

## **ABSTRACT**

PANCHOLIA, RONIL. Tackling the Challenges in Pediatric Brain Tumor Classification with Deep Learning. (Under the direction of Dr. Nagiza Samatova).

This research attempts to tackle the challenges in brain tumor classification for patients with pediatric cancer. This work proposes computer vision based deep learning architectures to classify the type of tumor in pediatric brain cancer patients. The proposed methodology is based on supervised learning algorithms for classification.

Pediatric cancer can be analyzed in a non-invasive way using Magnetic Resonance Imaging (MRI) images. These images serve as a 3D reconstruction of the brain. The goal of this work is to exploit advances in AI and deep learning to do feature extraction from these MRI scans to distinguish between different types of brain tumors.

In this research, we have explored different approaches to use the MRI scans as inputs to convolutional neural networks to perform the task of tumor classification. An analysis of the pros and cons of each of the proposed architectures is done based on experimental results.

© Copyright 2020 by Ronil Pancholia

All Rights Reserved

# Tackling the Challenges in Pediatric Brain Tumor Classification with Deep Learning

by  
Ronil Pancholia

A dissertation submitted to the Graduate Faculty of  
North Carolina State University  
in partial fulfillment of the  
requirements for the Degree of  
Master of Science

Computer Science

Raleigh, North Carolina  
2020

APPROVED BY:

---

Dr. Kemafor Ogan

---

Dr. Raju Vatsavai

---

Dr. Nagiza Samatova  
Chair of Advisory Committee

## **DEDICATION**

This work is dedicated to my parents for their unwavering love and support.

## **ACKNOWLEDGEMENTS**

I would like to thank my advisor Dr. Nagiza Samatova for her guidance in this dissertation. Her motivation, knowledge, and passion for this project helped me have a great experience in my first academic research. Dr. Samatova also helped me connect with doctors working directly on the problem of tumor classification and I really appreciate all the help I received from Dr. Kristen Yeom and her research team, Katie Shpanskaya, Edward Lee, Lily Kim, and Michelle Han who are domain experts in the field.

I would also like to thank my lab colleagues Yifan Zhao, Parth Nagori, Aman Chauhan, and Anshul Atreik for their help and support. They helped me with insightful discussions related to the research and provided invaluable suggestions.

Finally, I would like to thank my friends and family for their support without which this work would not have been possible.

# TABLE OF CONTENTS

<b>List of Tables</b> . . . . .	<b>vi</b>
<b>List of Figures</b> . . . . .	<b>vii</b>
<b>Chapter 1 INTRODUCTION</b> . . . . .	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Problem Statement . . . . .	4
1.3 Medical Background . . . . .	4
1.3.1 Definitions and Terminology . . . . .	5
1.3.2 Modalities . . . . .	5
1.3.3 Axes . . . . .	6
1.4 Machine Learning Background . . . . .	6
1.4.1 Supervised Learning . . . . .	7
1.4.2 Neural Networks and Deep Learning . . . . .	8
1.4.3 Softmax Classification and Cross-Entropy Loss . . . . .	8
1.4.4 Convolutional Neural Networks . . . . .	8
1.4.5 AlexNet . . . . .	10
1.4.6 Residual Networks . . . . .	10
1.4.7 Inception Network . . . . .	11
1.4.8 Dropout Regularization for Deep Neural Networks . . . . .	13
1.4.9 Transfer Learning . . . . .	13
<b>Chapter 2 METHODOLOGY</b> . . . . .	<b>14</b>
2.1 Heuristics based Slice Analysis . . . . .	15
2.2 2D Methods . . . . .	16
2.2.1 Per Slice Classification . . . . .	17
2.2.2 Ensemble Per Slice Classification . . . . .	18
2.3 3D Methods . . . . .	18
2.3.1 3D Block Classification . . . . .	20
2.3.2 Sampling Slices for 3D Classification . . . . .	21
2.3.3 Multi-Slice architecture using shared weights . . . . .	22
2.4 Dealing with Class Imbalance . . . . .	25
2.4.1 Delayed Back-Propagation . . . . .	25
2.4.2 Loss Re-Weighting . . . . .	26
2.5 How to Combine Multi-Modal information? . . . . .	26
<b>Chapter 3 RESULTS</b> . . . . .	<b>29</b>
3.1 Pediatric Tumor Dataset . . . . .	29
3.1.1 Dicom File Format . . . . .	30
3.1.2 Brain Tumor Classes . . . . .	31
3.2 Pediatric Brain Age Dataset . . . . .	31

3.3	Comparison of Methods . . . . .	32
3.3.1	Ensemble 2D Slice Classification . . . . .	32
3.3.2	3D Block Classification . . . . .	33
3.3.3	Multi-Slice Shared Weights Classification . . . . .	33
<b>Chapter 4</b>	<b>RELATED WORK . . . . .</b>	<b>35</b>
4.1	BRATS Challenge . . . . .	35
4.2	Deep Learning on MRI datasets . . . . .	37
<b>Chapter 5</b>	<b>CONCLUSION AND FUTURE WORK . . . . .</b>	<b>39</b>
5.1	Conclusions . . . . .	39
5.2	Future Work . . . . .	40
<b>References</b>	<b>. . . . .</b>	<b>42</b>

## LIST OF TABLES

Table 2.1	2D slice classification model. The CNN backbone can be any feature extractor such as AlexNet, ResNet, VGG etc. . . . .	18
Table 2.2	3D Block Classification Model. . . . .	20
Table 2.3	Multi-Slice Shared Weights Model. . . . .	25
Table 3.1	MRI machine manufacturers and the intensity values for the two different models: 1.5T and 3T. . . . .	30
Table 3.2	Some of the fields in a DICOM. . . . .	30
Table 3.3	Table showing distribution of number of patients with different tumor classes in T2-weighted modality. . . . .	31
Table 3.4	Ensemble slice classification results for specific CNN backbone architectures using either max-confidence or majority vote. . . . .	32
Table 3.5	3D Block Classification results with blocks of different depths. . . . .	33
Table 3.6	Results for different approaches of combining features for Multi-Slice Shared Weights Model. . . . .	34
Table 3.7	Most optimal results for all of the methods discussed in this work. . .	34

## LIST OF FIGURES

Figure 1.1	Google Search trends for the query "Artificial Intelligence." Y-axis represents the interest, where 100 means the maximum interest. . . .	2
Figure 1.2	Google Search trends for the query "Deep Learning." Y-axis represents the interest, where 100 means the maximum interest. . . . .	3
Figure 1.3	The three different axes Axial, Coronal, and Sagittal. Image Source: <a href="https://casemed.case.edu/clerkships/neurology/">https://casemed.case.edu/clerkships/neurology/</a> . . . . .	6
Figure 1.4	Sample MRI scans of a brain in the three different axes - Axial, Coronal and Sagittal in order. Source: <a href="https://radiologykey.com/brain/">https://radiologykey.com/brain/</a> . . . . .	7
Figure 1.5	AlexNet model architecture. Source: (?) . . . . .	10
Figure 1.6	A residual block with a skip-connection. Source: (He et al. 2016). . . .	11
Figure 1.7	A dense network with five layers. Source: (Huang et al. 2017). . . . .	12
Figure 1.8	An Inception block. Source: (Szegedy et al. 2015). . . . .	12
Figure 2.1	Number of tumor slices at every index for DIPG patients. . . . .	15
Figure 2.2	A schematic diagram of the 2D model architecture. . . . .	17
Figure 2.3	Convolution neural network architecture for transfer learning with age regression. . . . .	19
Figure 2.4	3D MRI block convolution neural network architecture. . . . .	21
Figure 2.5	Left: Building block of a residual function, Right: Building block of a residual function with a linear bottleneck. . . . .	22
Figure 2.6	Multi-Slice architecture with convolutional neural network based feature extractors that share weights. . . . .	23
Figure 2.7	Summing up features A and B. . . . .	27
Figure 2.8	Concatenating features A and B. . . . .	27
Figure 2.9	Bilinear Fusion for features A and B. . . . .	28
Figure 4.1	Schematic visualization of the network architecture used in 3D MRI Brain Tumor Segmentation Using Autoencoder Regularization, (Myronenko 2018). . . . .	36
Figure 4.2	U-Net model architecture. . . . .	37
Figure 4.3	TwoPathCNN architecture proposed in Havaei et al. (2017). . . . .	38
Figure 5.1	Number of tumor slices at every index for patients with different tumor types. . . . .	41

# CHAPTER

## 1

# INTRODUCTION

## 1.1 Motivation

Artificial Intelligence (AI) is one of the emerging fields of Computer Science. This can be seen in Figure 1.1, that shows the increasing interest overtime on Google search queries for the term "Artificial Intelligence." Deep Learning is a field of AI and machine learning that deals with learning algorithms inspired by the working of a human brain. These learning algorithms are called artificial neural networks and are trained using the back-propagation algorithm. Convolutional Neural Networks are designed to work well with images. It has been shown that convolutional neural networks outperform many traditional feature-based machine learning algorithms for problems where the inputs are images (Alom et al. 2018). Convolutional neural networks were first introduced in 1998 (LeCun et al. 2001) and have been around for almost two decades now. However, only with the recent advances in GPU hardware and the introduction of the ReLU activation function (Krizhevsky et al. 2012), deep learning has started to take off. This can be seen in figure 1.2 that shows a spike in interest on Google search queries for the term "Deep Learning."

Specifically, the work from (Lee et al. 2017) provides an excellent overview of the appli-

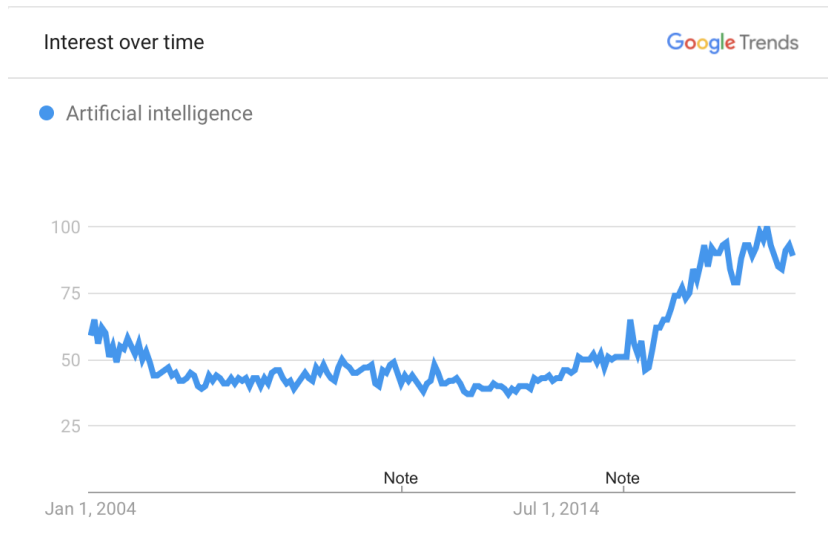


Figure 1.1: Google Search trends for the query "Artificial Intelligence." Y-axis represents the interest, where 100 means the maximum interest.

cations of AI and deep learning in medical imaging. This survey also compares traditional machine learning approaches to deep learning approaches in the context of medical imaging. As mentioned in this survey, convolutional neural networks have been proven to be superior over traditional machine learning approaches for medical imaging, especially, in image segmentation and registration for cancer tumor diagnosis.

Pediatric Cancer is cancer that usually occurs in children sometime between birth until they reach the age of 15 years. Brain cancers account for about 15% of pediatric cancers and are the second most common type of cancer in children, after leukemia. The most common brain tumors are of the following four types: Medulloblastoma, Diffuse Intrinsic Pontine Glioma, Ependymoma, Pilocytic. Each of these tumor types have different treatment strategies and prognosis. Being able to identify the type of tumor without the need for surgery is extremely valuable. Magnetic Resonance Imaging (MRI) images can be used for cancer diagnosis. However, it is only possible to do so after decades of medical training. An automated way of identifying the tumors would be extremely useful.

MRI is a non-invasive way that helps doctors to map the brain of a cancer patient in a 3D image, that can be used for further diagnosis. This work focuses on proposing deep learning techniques to address the issues with image classification on MRI images.

Another application of classification using MRI data surfaces when doctors want to develop drugs for such cancers. Doctors have to sample patients for a study, and this

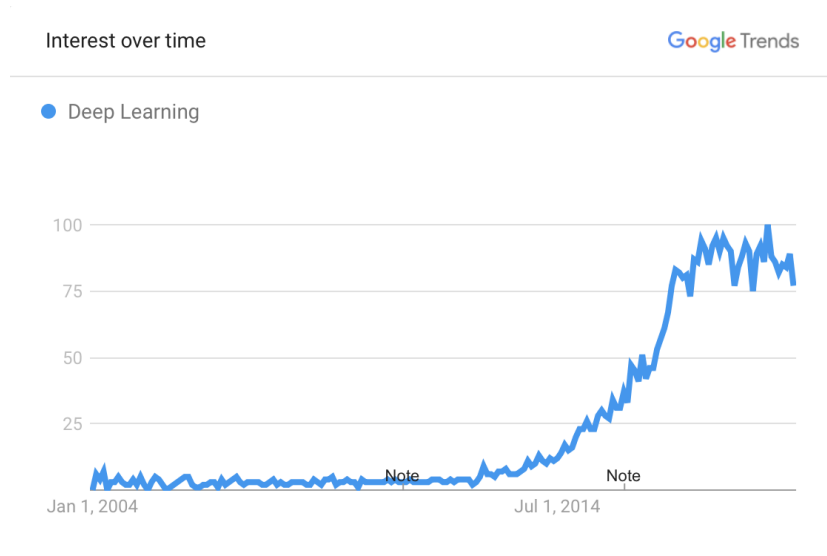


Figure 1.2: Google Search trends for the query "Deep Learning." Y-axis represents the interest, where 100 means the maximum interest.

becomes a very hard task when dealing with children. They typically like to have a series of tests and experiments on a patient and analyze the change in the patient's condition. Different stages of cancer have a different life expectancy. If life expectancy can be classified by looking at the MRI scans, the doctors can make an informed decision while choosing a subject for study. An automated deep learning model for classification using MRI scans as input can help make these decisions.

The main contributions of this dissertation are the following:

1. Identified effective ways to handle 3D MRI data for pediatric brain tumor patients.
2. Exploited the advances in AI and deep learning to do feature extraction from MRI data with the goal of brain tumor classification.
3. Proposed different deep learning architectures for brain tumor classification.
4. Analyzed the pros and cons of each of the proposed architectures based on experimental results.

## 1.2 Problem Statement

Deep learning algorithms have proven to be very successful in solving detection tasks on real-world images (Druzhkov and Kustikova 2016). The goal of this dissertation is to propose deep neural network architectures that can input Magnetic Resonance Imaging (MRI) images to identify the type of tumor in a child's brain. This study deals with the classification of MRI scans into one of the four rare pediatric tumor types:

1. Medulloblastoma (MB)
2. Diffuse Intrinsic Pontine Glioma (DIPG)
3. Ependymoma (EP)
4. Pilocytic (PILO)

Some of the main challenges in brain tumor classification as follows:

1. Supervised learning algorithms require labeled data that is expensive to collect. There is a severe lack of pediatric brain tumor MRI data. Deep learning algorithms on real world images work with millions of training examples as compared to a few hundreds in brain MRI.
2. MRI data is in 3D and has multiple modalities. This makes feature extraction very difficult and computationally expensive.
3. Some of the tumors occur very rarely; this leads to an imbalance in the dataset.
4. MRI technology has evolved over the years. The datasets were collected over a period of ten years. There are various types of machines from different manufacturers. The MRI camera resolution is usually different for different machines. The datasets are collected from different hospitals, this adds to the variability.

## 1.3 Medical Background

The following section provides an overview of some basic definitions and terminology to get an understanding of the medical background required for this work.

### **1.3.1 Definitions and Terminology**

#### **Magnetic Resonance Imaging**

Magnetic Resonance Imaging (or MRI) (Liang and Lauterbur 2000) is a common procedure of scanning internal organs like brain, knees, etc. It is based on the magnetization properties of atomic nuclei. A series of radio frequency pulses are introduced to the tissue being examined. These radio frequency pulses are perturbed and different responses are collected.

#### **Relaxation Times**

The tissue being examined can have different relaxation times (LeBlanc et al. 2000). The relaxation times are categorized as the following:

1. Longitudinal Relaxation Time (or T1)
2. Transverse Relaxation Time (or T2)

Longitudinal Relaxation Time (or T1) is defined as the time taken by a spinning proton to re-align with an external magnetic field. Transverse Relaxation Time (or T2) is defined as the time taken by a spinning proton to lose coherence with an external magnetic field.

#### **Magnetic Resonance Imaging Sequences**

To scan a brain, a camera moves in front of a brain and takes a "picture" after every few millimeters. These pictures, when combined in the correct order, can give us a 3D reconstruction of the brain scanned through the MRI scanner. The distance a camera moves between subsequent scans is the resolution of the camera. So, a camera with a higher resolution provides a higher number of 2D scans as compared to a camera with a lower resolution.

### **1.3.2 Modalities**

The MRI sequences can be generated based on different relaxation times. Most MRI sequences are either T1-weighted or T2-weighted. There is a different kind of MRI sequence that can be generated through the diffusion of water molecules. This diffusion helps to generate contrast in the magnetic resonance scans. The process of diffusion is explained in (Baliyan et al. 2016). In this work, three kinds of magnetic resonance scans are discussed.

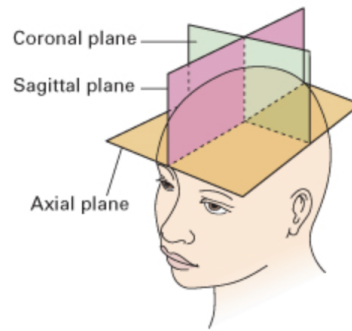


Figure 1.3: The three different axes Axial, Coronal, and Sagittal. Image Source: <https://casemed.case.edu/clerkships/neurology/>

1. T1-weighted
2. T2-weighted
3. DWI (or diffusion)

### 1.3.3 Axes

Any 3D scan, including an MRI scan, can be taken along one of the following three axes:

1. Axial
2. Coronal
3. Sagittal

An intuitive way to visualize the three axes is to assume if the camera is moving from either a person's forehead to neck, or from left ear to right, or the camera is on the top of head and is moving from nose to back.

The three axes are shown in Figure 1.3 and a sample scan along them is shown in Figure 1.4.

## 1.4 Machine Learning Background

Over the past decades, the field of AI and machine learning has rapidly evolved, as seen in figure 1.1, and is now being used to help make a computer make complex decisions.

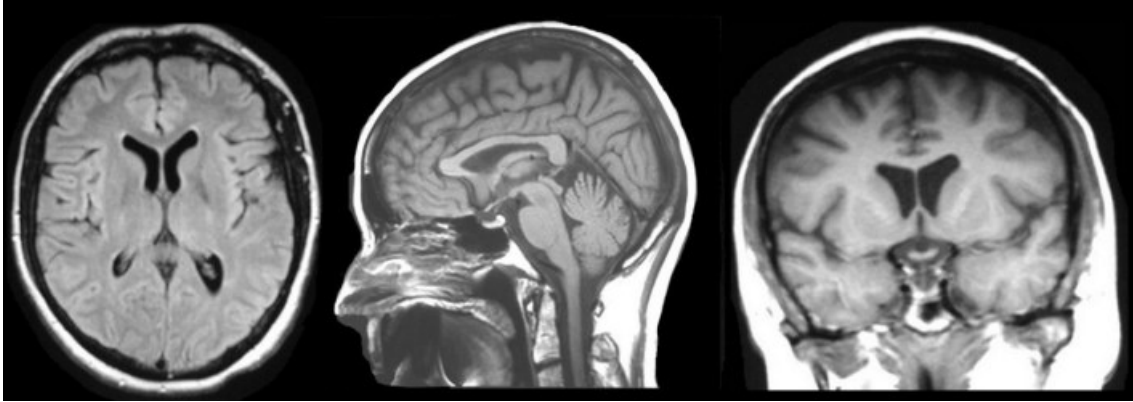


Figure 1.4: Sample MRI scans of a brain in the three different axes - Axial, Coronal and Sagittal in order. Source: <https://radiologykey.com/brain/>.

Traditional machine learning approaches work well with structured data, i.e. the data in forms of tables and unstructured data remains a largely unsolved problem, especially, for images (Alom et al. 2018).

The field of deep learning has seen rapid growth since the introduction of AlexNet (Krizhevsky et al. 2012). The introduction of ReLU in this paper was a major milestone for the progress of deep learning. ReLU helped to some extent in solving the issue of vanishing gradient problem. The following sections provide some background on deep learning, specifically, for some computer vision techniques.

### 1.4.1 Supervised Learning

Supervised learning is a class of problems where, for a given input  $X$ , an output  $Y$ , the goal is to learn a mapping  $f$ , such that  $y = f(X)$ . In supervised learning, the training dataset can be considered as a supervisor for the learning process. Supervised learning can be further grouped into two classes of problems:

1. Regression: A problem of predicting a real-valued continuous output such as temperature or stock price.
2. Classification: A problem of categorizing a sample into one of many classes such as animals or sentiments.

Pediatric brain tumor classification is modeled as a supervised learning problem in this work and a dataset of hand-labeled classes is used to learn a model.

## 1.4.2 Neural Networks and Deep Learning

Artificial neural networks are a class of algorithms that are inspired by and built to mimic a human brain. Neural networks typically contain an input layer, one or more hidden layers, and an output layer.

Neural networks containing more than one hidden layer are said to be deep neural networks. Deep neural networks have proven to be of great use in solving supervised learning problems (Alom et al. 2018).

## 1.4.3 Softmax Classification and Cross-Entropy Loss

A popular choice for classification problems is Support Vector Machines or SVMs (Boser et al. 1992). SVMs work well when the data is linearly separable. Softmax is an alternative to SVM and supports all kinds of datasets. The main difference in softmax and SVM comes in the loss function they both use:

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad (1.1)$$

The softmax function, as shown above, will provide a transformation of outputs to probabilities that sum to one. The loss function for softmax classification is cross entropy loss, that is the negative log-likelihood of a class.

Cross entropy loss can be used to measure the performance of a classification model whose output is a probability score. Cross entropy loss can be defined by the following equation.

$$\text{Cross Entropy Loss} = - \sum_{c=1}^m y_c \log(\hat{y}_c), \quad (1.2)$$

where  $y_c$  is the binary indicator of the classification is correct or not,  $\hat{y}_c$  is the probability of the sample belonging to the current class  $c$ , and  $m$  is the number of possible classes.

## 1.4.4 Convolutional Neural Networks

The ability of deep neural networks to outperform traditional methods in computer vision related problems have been mostly because of convolutional neural networks, or CNNs. CNNs have now shown to be very useful in many state-of-the-art algorithms on image data such as object detection, object localization, image retrieval, segmentation, and classification.

CNNs are composed of multiple layers performing convolutional operations and adding non-linearity. LeNet (LeCun et al. 2001) was one of the first and most influential works in using convolutional neural networks for image classification.

A convolution neural network tries to perform a series of transformations to the input image by learning a set of kernels or weight matrices at each layer. These kernels extract locally correlated features at each layer. The kernel weights are learnt using the backpropagation algorithm. Each convolution operation is usually followed by a non-linearity like ReLU that allows them to model complex functions. The invariance of CNNs to rotations and translations make CNNs excellent candidates for fast feature extraction in image-related tasks.

Convolutional neural networks usually have three main components:

1. Convolution Layer
2. Pooling Layer
3. Non-linearity or Activation

A convolution layer tries to learn the kernels or weights by dividing the input into a set of grids and multiplying the input with a set of learned weights from the kernel. The kernel is translated over the whole input grid that helps convolutional networks become invariant to translation. The kernel shares the weights from each convolving step in a layer, thus making the layer memory efficient. The outputs of these layers are feature maps that have refined features extracted from the previous layer. The convolution operation could also have a different number of filters, different filter sizes and different strides in each layer. There are some variants of standard convolutions such as dilated convolutions (also known as atrous convolutions) (Chen et al. 2017a). These convolutions perform the convolution operation in the reverse direction and lead to an increase in height and width with respect to the input. These convolutions have proven extremely effective in state-of-the-art algorithms, such as DeepLabV3 (Chen et al. 2017b) for image segmentation.

A pooling layer is a subsampling operation performed on the input to reduce the dimensionality. Since the convolution operation already extracts locally correlated features from the input, all of the input information becomes less important, and the input can be downsampled to reduce the time and space complexity. The pooling layer usually does not have any learnable parameters. There could be different algorithms for pooling, for example, max pooling and average pooling. Adaptive pooling (Saeedan et al. 2018) is a type of pooling layer that can be used to modify the output size of the feature maps produced



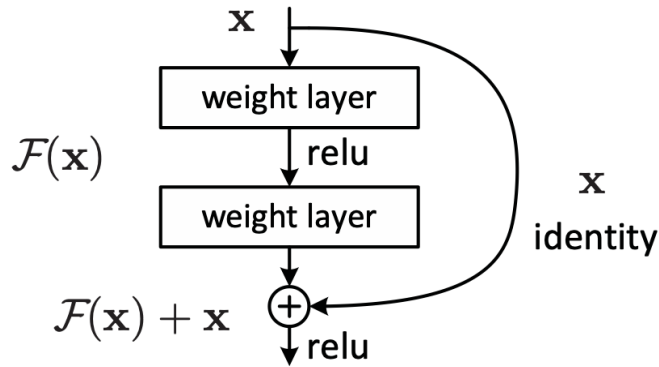


Figure 1.6: A residual block with a skip-connection. Source: (He et al. 2016).

This degradation is caused because of the use of ReLU activation in the network after every layer. As ReLU set almost half of the activations to zero, most of the gradients become zero and there is no learning happening. To resolve this issue, (He et al. 2016) proposed ResNets where the authors suggested adding a *skip-connection* at every layer to preserve the input and help the gradient propagation. This can be formally written as, instead of learning a mapping  $f(X)$ , it is possible to learn a mapping  $X + f(X)$ . This idea of residual networks has helped pushed networks to a much deeper level.

An example of a residual block is shown in Figure 1.6.

DenseNets (Huang et al. 2017) take the idea further by arguing to concatenate feature map outputs from the previous layer with the input to subsequent layers. This calls for feature exploitation and feature re-use as compared to feature exploration in the case of ResNets.

An example of a DenseNet with five layers is shown in Figure 1.7.

### 1.4.7 Inception Network

The inception network (Szegedy et al. 2015) provides a good set of general guidelines and best practices while building deep convolutional neural networks. It proposed the idea of split-transform-aggregate. The inception network provides the principle to split the input into multiple groups, perform the transformation in each group and aggregate the results for each group.

An example of an Inception block is shown in Figure 1.8.

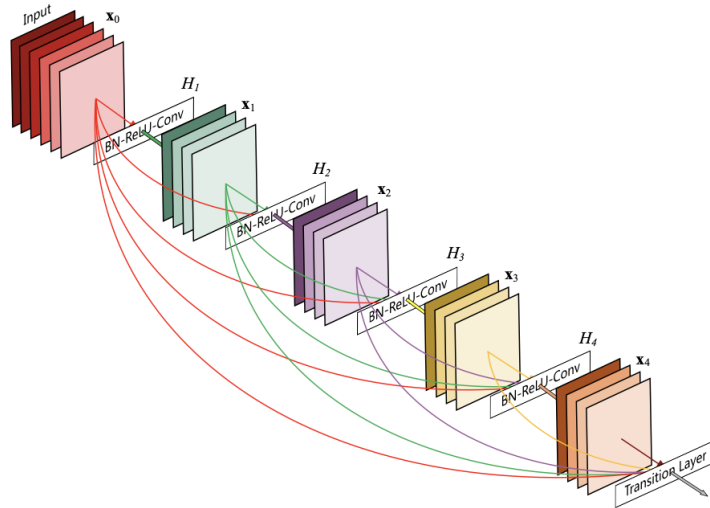


Figure 1.7: A dense network with five layers. Source: (Huang et al. 2017).

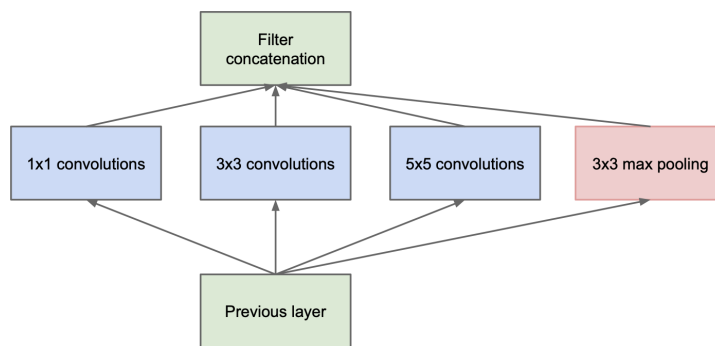


Figure 1.8: An Inception block. Source: (Szegedy et al. 2015).

### **1.4.8 Dropout Regularization for Deep Neural Networks**

Just like many other models in machine learning, deep neural networks are also prone to overfitting. Overfitting occurs when a model tries to memorize a dataset by learning noise in the dataset. This usually occurs when the model is too complex or there is of lack of training data.

Dropout regularization (Srivastava et al. 2014) is a popular technique to address overfitting in deep neural networks. It involves randomly removing deleting some nodes from a layer while training. While inference, the expected values of the node outputs are used. This makes the model more robust to individual noise elements.

### **1.4.9 Transfer Learning**

The work from (Yosinski et al. 2014) argued that weights learned on a deep learning task are transferable to other similar tasks. What this means is that the weights can be re-used from a model trained on one task to bootstrap a model that has to be trained on another similar task. Transfer learning in image tasks has been very successful with the release of models trained on the ImageNet dataset (Deng et al. 2009).

ImageNet is a dataset of real-world images. Using weights from a model pre-trained on this dataset has proven extremely useful in training models on different datasets with real-world images for a variety of tasks like image classification, object detection, object localization, etc.

## CHAPTER

# 2

## METHODOLOGY

The goal of this work is to compare and contrast the methods that can be used to perform classification on pediatric tumor data. This chapter includes the various methods used to apply different learning algorithms to classify the input Magnetic Resonance Imaging (MRI) images of a brain.

The following methods have been devised based on the availability of the dataset. The dataset used has MRI scans of only T-2 modality along only the axial plane. The input for the task of pediatric classification comes from MRI images. As discussed in 1.3.1, MRI images are volumetric in nature. These images scan the brain in 3D. There are two ways this data was used.

1. Split the brain into 2D slices
2. Treat the brain as 3D block

The design of a deep learning architecture would depend on what kind of input data is present. Hence, the experiments are divided into these two methods, 2D, and 3D. The following sections have the details of various experiments and models built using 2D and 3D inputs.

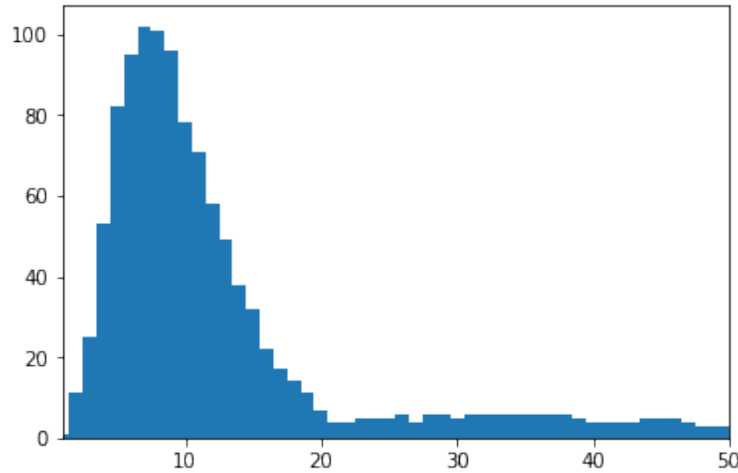


Figure 2.1: Number of tumor slices at every index for DIPG patients.

## 2.1 Heuristics based Slice Analysis

It was observed that all MRI slices of a brain do not contain tumor. Since the tumor is also a 3D object, some slices will have a small piece of the tumor, some slices will have a larger piece and some will not have any tumor. Intuitively, this makes sense because the tumor is only located in a small region of the brain and not all throughout. Thus, the assumption that all the slices for a brain will have the tumor is wrong. However, for all the slices that do have a tumor, the assumption still that the tumor will be of the same type holds true.

Each MRI scan produces 40-100 image slices depending on the resolution of the machine. It was also observed that each Magnetic Resonance Imaging scan had at least 5-6 (usually contiguous) slices that contained tumor. Out of these slices, the exact slices that have a tumor are known. Since all of the four classes of tumors in consideration lie near the back of the brain, it might be possible to heuristically determine the location of the slices that ordinarily contain tumor. Using the dataset, the slice locations that contain tumor can be bucketed with indices starting from 0.

Figure 2.1 shows a plot of the number of slices containing a tumor at an index for about 100 brain scans from patients with DIPG tumor. From this figure, it is observed that most of the brain scans have tumors in indices 3 to 17. Hence, the focus in the following subsection is only on these slices to build the 2D models.

## 2.2 2D Methods

For a brain, a set of MRI image slices and a class label that describes the tumor class it belongs to are present. Since the MRI images are a sequence of slices, it is possible to split them and treat them as separate images. The class label for each of those slices can be assumed as the class label of the whole brain. This is a reasonable assumption because for a given brain there is only one possible type of tumor and a single brain will not have two types of tumors simultaneously in our dataset.

Using this technique, a dataset can now be created with an input MRI image slice and an output tumor type. This image can now be fed into a convolutional neural network to extract features and classified into one of the four tumor types. As a baseline model, AlexNet (Krizhevsky et al. 2012) was used for classification. Each of the input images was resized to 256x256.

Transfer learning is demonstrated to be extremely effective in image classification, (Yosinski et al. 2014). A model pre-trained on ImageNet dataset (Deng et al. 2009) was used with transfer learning. However, ImageNet is a real-world dataset with 3-channel RGB images. Since the MRI images are single-channel greyscale images, they were replicated along channel dimension to make them 256x256x3 with mocked RGB channels. After establishing a baseline, further experiments were conducted with ResNet(He et al. 2016), Inception (Szegedy et al. 2015), and DesneNet(Huang et al. 2017) architectures in a similar setting. Some details on these architectures are provided in Sections 1.4.5, 1.4.6, and 1.4.9. A schematic diagram of the model architecture is provided in Figure 2.2.

As mentioned above, since all the MRI images are grayscale images, these images have pixel values in the range 0-255. As mentioned in Section 3.1, the dataset was collected from two different sites. These sites (or hospitals) usually do not have the exact same MRI scanning machines. Hence, depending on the model and the manufacturer, the intensities of exposure/brightness in the images might differ. Table 3.1 shows these manufacturers and their intensity information. Given the variability in the intensity of the images, treating them on the same intensity scale does not seem reasonable as the tumor regions are usually identified by the relative brightness of the images. Hence, the pixel values in these images were re-scaled such that the pixel values lie in the range 0-1. This normalization helps the model become agnostic of the MRI machine type as this relative intensity normalization helps expose the regions that have relatively higher pixel values.

These experiments are based on the idea to classify tumor type by using just one MRI image slice of the brain.

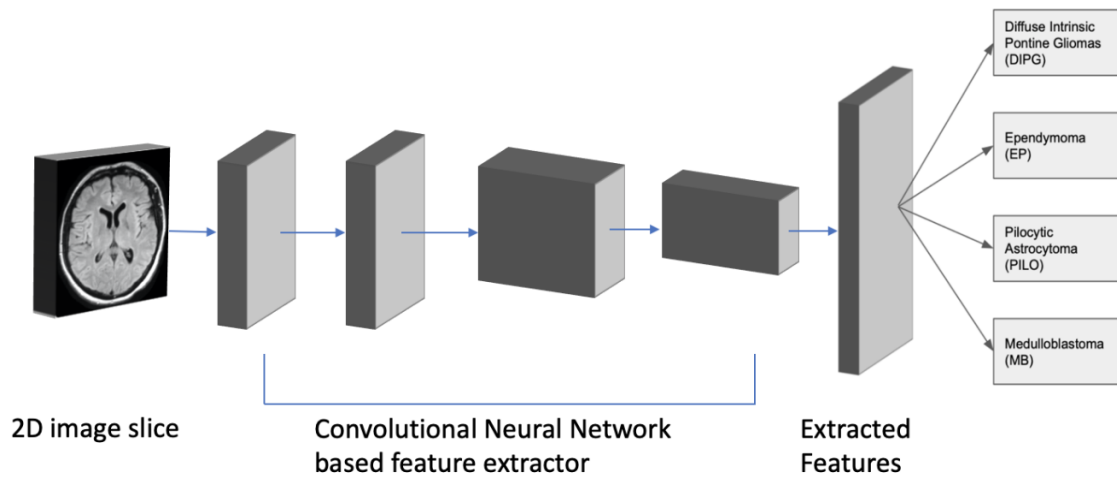


Figure 2.2: A schematic diagram of the 2D model architecture.

The MRI image slices can be divided in an MRI sequence into two categories:

1. Slices that contain tumor
2. Slices that do not contain tumor

Based on the analysis done in Section 2.1, the slices that contain tumor can be directly sampled and used for 2D classification approaches.

The following methods determine how the 2D slices can be used for training a classification model and how to use the results of the 2D slice classification to make a final prediction.

### 2.2.1 Per Slice Classification

For this experiment, it is assumed that the slices that contain tumor are already known and the goal is to identify what type of tumor is present. All the non-tumor slices are removed from the dataset for this experiment. Now for each slice, it is possible to predict a tumor class with some probability. The results might be ambiguous when different image slices in the same patient are classified as different tumor classes.

Table 2.1 shows the general architecture that was used. Different backbone architectures such as AlexNet, ResNet-18, ResNet-34, and ResNet-50 were experimented with as a feature extractor.

Table 2.1: 2D slice classification model. The CNN backbone can be any feature extractor such as AlexNet, ResNet, VGG etc.

	Layer	Description
1	Input - 2D Slice	$H1 \times W1 \times 3$
2	CNN Backbone	$H2 \times W2 \times C$ - size depends on backbone
3	AdaptiveAvgPool2D	$1 \times 1 \times C$
4	Linear	256
5	Dropout	probability 0.5
6	Softmax	4

### 2.2.2 Ensemble Per Slice Classification

For this experiment, again we predict the tumor class for all MRI image slices that contain tumor. Now for each slice, it is possible to predict a tumor class with some confidence. To find the final prediction, two different approaches were tried in this case:

1. Majority Vote: A simple majority voting technique can be applied to the predictions of all the slices to find the class for the whole brain scan.
2. Max confidence: The final class to be from the slice that predicts with the highest confidence will be considered.

The architecture used here is the exact same as in Section 2.2.1, that is mentioned in Table 2.1.

## 2.3 3D Methods

The MRI image scans provide a 3D scan of the brain. Hence, creating a 3D convolutional neural network seems intuitive if all the spatial correlations in the scans are to be used.

However, it becomes challenging to load huge blocks into memory for training a deep convolutional neural network model. This is primarily because one typically requires a graphics processing unit (GPU) to train deep learning models. GPUs tend to have a low amount of memory as compared to CPUs. If a model cannot fit in GPU memory, it is not trivial to train it. There are some ways to get around that issue as discussed in (Krizhevsky et al. 2012). These techniques usually involve manually segregating the model and offloading parts of it to another GPU(s) in a distributed setting. This step makes it increasingly hard to train and implement.

Another issue with using the whole 3D MRI scan of a brain occurs because of how tumor data is usually collected. As discussed in Section , the MRI image scans come from hospitals in various sites. These hospitals might not have the same type of equipment. The number of slices in an MRI scan depends on the resolution of the camera used. Since the number of slices is not fixed, it is not possible to create a fixed-sized 3D block of images that will be used to train a 3D convolutional neural network model.

Each image is of a different resolution. So stacking multiple images in a block will not work unless all images were resized to a fixed size. To get around this issue, all the images were resized to 256x256.

Transfer learning (Yosinski et al. 2014) from a model pre-trained on ImageNet dataset (Deng et al. 2009) is not straightforward in the case of 3D blocks. This is because ImageNet is a real-world dataset with 3-channel RGB images. But the 3D block of MRI images are single-channel greyscale images with a depth  $d = 15$ . So, no transfer learning from ImageNet was directly possible in this case. Since the model will severely overfit on the small dataset, hence a regression model on a dataset of adult brains with the labels for their ages was pre-trained. Transfer learning was used with this pre-trained model, as described in Section 3.2. Figure 2.3 shows a schematic diagram of the model used for transfer learning with age regression.

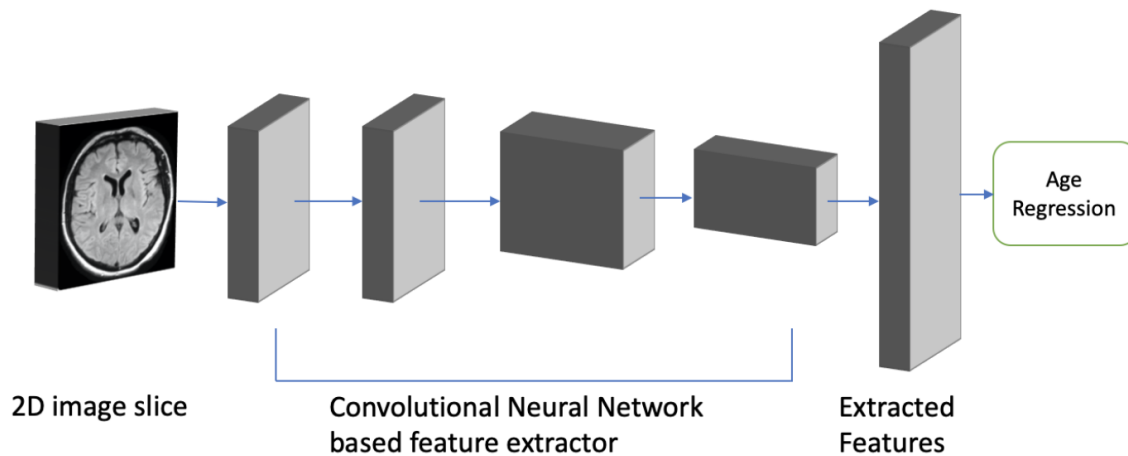


Figure 2.3: Convolution neural network architecture for transfer learning with age regression.

Similar to Section 2.2, experiments were conducted with an architecture built using ideas borrowed from AlexNet Krizhevsky et al. (2012), ResNet(He et al. 2016), Inception (Szegedy et al. 2015), and DesneNet(Huang et al. 2017) architectures. Based on the analysis

Table 2.2: 3D Block Classification Model.

	Layer	Kernel Size	Description
1	Input - 3D Slice		H x W x d x 1
2	Conv3d	3 x 3 x 3	64 kernels
3	MaxPool3d	3 x 3 x 3	
4	Conv3d	3 x 3 x 3	192 kernels
5	MaxPool3d	3 x 3 x 3	
6	Conv3d	3 x 3 x 3	256 kernels
7	MaxPool3d	3 x 3 x 3	
8	AdaptiveAvgPool		1 x 1 x 1 x C
9	Dropout		probability 0.5
10	Linear		4096 outputs
11	Dropout		probability 0.5
12	Linear		256 outputs
13	Dropout		probability 0.5
14	Softmax		4 outputs

done in Section 2.1, it is clear that we need not use all the MRI scans as input to the model. The following two methods can be used to work around the above-mentioned challenges.

### 2.3.1 3D Block Classification

As discussed in Section 3.1, since only a few slices have a presence of tumor a 3D block of all the slices need not be created. Instead, it is possible to create a 3D block using just those 15 slices located at index 3 to index 17.

This block of depth,  $d = 15$  will comfortably capture all the locations of the tumor in the brain. This model with 15 slices of MRI image scans had a lot of parameters and was extremely prone to overfitting. Hence, the next logical step is to try to reduce overfitting. This was done by the addition of a few dropout layers with dropout probability 0.5. The 3D model architecture is shown in Figure .

Table 2.2 shows the architecture that was used for 3D block classification.

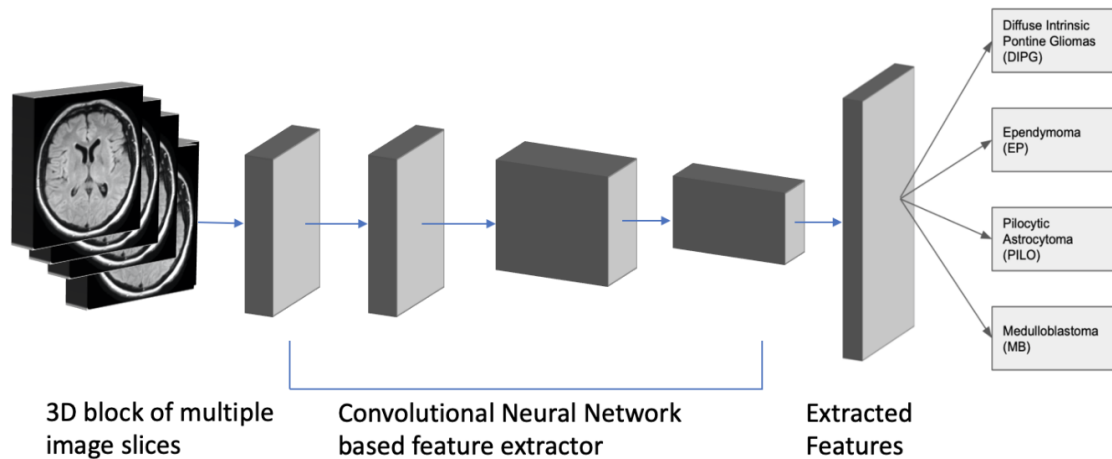


Figure 2.4: 3D MRI block convolution neural network architecture.

### 2.3.2 Sampling Slices for 3D Classification

Even after adding dropout layers, the model was still overfitting because of an extremely high number of parameters. Hence, the next logical thing to do was to reduce model complexity while preserving the 3D spatial correlations among the MRI image scans.

To overcome this issue of high model complexity, the depth  $d = 15$  of a block could be reduced. This would mean that multiple 3D block slices are present that could fit in the range of index locations 3 to 17. For example, if we use depth  $d = 5$ , we can get the following slices from the brain:

1. Slice from index location 3 to 7
2. Slice from index location 8 to 12
3. Slice from index location 13 to 17

This means that it is possible to create three samples using just one MRI scan of the brain. Taking this idea further, it is also possible to create overlapping 3D blocks from the MRI scans. The amount of overlap will determine how many samples that are generated. So, we generate samples with an overlap factor  $o = 2$ . The overlap factor determines how many slices to overlap in each sample generation. For example, with overlap factor  $o = 2$ , and depth  $d = 5$ , we can get the following slices from the brain:

1. Slice from index location 3 to 7

2. Slice from index location 5 to 9
3. Slice from index location 7 to 11
4. Slice from index location 9 to 13
5. Slice from index location 11 to 15
6. Slice from index location 13 to 17

As we can see above, using this technique it is possible to augment the samples by a factor of six increasing the dataset size and reducing the model complexity at the same time.

### 2.3.3 Multi-Slice architecture using shared weights

As discussed in Section 2.3, using 3D blocks seems like a good idea. Based on the experimentation in Section 2.3.1, it becomes challenging to fit everything into memory and efficiently train the model. Section 2.3.2 attempts to overcome these problems to some extent. It reduces the memory footprint of the model up to 3 times when we choose a depth  $d = 5$ , but these optimizations are not enough to get over the issue of having a large number of trainable parameters.

To overcome this issue of having an extremely high number of parameters, a few approaches could be taken. AlexNet (Krizhevsky et al. 2012) introduced the idea of a grouped convolution that can be used to train different parts of a model on different GPUs. Grouped convolutions have been successfully used in MobileNet (Howard et al. 2017), MobileNetV2 (Sandler et al. 2018), and other on-device models to reduce the memory footprint and reduce computation time complexity. ShuffleNet (Zhang et al. 2018) introduced the channel shuffle operation to reduce the correlation between filters and channel groups.

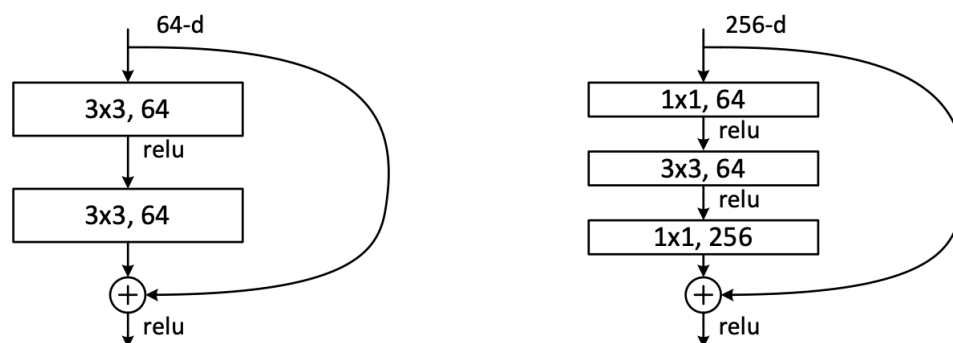


Figure 2.5: Left: Building block of a residual function, Right: Building block of a residual function with a linear bottleneck.

MobileNetV2 and residual networks (He et al. 2016) have successfully demonstrated the idea of using linear bottlenecks to improve computational speed and reduce the number of learnable parameters. This involves first using a 1x1 convolution operation to reduce the number of channels in the feature maps. 1x1 convolutions are extremely good for this purpose as they provide a relatively inexpensive way to reduce the number of parameters without changing the receptive field. Once the channel dimension of the feature maps has been reduced, a more expensive 3x3 convolution can be performed with a subsequent 1x1 convolution to again increase the channel dimension. Figure 2.5 shows an example of such a linear bottleneck with a residual function.

All of these techniques can help to reduce the number of learnable parameters in the model. However, using these techniques successfully requires more training data. Unfortunately, as datasets for MRI images are scarce and expensive to collect, we do not have a lot of such images. In our experiments, we discovered that using one of the above-mentioned techniques is not sufficient to solve this problem.

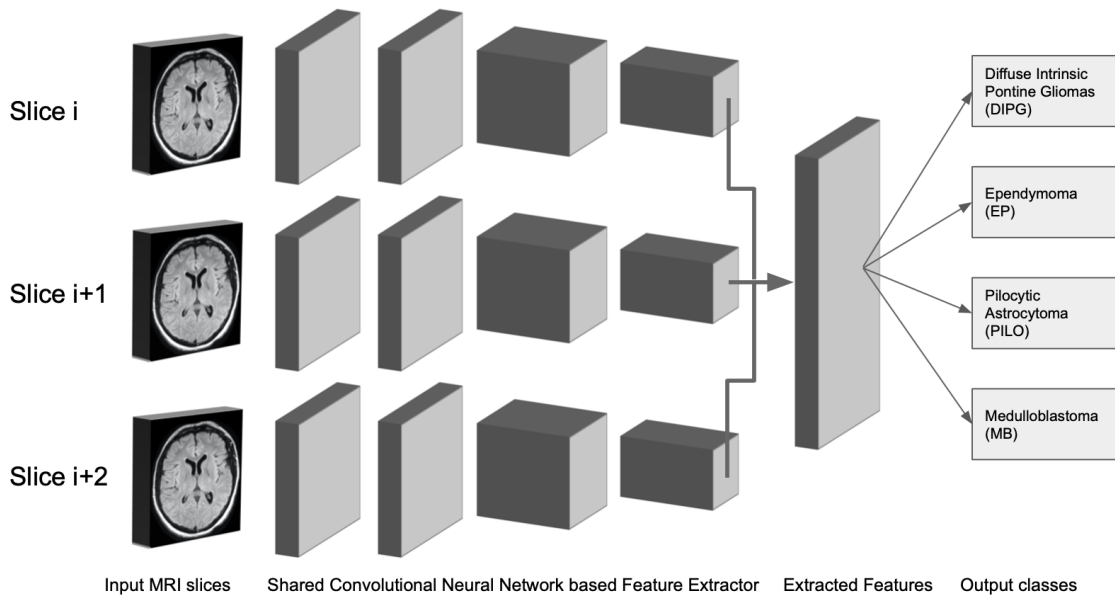


Figure 2.6: Multi-Slice architecture with convolutional neural network based feature extractors that share weights.

When a 3D block of an MRI image scans is generated, it is essentially a set of 2D images stacked on top of one another. So, we can process all the 2D images one at a time and

pass them through a convolutional neural network feature extractor. This will give us a set of features for each of the image slice of the 3D brain scan. Now, we can combine these extracted features from all of the 2D image slices and return a final softmax output that will classify all of the images in the brain. This still has the same issue of a high number of learnable parameters as we are now using multiple convolutional neural network feature extractors to process a single brain.

Since all of these images are MRI scans, the filters of the convolutional neural network feature extractor are being exposed to images that look very similar. Hence, there is no need to learn different filters for different slices of the 3D block. This is because all the images of the 3D block are similar-looking 2D slices of the brain. This means that we can pass the 2D image slices through a convolutional neural network feature extractor and share the weights for each of all these feature extractors. This means we will only need as many numbers of learnable parameters as required for a 2D convolutional neural network feature extractor and still process the whole 3D brain scan with it. This idea of sharing weights is very similar to how a typical convolutional neural network filter shares weights that are computed after translating over different parts of an image.

The next logical question that comes up is how many slices should be considered? Based on the analysis done in Section 2.1, the tumor is only present in approximately fifteen brain slices. Hence, to reduce the computation required, we can use just these slices for classification. The next question that arises is, do we need all these fifteen brain slices? Based on the analysis done in Section 2.3.2, we can use the same hyper-parameter depth,  $d$ . This hyper-parameter depth can be tuned to find the optimal number of slices to be used for classification in the same way as in Section 2.3.2. Similar to what we saw in the 2D architecture, this depth-based sampling helps us in two ways. First, to increase the number of training samples we can use. Second, to reduce the computational complexity of the model by reducing the number of floating-point operations (FLOPS) per brain scan.

Figure 2.6 shows an image of how we can use a set of convolutional neural network feature extractors by sharing weights. Section 2.5 discusses some ways to combine the features learned by the different convolutional neural network feature extractors.

Table 2.3 shows the architecture that was used for the 2D slice shared weights classification model.

Table 2.3: Multi-Slice Shared Weights Model.

	Layer	Kernel Size	Description
1	Input - 2D Slice		H x W x 1
2	Conv2d	3 x 3	64 kernels
3	MaxPool2d	3 x 3	
4	Conv2d	3 x 3	192 kernels
5	MaxPool2d	3 x 3	
6	Conv2d	3 x 3	256 kernels
7	MaxPool2d	3 x 3	
8	AdaptiveAvgPool		1 x 1 x C
9	Dropout		probability 0.5
10	Linear		4096 outputs
11	Dropout		probability 0.5
12	Linear		256 outputs
13	Dropout		probability 0.5
14	Softmax		4 outputs

## 2.4 Dealing with Class Imbalance

As discussed in Section 3.1, the dataset used for this work is imbalanced. To address this issue, the following two techniques were experimented with.

1. Delayed back-propagation
2. Loss re-weighting

### 2.4.1 Delayed Back-Propagation

Due to class imbalance in the dataset, a lot of mini-batches might not see the minority classes. Hence the weight updates made in those mini-batches might be biased towards the class.

To address this issue, the loss is collected over multiple mini-batches until we see at least  $N$  samples for each tumor class, where  $N$  is a hyper-parameter. Once every tumor class sees at least  $N$  samples, the error back-propagation is done.

## 2.4.2 Loss Re-Weighting

Another way to address the class imbalance is to re-weight the output from the loss function. To handle the class imbalance in the loss function, it can be scaled inversely proportional to the number of samples per class in the dataset.

Such a balanced loss function can be written as

$$\text{Balanced Loss} = \frac{L(y_c, \hat{y}_c)}{\frac{N_c}{\sum_c N_c}} \quad (2.1)$$

where  $L$  is the unbalanced loss function,  $y_c$  is the binary indicator whether the classification is correct or not,  $\hat{y}_c$  is the probability of the sample belonging to the current class  $c$ .

By scaling the loss, we can get the balanced cross-entropy loss from equation (2.1) above, that can be formulated as

$$\text{Balanced Cross Entropy Loss} = - \sum_{c=1}^m \frac{y_c \log(\hat{y}_c)}{\frac{N_c}{\sum_c N_c}} \quad (2.2)$$

where  $y_c$  is the binary indicator whether the classification is correct or not,  $\hat{y}_c$  is the probability of the sample belonging to the current class  $c$ , and  $m$  is the number of possible classes. This technique was found to be more superior to delayed back-propagation and was used in all subsequent experiments.

## 2.5 How to Combine Multi-Modal information?

This section deals with how to combine information obtained from multiple sources. Once we have features extracted from different sources, an important question that arises is how to combine that information for further processing. In the above experiments, this question arises multiple times. This problem comes when we want to combine features extracted with brain scans of different modalities, like T2 and DWI. This issue also occurs when we want to combine features extracted from different axes, like axial, coronal or sagittal. Another case that deals with combining multi-modal information is described in Section 2.3.3, where we try to combine information from features extracted by different 2D slices.

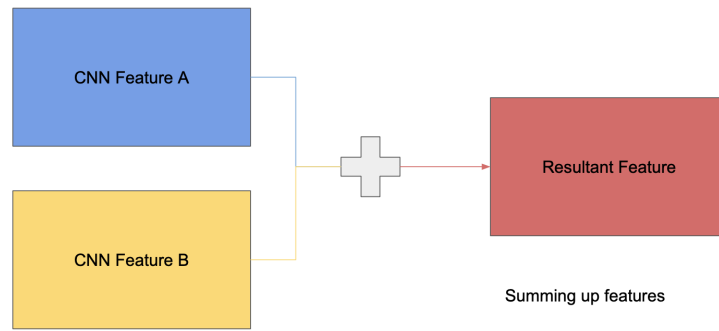


Figure 2.7: Summing up features A and B.

This problem can be tackled in a few different ways. Historically, people have approached this problem of combining features in various ways that could be simple or complex. Some of the simple ways to combine features have been used in residual networks and dense networks.

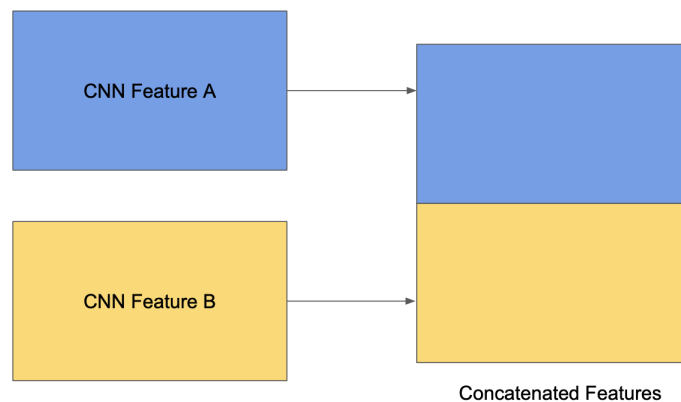


Figure 2.8: Concatenating features A and B.

Residual networks (He et al. 2016), tackle this problem by summing up the features using a "+" operation on the features (as shown in Fig. 2.7). Dense networks (Huang et al. 2017) deal with this issue by concatenating the features (as shown in Fig. 2.8). Doing a summation as in residual networks promote feature exploration as the layers after the summation operation will try to explore new features from these resultant "summed up" features. In contrast, dense networks promote feature exploitation by providing all possible features to the subsequent layers where the layers can re-use the features computed in the previous layers.

Both the above ways to combine features are hard-coded and force the model to combine features in the given predefined way. This makes the process fast and inexpensive but it comes at the cost of making the model less flexible. This can be avoided by using a bilinear fusion module (as shown in Fig. 2.9) for this operation. This module learns extra parameters in the form of  $W$  and  $b$  matrices, thus ensuring a rich vectorial representation that can encode even the most complex correlations between the different features. This involves learning weights of the matrix  $W$  by taking into account both features  $A$  and  $B$ .

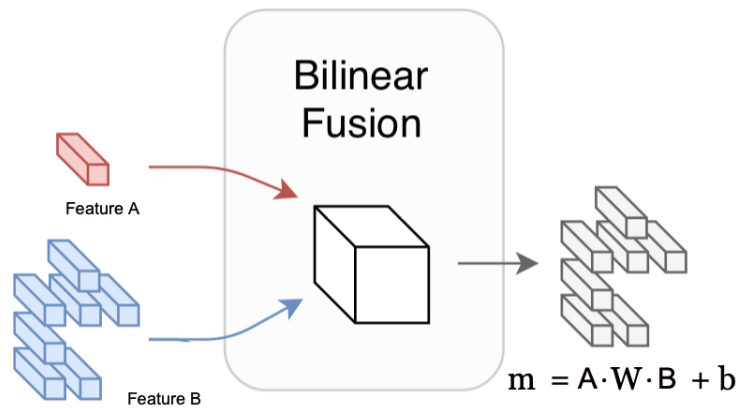


Figure 2.9: Bilinear Fusion for features A and B.

Using this learned vectorial representation can help overcome the above-mentioned issue of early assumptions on the model.

## CHAPTER

### 3

## RESULTS

### **3.1 Pediatric Tumor Dataset**

This Pediatric Tumor dataset was collected from two different hospitals. The dataset contains MRI image scans of patients who have been diagnosed with one of the following four kinds of brain tumor - Medulloblastoma, Diffuse Intrinsic Pontine Glioma, Ependymoma or Pilocytic. This dataset focuses on rare types of tumors and this is one the reason why the total amount of MRI scans of patients is less. The dataset contains MRI scans with T2-weighted modality, and DWI modality. However, the number of scans available as DWI is relatively low and thus, only T2-weighted scans were used in the experiments discussed in Chapter 2. All of the scans provided in the dataset were in the axial plane.

Different hospitals, especially the ones across different cohorts, usually do not have the same MRI scanner machines. Hence, depending on the model and the manufacturer, the intensities of exposure/brightness in the images might differ. Table 3.1 lists these manufacturers and their intensity information and some statistics for the scans in the dataset.

Table 3.1: MRI machine manufacturers and the intensity values for the two different models: 1.5T and 3T.

Field Scanner	Maximum	Mean	Std. Deviation	Median
GE-1.5T	5537	1742.24	900.807	1515.50
GE-3T	10599	5157.49	1518.842	5143.00
Philips-1.5T	2807	731.01	502.012	539.50
Philips-3T	525	479.00	26.092	474.00
Siemens-1.5T	3615	1549.08	657.380	1306.50
Siemens-3T	3062	2374.44	291.936	2348.00

### 3.1.1 Dicom File Format

Medical datasets with Magnetic Resonance Imaging (MRI) scans are usually stored in dicom image format. Dicom stands for Digital Imaging and Communications in Medicine and is the industry standard in storing and viewing these files.

For viewing these images on a Mac computer, OsiriX software can be used. These files can also be opened and read programmatically using the pydicom library in python.

Table 3.2: Some of the fields in a DICOM.

Field No.	Field Name	Description
1	PatientName	Hashed of the patient (example - ST001)
2	PatientID	A unique identifier specific to a patient in a study
3	PatientAge	Age of the patient at the time of the scan
4	PatientSex	Sex of the patient
5	SliceLocation	Exact co-ordinates for current image slice
6	PatientWeight	Weight of the patient at the time of the scan
7	SeriesDescription	Modality for the MRI Scan (example - T2 or DWI)
8	ImagePositionPatient	Relative position of the image slice

Medical data such as this brain tumor classification data has personally identifiable information (PII). It has to be first anonymized before it can be used for any model training or statistical analysis purposes. To do this anonymization we were able to write a parser for the dicom images using pydicom that will replace personal information with hashed token and write the modified dicoms as new files. These anonymized files can now be used for training models. Table 3.2 shows some of the fields present in a dicom file before and after

Table 3.3: Table showing distribution of number of patients with different tumor classes in T2-weighted modality.

		Hospital 1	Hospital 2	Total
1	MB	135	146	281
2	DIPG	95	32	127
3	EP	77	48	125
4	PILO	130	0	130
Total		437	226	663

anonymization.

### 3.1.2 Brain Tumor Classes

As mentioned earlier, the four types of tumor classes present in the dataset are Medulloblastoma (MB), Diffuse Intrinsic Pontine Glioma (DIPG), Ependymoma (EP) or Pilocytic (PILO). The dataset is not perfectly balanced and there is some imbalance in the number of patients who were diagnosed with some of the classes. Table 3.3 shows the distribution of the four tumor classes in the dataset. As we can see, the number of patients for different types of tumors from a single hospital is not very high. To overcome this problem the dataset from the different hospitals have been merged.

This dataset was split into training, testing, and validation sets. The splitting methodology differed for different methods discussed in chapter 2. To ensure no information leakage, the splitting for each method was done in such a way that all MRI planes of the same patient go in the same set. This was particularly handled for the cases where a patient makes multiple visits to the hospital to get MRI scans and hence, all his/her scans should go in the same dataset.

## 3.2 Pediatric Brain Age Dataset

As discussed in section 1.4.9, transfer learning has proven to be useful in problems where the amount of dataset is sufficient for a given task but similar data is available for other tasks (Yosinski et al. 2014). Since the number of MRI scans in the pediatric tumor dataset is not very high, some other brain dataset can be used to perform transfer learning. For this purpose, another dataset was used. This dataset was collected from one hospital site and had MRI scans of non-cancerous pediatric brains and their age. This dataset was also

Table 3.4: Ensemble slice classification results for specific CNN backbone architectures using either max-confidence or majority vote.

	Backbone CNN Architecture	Max Confidence Accuracy	Majority Vote Accuracy
1	AlexNet	66.45%	68.51%
2	ResNet-18	70.14%	71.95%
3	ResNet-34	67.33%	70.28%
4	ResNet-50	62.87%	65.28%

anonymized similar to the pediatric brain tumor dataset. This dataset had scans for 983 patients. Each of the patient scans had 15 image slices. This dataset is on the axial plane and the MRI scans are T2-weighted. Because of the similar nature of this dataset, it was suitable for transfer learning in our experiments.

### 3.3 Comparison of Methods

Experiments were run on the above dataset to evaluate the methods described in chapter 2. To compare the performance of different approaches, the accuracy metric was used. This section discusses each of the methods, the architectures used, the values of hyperparameters and the most optimal performance. All the results mentioned here were obtained by incorporating loss re-weighting as described in Section 2.4.2.

#### 3.3.1 Ensemble 2D Slice Classification

2D methods are described in section 2.2. 2D slice classification is the classification of each MRI image slice in one of the four classes. As mentioned in section 2.2.1, all the non-tumor slices were removed from the dataset for this experiment and results are based on per-slice classification. Table 2.1 shows the generic neural network architecture for the specific architectures that were experimented.

The main issue with 2D per-slice classification is to identify the class of the brain scan as a whole. In our experiments, it was observed that different slices of the brain sometimes erroneously give different tumor classes. As described in section 2.2.2, two approaches were experimented. Table 3.4 summarizes the results obtained by those two methods. The results seem reasonable and work as a good baseline.

Table 3.5: 3D Block Classification results with blocks of different depths.

	Depth $d$	Accuracy
1	3	68.50%
2	5	72.85%
3	10	71.18%
4	15	67.32%

### 3.3.2 3D Block Classification

In the experiments with 2D slice classification, the depth dimension of the brain is ignored. This motivates 3D block classification. As described in Section 2.3.1, and Section 2.3.2, two approaches were experimented. Table 3.5 summarizes the results obtained by those two methods.

It can be inferred from the results that the model is not able to learn when the slice depth is 15. This is because the number of learnable parameters with a depth size,  $d = 15$  is extremely high. Since the amount of training examples present is very low, the model is not able to learn so many parameters and is severely overfitting. We can also see that this issue is overcome to some extent when a low value of depth is selected.

### 3.3.3 Multi-Slice Shared Weights Classification

Using a depth hyper-parameter helped reduce the number of learnable parameters in section 2.3.1. However, a 3D input will still have a huge number of learnable parameters, making the model hard to learn. This method aims to overcome this problem by using a 2D network with a 3D input by treating each slice in the 3D input as an individual 2D input. As mentioned in section 2.5, different approaches to combine learned representations were experimented. For this experiment, a depth  $d = 5$ , was used. Table 3.6 provides the results for each of them. We can see that concatenation works the best, followed by bilinear fusion, followed by addition. Bilinear fusion is expected to work best with more amount of data as it introduces more learnable parameters that will require more training data.

The best results for each of the approaches are summarized in Table 3.7.

Table 3.6: Results for different approaches of combining features for Multi-Slice Shared Weights Model.

	Feature Fusion	Accuracy
1	Addition	71.41%
2	Concatenation	76.90%
3	Bilinear Fusion	73.19%

Table 3.7: Most optimal results for all of the methods discussed in this work.

	Method	Best Accuracy
1	Ensemble 2D Slice Classification (patient-wise results)	71.95%
2	3D Block Classification (15 slices per patient)	67.32%
3	3D Block Classification (ensemble of 5 slices per block)	72.85%
4	Multi-Slice Shared Weights Classification (ensemble of 5 slices per instance)	76.90%

## CHAPTER

### 4

## RELATED WORK

With the advancement of AI and deep learning, several researchers have attempted to solve problems related to brain tumors and MRI datasets. This section highlights some of those works. Most of the works done on brain tumors are on the segmentation of tumors in an adult brain.

The work from (Lee et al. 2017) provides an excellent overview of the applications of AI and deep learning in medical imaging. This survey also compares traditional machine learning approaches to deep learning approaches in the context of medical imaging. As mentioned in this survey, convolutional neural networks have been proven to be superior over traditional machine learning approaches for medical imaging - especially in image segmentation and image registration.

### **4.1 BRATS Challenge**

BRATS Challenge, (Menze et al. 2014) is a yearly competition organized by Perelman School of Medicine at the University of Pennsylvania. This competition is being organized since 2012 and aims to explore the state-of-the-art methods for the segmentation of brain tu-

mors in multimodal MRI scans. As this is a segmentation problem, the main metric used to compare performance is the dice coefficient (Zou et al. 2004). The dice coefficient is essentially the intersection over the union of the segmented data and the ground truth data. Although brain tumor segmentation does not have the same challenges as classification, the methods designed to be implemented on segmentation can serve as a good starting point for classification. All the MRI brain scans provided in the BRATS dataset are for adult brains.

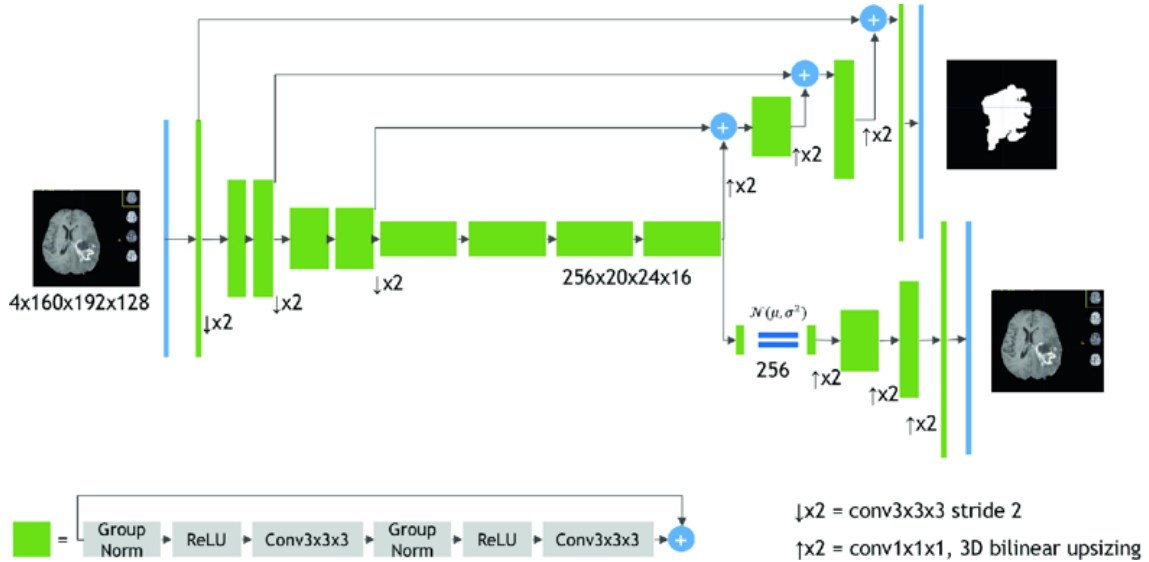


Figure 4.1: Schematic visualization of the network architecture used in 3D MRI Brain Tumor Segmentation Using Autoencoder Regularization, (Myronenko 2018).

The winner of BRATS challenge 2018 (Myronenko 2018), proposed a 3D autoencoder based model to perform segmentation. As shown in Figure 4.1, the model tried to create a 3D construction of the whole brain using an encoder-decoder style approach. The decoder branches out to give a segmentation output leading to a multi-task model where loss is evaluated at both branches. The loss function used in this work is composed of three loss terms.

$$L = L_{dice} + 0.1 * L_{L2} + 0.1 * L_{KL} \quad (4.1)$$

Here, the dice coefficient loss is responsible for segmentation and the L2 and KL divergence loss are responsible for decoder's reconstruction of the brain. This method achieved a dice coefficient of 0.81 on the core tumor region.

## 4.2 Deep Learning on MRI datasets

The survey paper (Lee et al. 2017), provides a general overview of the deep learning techniques used in medical imaging.

A recent work on Knee MRI datasets was to identify Anterior Cruciate Ligament (ACL) tears, (Bien et al. 2018). This work suggested an approach to extract features from different axes - axial, coronal, and sagittal. This was done through a proposed MRNet that will take three 2D image slices as input corresponding to each of the axes.

U-Net (Ronneberger et al. 2015), suggested an encoder-decoder style approach for segmentation that proved to be very useful for segmentation problems in the medical domain. This model could be trained with either pixel-wise binary cross-entropy or dice coefficient based loss function. Figure 4.2 shows the proposed U-Net model.

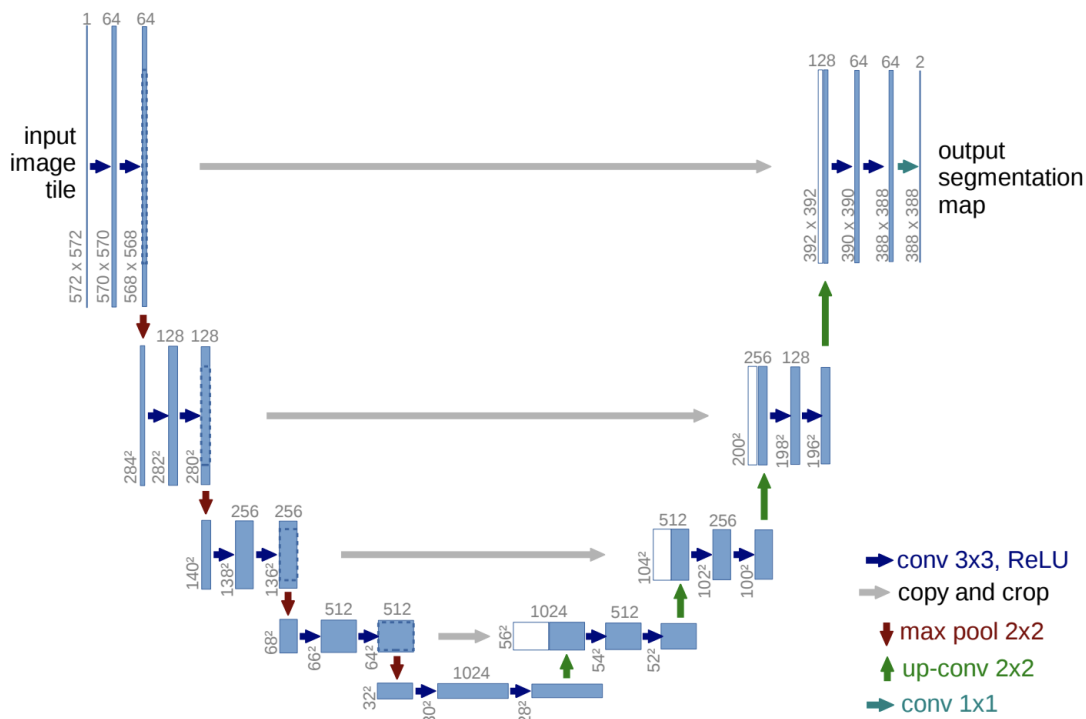


Figure 4.2: U-Net model architecture.

Another influential work in brain tumor segmentation was from (Havaei et al. 2017). The work proposed a dual-path network that used two different paths and had different sizes of kernels in each path. The motivation behind this dual-path architecture was that one path can learn the spatial location of the tumor and the other path can use those relevant

features to improve segmentation. Figure 4.3 shows the TwoPathCNN model proposed in this work.

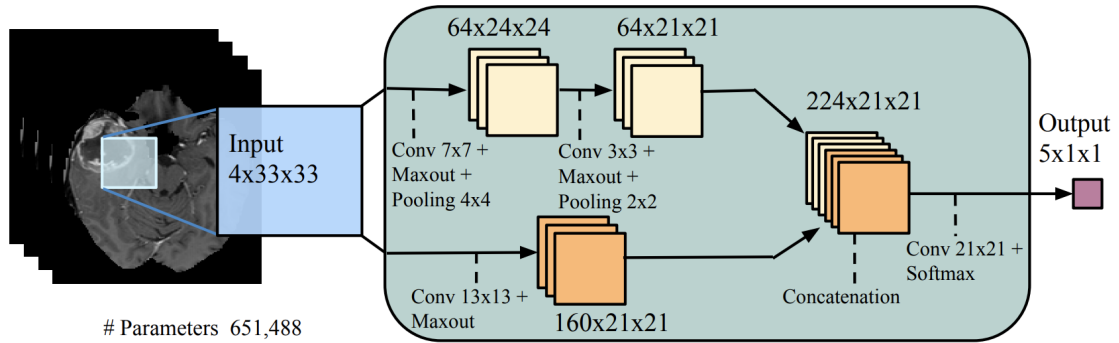


Figure 4.3: TwoPathCNN architecture proposed in Havaei et al. (2017).

## CHAPTER

# 5

## CONCLUSION AND FUTURE WORK

### 5.1 Conclusions

This work highlights some of the challenges of working with pediatric brain tumor problems. The proposed solutions using deep learning techniques act as a good starting point in understanding and working with MRI datasets. The proposed multi-slice shared weights neural network architecture seems to work the best for the pediatric tumor dataset. However, it is possible to get better results by using a more complex 3D block model with sampling slices. This is mainly because of the use of 3D convolutions that will explore the depth dimension of the brain as well. Overall, this work provides a list of the techniques that can be applied to solve brain tumor classification problems on MRI datasets.

The following is a list of challenges faced and the lessons learned in dealing with them.

1. Lack of data - Transfer learning from similar brain datasets helps in overcoming this challenge to a certain extent.
2. Typical data augmentation techniques are not applicable - Based on the type of model we use, some techniques such as the overlap technique mentioned in section 2.3.2 can be applied to increase the dataset size.

3. 3D MRI data makes feature extraction computationally expensive - Convolution neural networks work well for such unstructured data. Sampling slices instead of taking the whole brain helps reduce the computational complexity.
4. Training times are huge - There is no easy work around for this. 3D models take at least 2-3x more time to train as compared to 2D models. Cloud GPUs help accelerate the training process.
5. Some of the tumors occur very rarely that leads to an imbalance in the dataset - Loss re-weighting is an effective solution for this.
6. Variability in dataset is huge due to change in MRI technology, different machines from different manufacturers and different hospitals - Intensity normalization helps to a certain extent to reduce this variability. However, this is still an open issue.

## 5.2 Future Work

The main challenges faced in this work are open problems. Data availability is a major operational problem. This is partly because the tumor classes this work deals with are rare tumors. Even with the availability of specialized equipment, personally identifiable information (PII) becomes a hurdle in curating datasets for research purposes.

Image registration is an open problem with MRI datasets. It deals with the alignment of MRI slices in 3D space. Since a patient might move slightly during an MRI scan, it is possible that the image slices are not perfectly aligned. This becomes an issue for 3D block-based approaches where we try to incorporate the information from the depth dimension.

Skull stripping is another direction that can be explored. If it is possible to remove the skull from the MRI scan, the problem because relatively simpler as we do not have noisy inputs from the skull and the eyes distorting the tumor regions.

As mentioned in section 4.2, the work on Knee MRI datasets used different axes to extract out Anterior Cruciate Ligament (ACL) tears (Bien et al. 2018). A similar approach can be experimented for brain tumors if the data for different axes is available.

All the experiments done in this work are on T2-weighted modality. If datasets with multiple modalities are available, it might be possible to get better results with a multi-modal model where we combine T2 with DWI and T1 using techniques similar to the ones discussed in section 2.5.

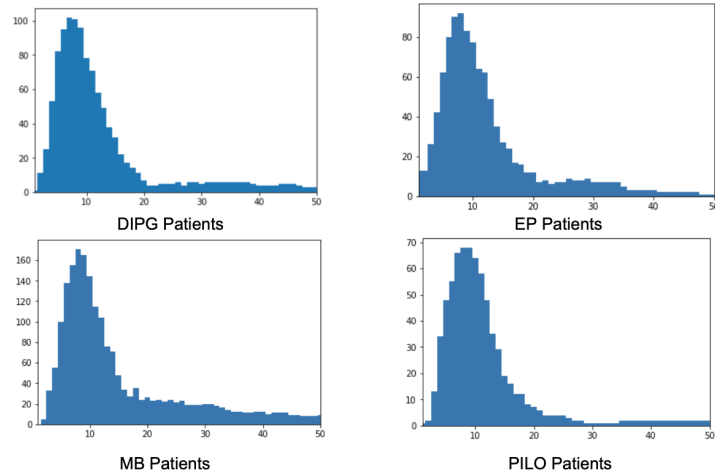


Figure 5.1: Number of tumor slices at every index for patients with different tumor types.

Figure 2.1 shows a plot of the number of slices containing a tumor at an index for about 100 brain scans from patients with DIPG tumor. From this figure, we can see that most of the brain scans have tumors in indices 3 to 17. This sampling of slices is the premise of this work.

This may not hold true for other tumor classes such as EP, MB or PILO. Figure 5.1 shows the plots for all the four tumor classes. Another issue lies with the difference in resolution of MRI machines. Since MRI scans have different number of slices ranging from 40-100, it is likely that the relative positions of the brain that slices with low resolution correspond to will not be the same as the slices with high resolution. Optimizing the slice indices that fit well with all of the tumor classes and different resolutions can be considered as a good next step towards this research.

## REFERENCES

- Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Van Esesn, B. C., Awwal, A. A. S., and Asari, V. K. (2018). The history began from alexnet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*.
- Baliyan, V., Das, C. J., Sharma, R., and Gupta, A. K. (2016). Diffusion weighted imaging: Technique and applications. *World journal of radiology*, 8(9):785.
- Bien, N., Rajpurkar, P., Ball, R. L., Irvin, J., Park, A., Jones, E., Bereket, M., Patel, B. N., Yeom, K. W., Shpanskaya, K., et al. (2018). Deep-learning-assisted diagnosis for knee magnetic resonance imaging: development and retrospective validation of mrnet. *PLoS medicine*, 15(11):e1002699.
- Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM.
- Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2017a). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848.
- Chen, L.-C., Papandreou, G., Schroff, F., and Adam, H. (2017b). Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*.
- Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.
- Druzhkov, P. and Kustikova, V. (2016). A survey of deep learning methods and software tools for image classification and object detection. *Pattern Recognition and Image Analysis*, 26(1):9–15.
- Havaei, M., Davy, A., Warde-Farley, D., Biard, A., Courville, A., Bengio, Y., Pal, C., Jodoin, P.-M., and Larochelle, H. (2017). Brain tumor segmentation with deep neural networks. *Medical image analysis*, 35:18–31.
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

- Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.
- LeBlanc, A., Lin, C., Shackelford, L., Sinitsyn, V., Evans, H., Belichenko, O., Schenkman, B., Kozlovskaya, I., Oganov, V., Bakulin, A., et al. (2000). Muscle volume, mri relaxation times (t2), and body composition after spaceflight. *Journal of applied physiology*, 89(6):2158–2164.
- LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (2001). Gradient-based learning applied to document recognition. In *Intelligent Signal Processing*, pages 306–351. IEEE Press.
- Lee, J.-G., Jun, S., Cho, Y.-W., Lee, H., Kim, G. B., Seo, J. B., and Kim, N. (2017). Deep learning in medical imaging: general overview. *Korean journal of radiology*, 18(4):570–584.
- Liang, Z.-P. and Lauterbur, P. C. (2000). *Principles of magnetic resonance imaging: a signal processing perspective*. SPIE Optical Engineering Press.
- Menze, B. H., Jakab, A., Bauer, S., Kalpathy-Cramer, J., Farahani, K., Kirby, J., Burren, Y., Porz, N., Slotboom, J., Wiest, R., et al. (2014). The multimodal brain tumor image segmentation benchmark (brats). *IEEE transactions on medical imaging*, 34(10):1993–2024.
- Myronenko, A. (2018). 3d mri brain tumor segmentation using autoencoder regularization. In *International MICCAI Brainlesion Workshop*, pages 311–320. Springer.
- Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer.
- Saeedan, F., Weber, N., Goesele, M., and Roth, S. (2018). Detail-preserving pooling in deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9108–9116.
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

- Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328.
- Zhang, X., Zhou, X., Lin, M., and Sun, J. (2018). Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6848–6856.
- Zou, K. H., Warfield, S. K., Bharatha, A., Tempany, C. M., Kaus, M. R., Haker, S. J., Wells III, W. M., Jolesz, F. A., and Kikinis, R. (2004). Statistical validation of image segmentation quality based on a spatial overlap index: scientific reports. *Academic radiology*, 11(2):178–189.