a fraction of their width and height, respectively. The parameter width(height)_shift_range: Specifies the range of random horizontal shift as a fraction of the total width(height).
- Zoom: The zoom transformation in ImageDataGenerator zooms into or out of images by a random factor. The parameter zoom_range: Specifies the range of random zoom. If a single float, [1-zoom_range, 1+zoom_range] is used as the zoom range. If a tuple, it represents the lower and upper bound of the zoom range.
- Horizontal flip: The horizontal flip transformation in ImageDataGenerator flips images horizontally with a certain probability. The parameter horizontal_flip: Specifies whether to randomly flip images horizontally. Default is False

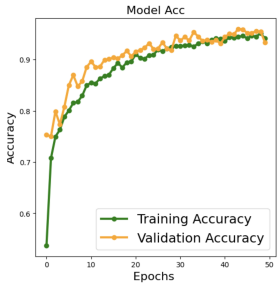

Figure 4: Accuracy curve

Architecture

The model I used is a convolutional neural network(CNN). In its core part, the be-ginning of the model is an initial convolutional layer, followed by a maxpooling layer with pool size of 2x2 to reduce the spatial dimensions of the feature maps while retaining the most important information. In order to maintain the simplicity and effectiveness of the model, ReLU activation is chosen as the activation layer. The flatten layer is used to convert the 2D features maps from the last max-pooling layer into a 1D vector, and fed into the fully connected layer for classification. The last layer is the fully connected layer and it performs classification based on the features extracted by the convolutional layers. The model in Figure 3 shows the summary of the architecture of the model.

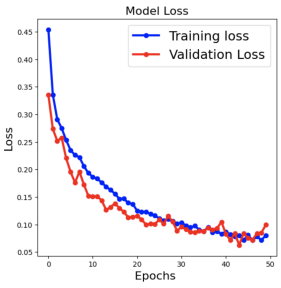

Figure 5: Loss curve

Result

The model is trained with 5712 images from the dataset, and the validation set contains 1143 images. The accuracy of the

training curve is about 95%. The training curse is in Figure 4 and Figure 5. The evaluation of the model with the test dataset shows an accuracy of 92.91%. In Figure 6 and 7, there are confusion matrix and the classification report.

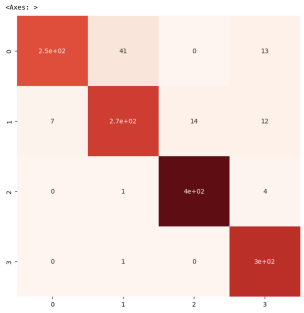

Figure 6: confusion matrix

```
Metrics for each class:
Class: glioma
Precision: 0.9723
Recall: 0.8200
F1-Score: 0.8897

Class: meningioma
Precision: 0.8639
Recall: 0.8922
F1-Score: 0.8778

Class: notumor
Precision: 0.9662
Recall: 0.9877
F1-Score: 0.9768

Class: pituitary
Precision: 0.9116
Recall: 0.9967
F1-Score: 0.9522

Overall Accuracy: 0.9291
```

Figure 7: Classfication report

References

Brain Tumor MRI Dataset(n.d.). Nickparvar, M. https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset

upGrad. (n.d.). Basic CNN Architecture. Retrieved from https://www.upgrad.com/blog/basic-cnn-architecture/