CS 512 Final Project

Brain Tumor Classification Using Convolutional Neural Networks

Illinois Institute of Technology, Chicago
Department of Computer Science

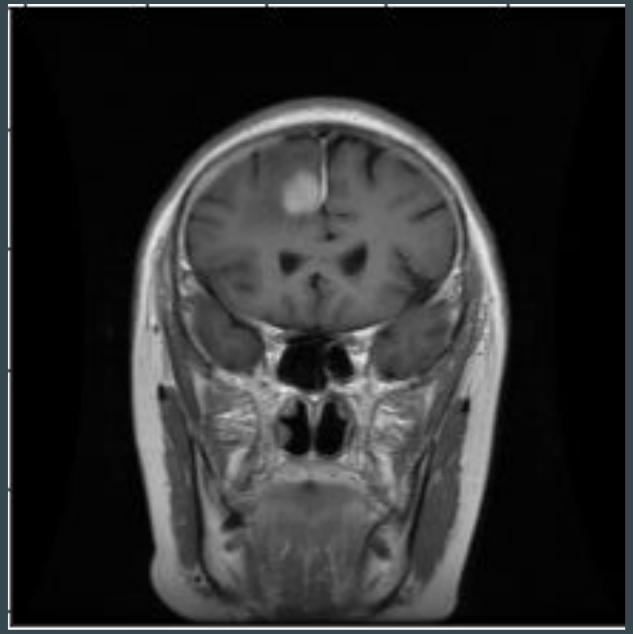
Kinjal Kachi
A20449343

Omkar Pawar
A20448802

Problem Statement

In this project we try to create a neural network model which can be used to predict the class of the Magnetic Resonance Imaging (MRI) scan if there exists any of the tumor mentioned below

- Meningioma
- Glioma
- Pituitary tumor.



Data Definition

The dataset that we will be using for our project is a brain tumor dataset posted by Jun Cheng on figshare.com. This data is organized in matlab data format (.mat file).
Dataset Consists of 3064 T1 Weighted contrast-enhanced images from 233 patients with three kinds of a brain tumor

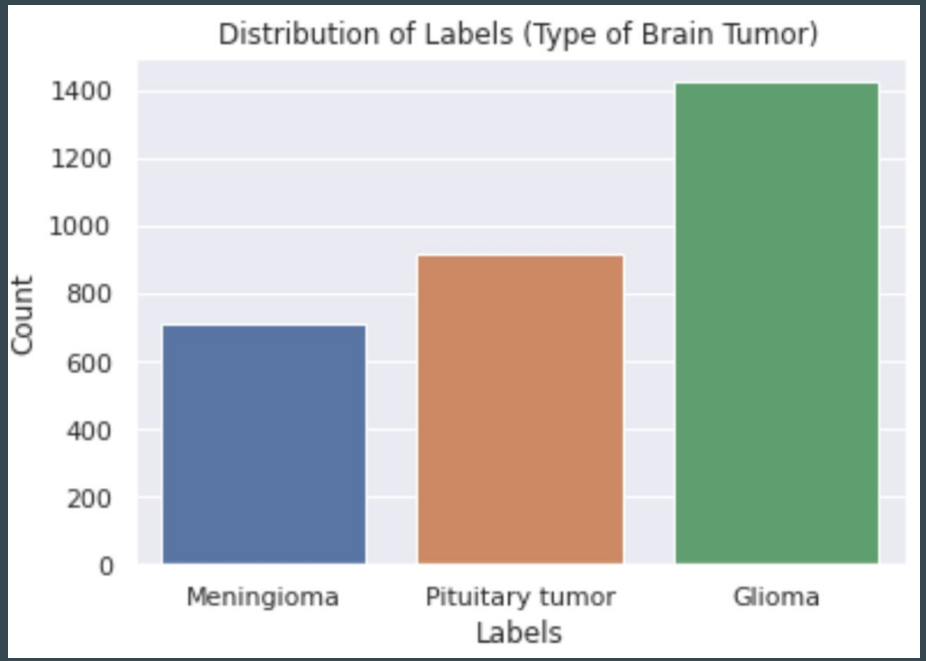
1) Meningioma (708 slices)
2) Glioma (1426 slices)
3) Pituitary tumour (930 slices).

Region of Interests (3 types of tumors) are segmented and provided in .mat file format as a part of the database.

Data Preprocessing

- Transform Data to Python List
- Seperate images and their respective labels
- View distribution of labels
- Use Stratified Sampling to split the data into training and validation sets
- Save these sets to pickle file for further use.

Distribution of Class Labels

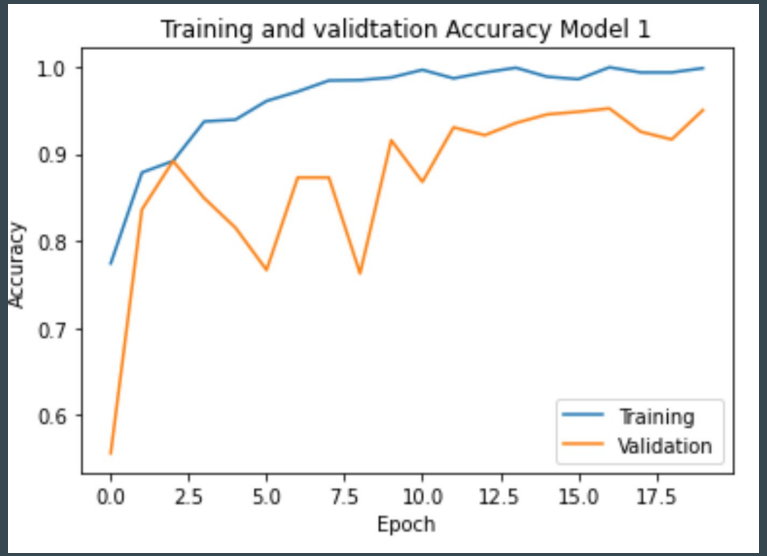


Data Modeling

Model 1

Defined a convolutional model that has the architecture as follows

- 2 Blocks of : Convolution -> Max Pooling -> Batch Normalization
- Dropout Layer with a factor of 0.3
- Flatten the convoluted image
- Dense Layer -> Output Layer



Model compiling and training parameters

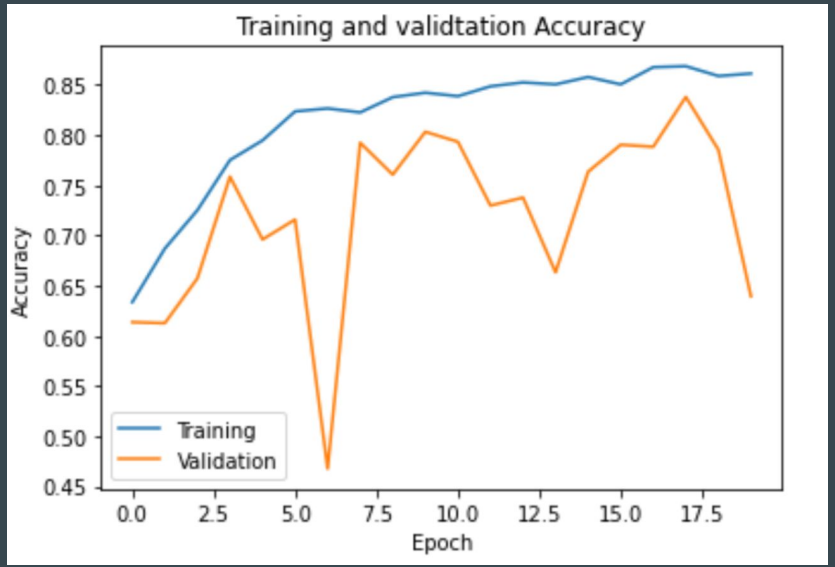
- Activation Function = reLu, softmax
- Loss Function = Categorical Crossentropy

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_2 (Conv2D)            (None, 254, 254, 64)      1664
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 127, 127, 64)      0
_________________________________________________________________
batch_normalization_2 (Batch (None, 127, 127, 64)      256
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 63, 63, 64)        36928
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 31, 31, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 31, 31, 64)        0
_________________________________________________________________
batch_normalization_3 (Batch (None, 31, 31, 64)        256
_________________________________________________________________
flatten_1 (Flatten)          (None, 61504)             0
_________________________________________________________________
dense_2 (Dense)              (None, 64)                3936320
_________________________________________________________________
dense_3 (Dense)              (None, 3)                 195
=================================================================
Total params: 3,975,619
Trainable params: 3,975,363
Non-trainable params: 256
```

Data Modeling

Model 2: Using Data Augmentation

- Same architecture as the previous model
- Implemented Data Augmentation at runtime using ImageGenerator from Keras
- Random flips(horizontal and vertical) and rotation of the image (by a factor of 90)
- Does not perform well.



Transfer Learning

ResNet 50

- Use the weights of already trained models and just change the input and output layers to make it work for our problem.
- ResNet 50 is a deep convolutional neural network used for image classification
- We freeze some layers so that the weights are not updates which reduces training time.

Results

Metric	Model 1	With Data Augmentation	ResNet 50
Training Accuracy	0.995	0.8688	0.9549
Training Loss	0.0048	0.4140	0.4828
Validation Accuracy	0.9523	0.8381	0.8689
Validation Loss	0.1754	0.4877	1.1077
No. of Epochs	20	20	10

Thank You