Heaven's Light is Our Guide

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Rajshahi University of Engineering & Technology, Bangladesh

Binary Classification of Brain MRI Images for Detection of Brain Tumor using Convolutional Neural Network

Author
Anika Bushra Chowdhury
Roll No. 143031
Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology

Supervised by
Emrana Kabir Hashi
Assistant Professor
Department of Computer Science & Engineering
Rajshahi University of Engineering & Technology

ACKNOWLEDGEMENT

First of all, I would like to express my deepest sense of gratitude to the almighty Allah for giving me the ability to complete this thesis work. I would like to express my sincere gratitude to my honorable supervisor, Emrana Kabir Hashi, Assistant Professor, Department of computer Science & Engineering, Rajshahi University of Engineering & Technology(RUET). This thesis would not have been completed without her support and guidance. Her constant encouragement gave me the confidence to carry out my work.

I would also like to express my sincere appreciation and gratitude to all of our respective teachers, Department of Computer Science & Engineering, Rajshahi University of Engineering & Technology(RUET) for their valuable suggestions, extending facilitations and inspirations.

Lastly, I would like to thank my parents, family members and friends for their optimism, support and encouragement.

Date:12 March 2021
RUET, Rajshahi

Anika Bushra Chowdhury

Heaven's Light is Our Guide

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Rajshahi University of Engineering & Technology, Bangladesh

CERTIFICATE

*This is to certify that this thesis report entitled "**Binary Classification of Brain MRI Images for Detection of Brain Tumor using Convolutional Neural Network**" Submitted by* **Name: Anika Bushra Chowdhury, Roll:*143031*** *in partial fulfillment of the requirement for the award of the degree of Bachelor of Science & Engineering of Rajshahi University of Engineering & Technology, Bangladesh is a record of the candidate own work carried out by her under my supervision. This thesis has not been submitted for the award of any other degree.*

Supervisor	External Examiner
---	---
Emrana Kabir Hashi	
Assistant Professor	
Department of Computer Science &Engineering	Department of Computer Science &Engineering
Rajshahi University of Engineering &Technology	Rajshahi University of Engineering &Technology
Rajshahi-6204	Rajshahi-6204

ABSTRACT

Classification of brain tumor is a critical task to assess the tumors and settle on a treatment resolution as per their classes. Although there are many imaging methods used to identify brain tumors, nevertheless MRI is most frequently used. The popularity of MRI relies on the fact that it has a condescending image quality despite being non-invasive and not involving any ionizing radiation. In recent times deep learning has shown an extraordinary performance on classification problems. Acknowledging that, CNN (Convolutional Neural Network) a class of deep neural networks is used in this thesis for binary classification of brain MRI images. The dataset includes 253 images in total, among which, 98 are non-tumor images and 155 are tumor images. The best results are shown by augmented dataset and pre-trained models. Without augmentation the dataset is tremendously small and tends to overfit the training data and thus results in bad performance. However, after augmentation the result does not improve much while using the simple 1, 2 or 3 convolutional layer architecture. The accuracy on unseen test data is 88.39%, 87.10% and 82.85% for 1, 2 and 3-layer models respectively. The pre-trained models have rather finer results other than ResNet50 which has the lowest accuracy rate of all models and it is 82.20%. VGG-16 has obtained 97.42%, VGG-19 has obtained 97.10% and InceptionV3 has obtained 96.45% accuracy on test data. Even though VGG-16 has the highest accuracy rate, it takes 13.8 minutes per epoch whereas, InceptionV3 takes only 6.6 minutes. Therefore, the most convenient CNN model would be InceotionV3 considering all the potentials such as the performance level as well as the computational cost.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

NN:	Neural Network.
CNN:	Convolutional Neural Network.
ANN:	Artificial Neural Network.
RGB:	Red Green Blue.
SVM:	Support Vector Machine.
ReLU:	Rectified Linear Unit.
ADAM:	Adaptive Moment Estimation.
CBAM:	Convolutional Block Attention Module.
GD:	Gradient Descent.
SGD:	Stochastic Gradient Descent.
MBGD:	Mini-Batch Gradient Descent.

Chapter 01

Introduction

Introduction:

1.1 Brain Tumor

Brain tumor can be characterized as unnatural and uncontrolled development in brain cells. Since the human skull is a rigid and volume bounded body, therefore, sudden development may influence a human capacity according to the involved part of the brain; additionally, it might spread into other body organs and influence human functions [1]. According to the world cancer report published by the World Health Organization (WHO), brain cancer accounts for less than 2% of human cancer; however, severe morbidity and complications are produced [2]. Cancer research corporation in the United Kingdom reported that there are about 5,250 deaths every year by the act of brain, other Central Nervous System (CNS) and intracranial tumors in the UK [3].

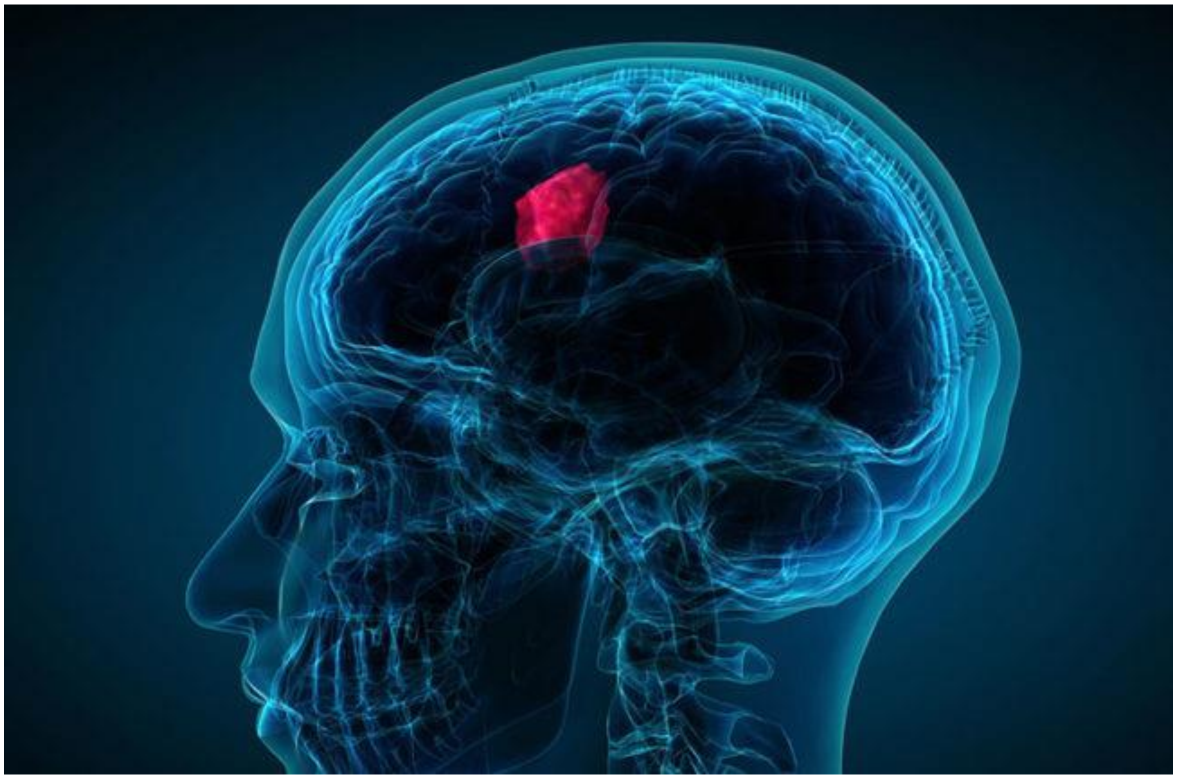


Figure 1.1: Brain Tumor.[1]

[1]https://health.clevelandclinic.org/what-are-the-actual-warning-signs-of-a-brain-tumor/

There are many ways to classify brain tumors, for example, primary and secondary tumors. The first one is about 70% of all brain tumors, while secondary tumors are the remaining 30%. This classification is determined according to tumor origin just as tumors first originate in the brain are called primary tumors. Whereas, tumors first arise in any other part of the body and then transferred to the brain are called secondary tumors, and most of them are malignant [4].

Also, Tumors can be classified as benign or malignant. Benign tumors are those that stay in their prior location without attacking other parts of the body. They do not grow to nearby structures or to inaccessible parts of the body. Benign tumors in general develop gradually and have well-defined borders. Malignant tumors have cells develop wildly and spread locally or potentially too far off destinations. Malignant tumors are cancerous. They expand to remote body parts with help of the bloodstream or the lymphatic system. Malignant tumors can spread rapidly and require treatment to avoid spread [5].

1.2 MRI Image

Various imaging methods can be utilized to identify and classify brain tumors. However, MRI is one of the most techniques. It is widely used because of the fact that it uses no ionizing radiation during the scan. MRI is the abbreviation of Magnetic Resonance Imaging which is a noninvasive diagnostic test that takes detailed images of the soft tissues of the body. Dissimilar to X-rays or CT scan, images are created with a magnetic field, radio waves, and a computer. It permits doctors to have a sight of spine or brain in slices, as if it were sliced layer-by-layer and a picture taken of each slice. This test can help analyze tumors, strokes, and plate hernia [6].

1.2.1 Benefits of MRI

- MRI is non-invasive.
- MRI does not involve any radiation.
- MRI differentiating specialist is less inclined to create a hypersensitive response that may happen when iodine-based substances are utilized for x-beams and CT examines.
- MRI gives amazingly clear, point by point pictures of delicate tissue structures that other imaging methods can't accomplish.

- MRI can without much of an effort make several pictures from practically any direction and in any orientation.
- Unlike strategies that inspect little pieces of the body (for example ultrasound or mammography) MRI tests can cover enormous parts of the body.
- MRI can decide whether a malignancy has spread, and help decide the best treatment.

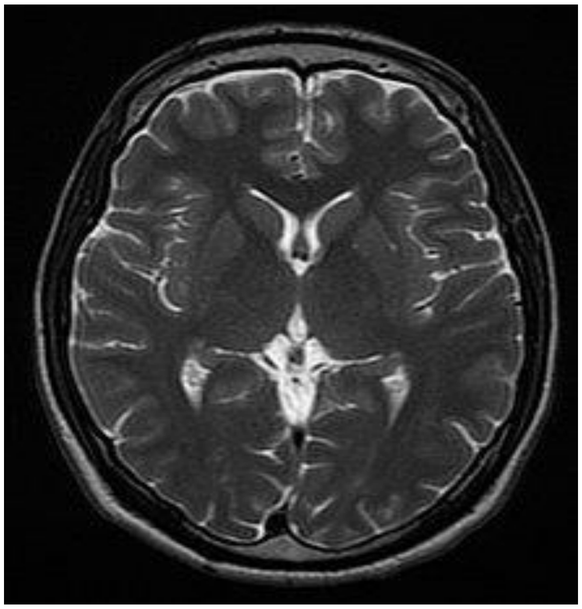


Figure 1.2: MRI Image of Human Brain.[2]

1.2.2 Disadvantages of MRI

- MRI is costly.
- MRI does not have the ability to recognize all cancers (i.e. breast cancers indicated by micro-calcifications).
- MRI cannot always tell if the tumor is malignant or benign (such as breast fibroadenomas), which could lead to a false positive outcome.

[2]Brain MRI Images for Brain Tumor Detection | Kaggle.” https://www.kaggle.com/navoneel/brain-mri-images-for-brain-tumor-detection

- MRI is not difficult, yet the patient must stay still in an encased machine, which might be an issue for claustrophobic patients.
- An undetected metal embed in a patient's body might be influenced by the solid magnet of the MRI unit.
- If a patient chooses to be sedated for the scanning, there is a slight danger related with utilizing the sedation medicine [7].

1.3 Motivation

Brain tumor is a deadly disease. The growth happens at a very restricted space which is why it creates lots of problems. When the tumor grows they cause pressure inside skull to increase, which can cause brain damage and be life-threatening. The limited space and rigidity of the brain makes it hard to detect.

Brain and other nervous system cancer is the 10th leading cause of death for men and women. It is estimated that 18,020 adults (10,190 men and 7,830 women) will die from primary cancerous brain and CNS tumors in 2020 [8].

Like many other disease early detection can increase the survival rate to a great extent. So, many ways are used and everyday many researchers are trying to find new and better ways to detect it. In the recent years CNN is used widely used in this area. The motivation for this research was this opportunity to use convolutional neural network in medical image analysis. MRI images were chosen instead of X-rays or Ct-scan because of its superior image quality also it does not involve any radiation.

1.4 Research Objective

The main objective of this thesis was to observe how the same dataset behave with different types of convolutional neural network. CNN is a huge part of deep learning and there are many different options and opportunities that can be taken to get human level accuracy in computer vision.

The objectives of this work have been:

- To tune a simple CNN architecture to observe how it works on a small dataset.
- To study and apply Transfer Learning.
- To classify brain tumor using simple CNN architecture and Pre-trained models.

- To analyze the results and distinguish among the models.
- To spot the most accurate and convenient model.
- To perceive why the models, give different level of accuracy on unseen test data.

1.5 Thesis Organization

This thesis is organized into seven chapters. Every chapter explicitly describes topics related to the thesis experiment. The overviews of these chapters are given below:

Chapter 1: Introduction

This chapter serves as an introduction of this thesis, the main objective as well as area of work is represented in this chapter. The motivation behind the work is also presented in this chapter.

Chapter 2: Literature Review

This chapter depicts some of the work done by researchers on this topic. Academic personnel all over the world is working every day to improve CNN architecture's use in computer vision. Some of their projects are described in this chapter.

Chapter 3: Convolutional Neural Network

The very basics and the components of the convolutional neural network architecture are narrated in this chapter. It constitutes of the description about the layers of an architecture, the loss functions, the optimization algorithm and the activation functions with their advantages and disadvantages. A brief introduction about the evaluation metrics is also there. This chapter basically shows the various options that can be taken to build a CNN model.

Chapter 4: Transfer Learning

In this chapter the pre-trained models are illustrated. The structure and basic building block of the models used in this thesis are described along with pictures. This chapter is dedicated to have a clear insight on the VGG-16, VGG-19, InceptionV3 and ResNet50 models.

Chapter 5: Methodology

This chapter explains the ways this thesis has followed. A summery about the dataset and how the images of the dataset are processed before they enter into the architecture are described here. Also how the augmentation process helps as well as the data is split are showed in this chapter.

This chapter also has the proposed basic CNN architecture description, the loss function, the activation functions it used as well as the optimizer with equations. Here the number of parameters all the models have is also represented in a table.

Chapter 6: **Experiments and Result Analysis**

This chapter presents the performance of the models. The accuracy graph is attached with every model to have a better understanding about its performance. An analysis about the results is also there which helps to decide which model gives the best and which model gives the worst performance.

Chapter 7: Conclusion and Future Research

This chapter provides a brief summary and concluding explanations regarding the experiments. It also suggests further research directions.

Chapter 02

Literature Review

Literature Review

2.1 Introduction

In recent time the increase of the use of deep learning in medical sector is noteworthy. Its ability to analyze a large amount of data in a short amount of time is the main reason of its popularity. Along with that the accuracy level is also very high. That's why many researchers around the world is working with it and trying to get better results all the time. A minor change in CNN architecture can result in huge change in accuracy level and there are so many different things to try like change in filter size, change in learning rate, different kind of pooling layers, activation functions, different datasets of various size. As the area of work is huge and research material is quite available, many researchers are getting interested in this field.

Amount of research done on brain tumor is also to a great extent. X-ray images, MRI images and CT scan images are generally used for brain tumor research. From some architectures only classification can be achieved, meaning they only tell if the image has tumor or not. But it is still unknown that what kind of tumor it is. A number of research is also done on architectures which tells what type of tumor is present in the image.

2.2 Related Works

Authors Hassan Ali Khan, Wu Jue, Muhammad Mushtaq and Muhammad Umer Mustaq proposed a CNN architecture, compared the result with three pre-trained models (VGG16, ResNet50, InceptionV3) and got a better result for their model [9].

They used Brain MRI Images for Brain Tumor Detection by Novoneel Chakrabarty as dataset and applied data augmentation on it to increase the size. It gives a bigger sample to work with and prevents overfitting. Canny Edge Detection [10] was used to crop the image as a pre-processing technique.

Their proposed model has an input size of 224*224*3. The image goes through 8 convolution layers, 4 max-pool layers and 1 fully connected layer before finally giving yes or no as an answer. The number of filters starts small, only 16, and as the model becomes deep the number of filters increase and it is 128 at the last max-pool layer.

For the convolution layers ReLu was used as activation function. The last fully connected layer uses sigmoid function as it is a binary classification problem and sigmoid gives result as a probability of something. They used Adam optimizer and loss function was Binary Cross Entropy. They also used transfer learning and trained the dataset with three pre-trained models. Proposed model had 100% accuracy while VGG16 had 96% and ResNet50 had 89% finally InceptionV3 had 75% accuracy. Their model was better in other evolution metrics too. Precision, Recall, F1-score and ROC-AUC score all were 100% which is much better than other models.

The paper authored by Priyansh Saxena, Akshat Maheshwari, Shivani Tayal, Saumil Maheshwari [11] applied VGG16, InceptionV3 and ResNet50 on the dataset by Novoneel Chakrabarty.
After preprocessing the dataset, the MRI images are cropped out of dark corners using crop normalization technique. Following that data augmentation is performed to introduce variations in the data samples by increasing the size of training sets which in turn helped to create a generalizable model and thus reducing overfitting.
ResNet50 had the best F1 score and highest test accuracy with the zero false-negative rate, which is the primary concern in an automatic diagnostic system and thus making it suitable for practical applications. The accuracy for this model is 95%. The area under the ROC curve and Cohen's Kappa coefficient are also highest for ResNet50 model. InceptionV3 suffered from overfitting and is just slightly better than a random classifier with an accuracy of 55%. VGG16 has a result in between with an accuracy of 90%. It is closer to ResNet50 but not as good as it.
Their primary goal was to observe how the deep and trained models work on a small dataset, which was achieved although some results were not as per expectation. Sometimes while working with small datasets and deep architectures the model train on the training images too well and learn the noises as well, so it does not give good performance on the unseen test data. This happened to these researchers. Their model learned the train data very well and as a result the test data was not as per expectation.

The authors Onkar Rajesh Mulay and Hemprashad Yashwant Patil got a very interesting insight about transfer learning in their paper [12].They enhanced the brightness of the image as a pre-processing step. According to them brightening of image plays an important role in classification of Brain Tumor images as the tumor in images seems to appear little lighter than rest of the image so, to enhance it every pixel of the images are made lighter. They also augmented the data to increase its volume as they used the dataset by Novoneel Chakrabarty, which is a rather small one. They used the pre-trained AlexNet architecture but with a twist. They used average pooling instead of max pooling layers, which amplified the result and gave an accuracy of 97.7%.

In their paper authors Mesut Toğaçar, Burhan Ergen, Zafer Cömert proposed a new architecture named BrainMRNet [13], which gives better result than VGG16, AlexNet and GoogleNet.
The BrainMRNet architecture is an end-to-end model. It includes the basic layers of CNN architectures in its main structure like convolution, pooling and others. In addition, the dense layer is used before the classification stage, which acts as a hidden layer. The dense layer is connected to the Softmax function, which produces 0–1 probabilities for each artificial neuron. For that reason, the sum of the probabilities given by the Softmax layer is equal to one.
The input size of BrainMRNet is 224*224. It uses Adam optimizer and a mini batch size of 16. The loss they used is Categorical Cross-entropy. Both the convolution layers and dense layers have ReLu activation function except the output layer, which uses softmax. It also has CBAM blocks and Residual blocks in between.
The accuracy achieved by BrainMRNet is 96.05% where as VGG16 has 84.48%, GoogleNet has 89.66% and AlexNet has 87.93%. The other evolution metrics Precision, F1 Score, Specificity and Sensitivity also gives a better number for the proposed method than the pre-trained models.

In the paper authors Rayene Chelghoum, Ameur Ikhlef, Amina Hameurlaine and Sabir Jacquir. [14] have used multiple architectures. The main focus of their work was transfer learning and they used 9 pre-trained models to train the same dataset. All the other criteria like pre-processing, augmentation and splitting are done the same for every model. So, the result just shows the difference among the models.
The models they used are AlexNet, GoogleNet, VGG-16, VGG-19, ResNet-18, ResNet-50, ResNet-101, ResNet-inception-v2, SENet. VGG-16 and VGG-19 gave the best results. By 50 epochs their accuracy was above 98%. After 90 epochs VGG-16 had 98.71% and VGG-19 had

98.47%. Also, AlexNet had 98.22% accuracy. On the other hand, SENet had 56.66% by 50 epochs, which then rose to 95.18% by 90. It had the lowest accuracy rate.
They also kept a record of the time each model took. The fastest model was AlexNet and the model that took the longest time was ResNet-inception-v2. So, the best performer in this task was AlexNet as it gave almost the best result in the quickest possible time.

Another paper in this area is authored by M. A. Bakr Siddiaue, S. Sakib, M. M. Rahman Khan, A. K. Tanzeem, M. Chowdhury, and N. Yasmin [15]. Here the authors used a dataset that consists of 253 brain MR images where 155 images are reported to have tumors. They proposed a Deep Convolutional Neural Network (DCNN) model and got 96% accuracy overall. The other metrics are Precision = 0.93, Sensitivity = 1.00, and F1-score = 0.97. Along with that, the average precision-recall score of the proposed model is 0.93, Cohen's Kappa 0.91, and AUC 0.95.

There are also many papers that worked on other classification techniques along with CNN and analyzed the results. One such is written by A. Kumar Pandey and K. C. James [16].
In this paper authors applied five different classification techniques including CNN, SVM, Decision Tree, Linear Discriminant Analysis and Logistic Regression. Their objective was to review different types of MRI brain tumor classification techniques and compare their performance measures.
According to them SVM, Decision tree, LDA and Logistic Regression algorithm needed preprocessing like Normalization, Intensity, Shape and Texture feature extraction, feature selection before training the model and PCA helped them to select the vital features but CNN did not need any preprocessing explicitly. They used pre-trained and well tested models. For CNN they used VGG16. Their acquired accuracies are 87.74% for SVM, 79.05% for decision tree, 76.67% for LDA, 96.83% for CNN and 89.72% for logistic regression. Other evolution metrics also has a higher number for CNN. CNN had a precision of 96.77%, recall of 98.03% and F1 score was 97.39%. The lowest performance was by linear discriminant analysis. The precision, recall and F1 score for it was 82.7%, 75.34% and 97.39% respectively. So, they concluded that the best model to use for brain tumor detection is Convolutional Neural Network.

Author Donghyu Kim used transfer learning to propose 2 methods of classification in his paper [17]. His methods were built upon existing classification techniques.
For first method he merged segmentation and classification algorithms. He combined VGG-16 and Resnet-50 and achieved test accuracy of 98% for most of its classifiers/ensembles. The second method was an extended convolutional neural network with added layers and for it the accuracy was 86.30%. Most of the proposed models in this paper outperformed the benchmark VGG-16 model.

Some researchers also work on multi classification. The paper authored by Sultan, Hossam H.Salem, Nancy M.Al-Atabany, Walid they did multi classification using CNN with two different datasets. Their proposed model first classified among Meningioma, Glioma and Pituitary tumors and got 96.13% accuracy. Then using the same model on a different dataset they detected if the tumor is Grade II, III or IV. This time the accuracy was 98.7% [18].

Chapter 03

Convolutional Neural Network

Convolutional Neural Network

Convolutional Neural Networks are deep learning algorithms that can recognize and classify features in images. They are most commonly used in visual image analysis. They have application in image classification, medical image analysis, natural language processing, image recognition, object detection and so on.

3.1 Origin and History

The name convolutional neural network came from the mathematical operation convolution, which is a specialized kind of linear operation. It indicates that these type of neural networks use convolution instead of normal matrix multiplication in at least one of their layers [19].

The first modern CNN was introduced in 1998 by Yann LeCun [20]. In his paper the MNIST dataset [21] was used for handwritten character recognition. It gained popularity and many research was done during the 1900 and early 2000s. By 2012 AlexNet [22] was used and it gave an excellent performance on ImageNet [23]. ImageNet is a public, freely-available dataset of images and their corresponding true labels and it currently includes 14,197,122 images.

3.2 CNN vs Human Brain

Often convolutional neural networks are told to be inspired from human brain as both human visual system and CNNs follow a simple-to-complex hierarchical structure. That means the simplest and most obvious feature is recognized first and then moved on to the more complex and imperceptible ones. But the implementation is quite different as brains are built using neurons and CNNs are built using mathematical operations (the history of CNN). However, many researches are done by comparing them.

Specially in case of object detection [24]. These researches helped enormously to develop self-driving cars, face recognition systems and many other sectors. That's why in present day these systems are near perfect and replacing humans for various tasks.

3.3 CNN Architecture

CNN is a multi-layer neural networks. CNN is feed-forwarding in nature since the output of the current layer becomes the input of the next layer. Neurons in the CNN have learnable parameters such as weights and biases. The network starts training itself with a forward pass. A list of connected layers transforms the input volume into an output volume. The prediction in the output layer is computed as probability that represents class scores. The predicted outcome is then compared with the true result to compute the error. In backpropagation, the computed error generates the gradient that flows in the backward direction. At each step, the parameters are tuned in such a direction that it tries to reduce the previously generated error [25] [26] [27]. This process continues in an iterative way until the model converges.

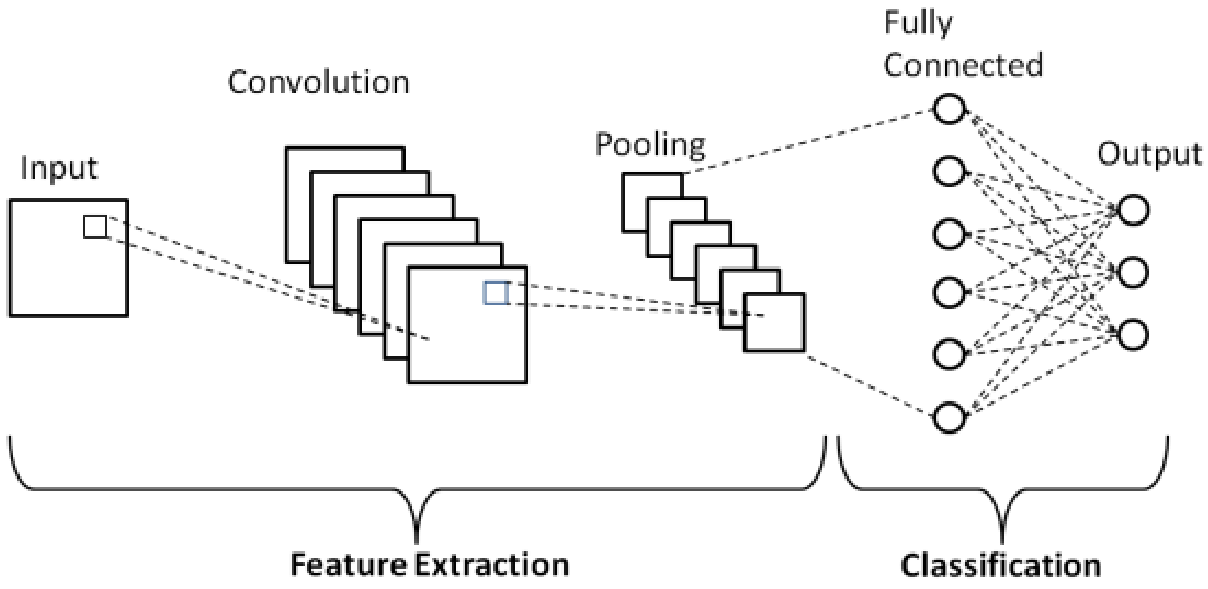


Figure 3.1: Schematic diagram of a basic CNN architecture [28].

3.4 Layers of CNN

A simple ConvNet is a sequence of layers, and every layer of a ConvNet transforms one volume of activations to another through a differentiable function. Three main types of layers are needed to build ConvNet architectures: Convolutional Layer**,** Pooling Layer, and Fully-Connected Layer. Stack these layers together to form a full ConvNet architecture**.**

3.4.1 Convolutional Layer

The convolutional layer is the key component of convolutional neural networks, and is always at least their first layer.

Its purpose is to detect the presence of a set of features in the images received as input. This is done by convolution filtering: the principle is to “drag” a window representing the feature on the image, and to calculate the convolution product between the feature and each portion of the scanned image. A feature is then seen as a filter: the two terms are equivalent in this context [29].

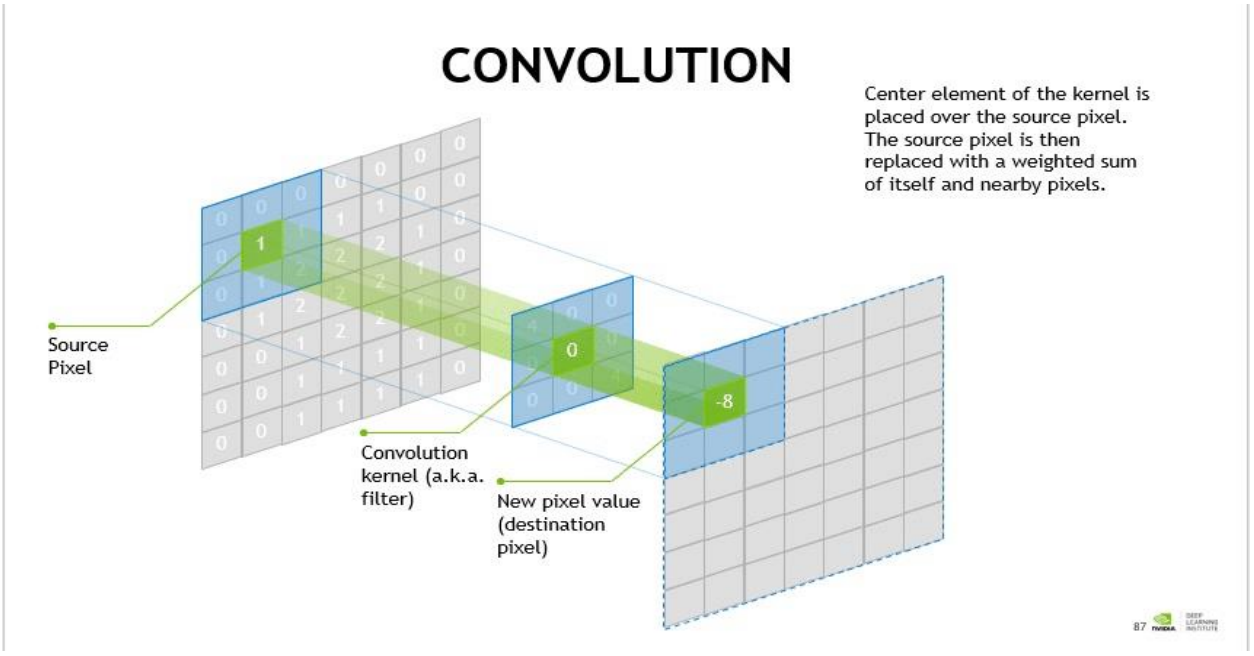

CONVOLUTION
Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.
Source Pixel
Convolution kernel (a.k.a. filter)
New pixel value (destination pixel)

Figure 3.2: Convolution Layer [30].

The convolutional layer thus receives several images as input, and calculates the convolution of each of them with each filter. The filters correspond exactly to the features we want to find in the images. A feature map is obtained for each pair (image, filter), which tells us where the features are in the image: the higher the value, the more the corresponding place in the image resembles the feature. Each feature map is W×H×D, where W is its width in pixels, H is its height in pixels and D the number of channels (1 for a black and white image, 3 for a color image).The convolutional layer has four hyper-parameters:

a) The number of filters K.
b) The size of filters (F): Each filter is of dimensions F×F×D pixels.
c) The S step with which you drag the window corresponding to the filter on the image. For example, a step of 1 means moving the window one pixel at a time.
d) The Zero-padding P: add a black contour of P pixels' thickness to the input image of the layer. Without this contour, the exit dimensions are smaller. Thus, the more convolutional layers are stacked with P=0, the smaller the input image of the network is. A lot of information is lost quickly, which makes the task of extracting features difficult [31].

3.4.2 Pooling Layer

Pooling layer is often placed between two layers of convolution: it receives several feature maps and applies the pooling operation to each of them. Pooling has various forms. For example, Max pooling, Average pooling, General pooling et cetera.

The pooling operation consists in reducing the size of the images while preserving their important characteristics**.** The pooling layer reduces the number of parameters and calculations in the network**.** This improves the efficiency of the network and avoids over-learning [31].The pooling layer has two hyper-parameters:

a) The size of the cells (F): The image is divided into square cells of size F×F pixels.

b) The step (S): Cells are separated from each other by S pixels.

The most popular forms of pooling are Max pooling and Average pooling.

3.4.2.1 Max Pooling

Max pooling is done to in part to help over-fitting by providing an abstracted form of the representation. As well, it reduces the computational cost by reducing the number of parameters to learn and provides basic translation invariance to the internal representation. Max pooling is done by applying a max filter to non-overlapping sub-regions of the initial representation [32].

3.4.2.2 Average Pooling

Average pooling method smooths out the image and hence the sharp features may not be identified when this pooling method is used. Here, rather than a max value, the average for each block is computed.

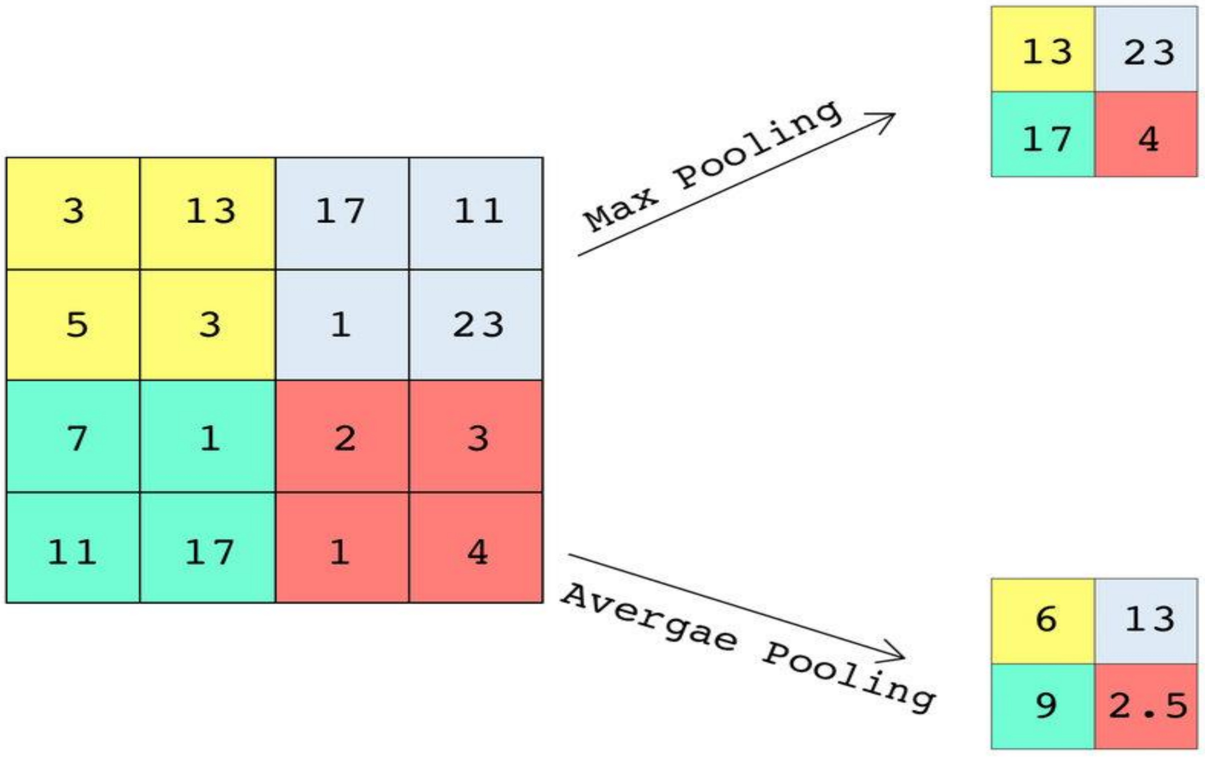


Figure 3.3: Max and Average Pooling [33].

Average Pooling is different from Max Pooling in the sense that it retains much information about the “less important” elements of a block, or pool. Whereas Max Pooling simply throws them away by picking the maximum value, Average Pooling blends them in. This can be useful in a variety of situations, where such information is useful [34].

3.4.3 Fully Connected Layer

The fully connected layer determines the relationship between the position of features in the image and a class. It can be several types [35]:

Fully connected input layer: It “flattens” the outputs generated by previous layers to turn them into a single vector that can be used as an input for the next layer.

Fully connected layer: It applies weights over the input generated by the feature analysis to predict an accurate label.

Fully connected output layer: It generates the final probabilities to determine a class for the image.

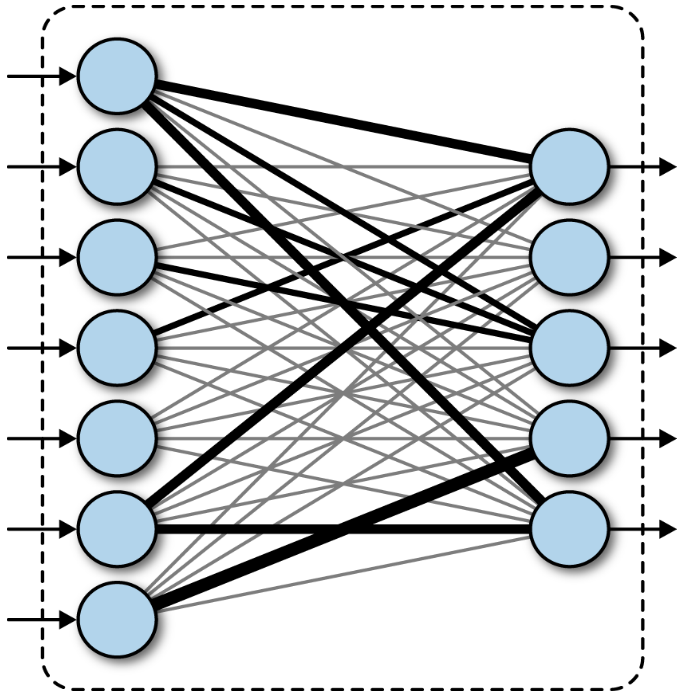


Figure 3.4: A Fully Connected Layer in a CNN [36].

3.5 Loss Function

The Loss Function is one of the important components of Neural Networks. Loss is nothing but a prediction error of the network. And the method to calculate the loss is called Loss Function. In simple words, the loss is used to calculate the gradients. And gradients are used to update the weights of the. This is how a Convolutional Neural Net is trained.

Over the years so many different types of loss functions are being used by the researchers and they got an insight which function is suitable for a certain kind of problem [37].

3.5.1 Mean Squared Error

MSE loss is used for regression tasks. As the name suggests, this loss is calculated by taking the mean of squared differences between actual and predicted values.

For Example, a network which takes house data and predicts house price. In this case, this loss can be used. Basically, in the case where the output is a real number, mean squared error should be used.

3.5.2 Binary Cross Entropy

BCE loss is used for the binary classification tasks. While using this loss function, only one output node is needed to classify the data into two classes. The output value should be passed through a sigmoid activation function and the range of output is (0 – 1).

For example, a convolutional neural network that takes atmosphere data and predicts whether it will rain or not. If the output is greater than 0.5, the network classifies it as rain and if the output is less than 0.5, the network classifies it as not rain. More the probability score value, the more the chance of raining.

3.5.3 Categorical Cross Entropy

When doing multi-class classification task, one of the loss function to go with is this one. If CSE loss function is being used, there must be the same number of output nodes as the classes. And the final layer output should be passed through a softmax activation so that each node output a probability value between (0–1).

For example, a network that takes an image and classifies it into a cat or dog. If the cat node has a high probability score, then the image is classified into a cat otherwise dog. Basically, whichever class node has the highest probability score, the image is classified into that class.

3.5.4 Sparse Categorical Cross Entropy

This loss function is almost similar to CSE except for one change. When using SCCE loss function, one hot encode the target vector is not needed. If the target image is of a cat, simply 0 is passed, otherwise 1. Basically, whichever the class is just the index of that class is passed.

3.6 Optimization Algorithm

Optimizers are algorithms or methods used to change the attributes of convolutional neural networks such as weights and learning rate in order to reduce the losses. They are used to solve optimization problems by minimizing the function [38].

3.6.1 Gradient Descent

Gradient Descent is the most basic but most used optimization algorithm. It’s used heavily in linear regression and classification algorithms. Backpropagation in neural networks also uses a gradient descent algorithm.

Advantages:

a. Easy computation and also easy to implement.

Disadvantages:

a. May trap in local minima.
b. Weights are changed after calculating gradient on the whole dataset. So, if the dataset is too large than this may take years to converge to the minima.
c. Requires large memory to calculate gradient on the whole dataset.

3.6.2 Stochastic Gradient Descent

It’s a variant of Gradient Descent. It tries to update the model’s parameters more frequently. In this, the model parameters are altered after computation of loss on each training example. So, if the dataset contains 1000 rows SGD will update the model parameters 1000 times in one cycle of dataset instead of one time as in Gradient Descent.

Advantages:

a. Frequent updates of model parameters hence, converges in less time.
b. Requires less memory as no need to store values of loss functions.
c. May get new minima’s.

Disadvantages:

a. High variance in model parameters.
b. May shoot even after achieving global minima.
c. To get the same convergence as gradient descent needs to slowly reduce the value of learning rate.

3.6.3 Mini-Batch Gradient Descent

It’s best among all the variations of gradient descent algorithms. It is an improvement on both SGD and standard gradient descent. It updates the model parameters after every batch. So, the dataset is divided into various batches and after every batch, the parameters are updated.

Advantages:

a. Frequently updates the model parameters and also has less variance.
b. Requires medium amount of memory.

Disadvantages:

a. The update of MBGD is much noisy compared to the update of the GD algorithm.
b. Take a longer time to converge than the GD algorithm.
c. May get stuck at local minima [39].

3.6.4 Adam

Adaptive Moment Estimation works with momentums of first and second order. The intuition behind the Adam is not to roll so fast just because jumping over the minimum can happen, the velocity is decreased a little bit for a careful search. In addition to storing an exponentially decaying average of past squared gradients, it also keeps an exponentially decaying average of past gradients. Adam is considered the best optimization algorithm and thoroughly used by researchers. That's why it is used in this thesis and discussed later along with equations.

Advantages:

a. The method is too fast and converges rapidly.
b. Rectifies vanishing learning rate, high variance.

Disadvantages:

a. Computationally costly.

3.7 Activation Function

In neural network the activation function is a mathematical "gate" in between the input feeding the current neuron and its output going to the next layer. It can be as simple as a step function that turns the neuron output on and off, depending on a rule or threshold. Or it can be a transformation that maps the input signals into output signals that are needed for the neural network to function [40].

The Activation Functions can be basically divided into 2 types [41]:

1. Linear Activation Function
2. Non-linear Activation Functions

Non-Linear Activation Functions

Modern neural network models use non-linear activation functions. They allow the model to create complex mappings between the network's inputs and outputs, which are essential for learning and modeling complex data, such as images, video, audio, and data sets which are non-linear or have high dimensionality.

Sigmoid

Advantages

- Smooth gradient, preventing "jumps" in output values.
- Output values bound between 0 and 1, normalizing the output of each neuron.
- Clear predictions-For X above 2 or below -2, tends to bring the Y value (the prediction) to the edge of the curve, very close to 1 or 0. This enables clear predictions.

Disadvantages

- Vanishing gradient-for very high or very low values of X, there is almost no change to the prediction, causing a vanishing gradient problem. This can result in the network refusing to learn further, or being too slow to reach an accurate prediction.
- Outputs not zero centered and computationally expensive.

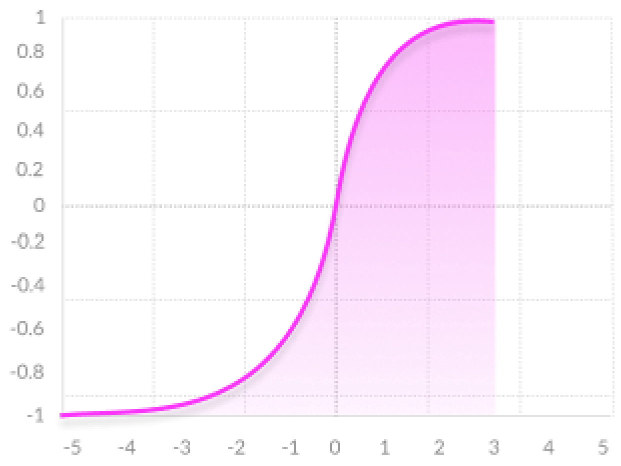


Figure 3.5: Sigmoid [40].

TanH / Hyperbolic Tangent

Zero centered-making it easier to model inputs that have strongly negative, neutral, and strongly positive values.

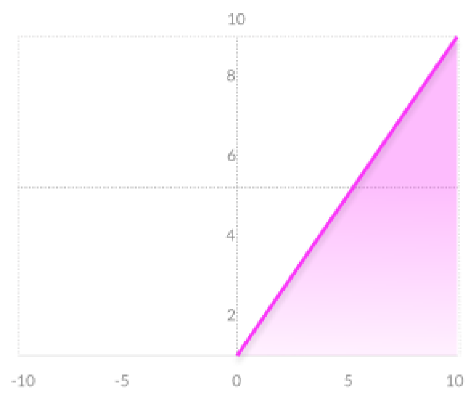


Figure 3.6: TanH [40].

ReLU (Rectified Linear Unit)

Advantages

- Computationally efficient-allows the network to converge very quickly
- **Non-linear**—although it looks like a linear function, ReLU has a derivative function and allows for backpropagation

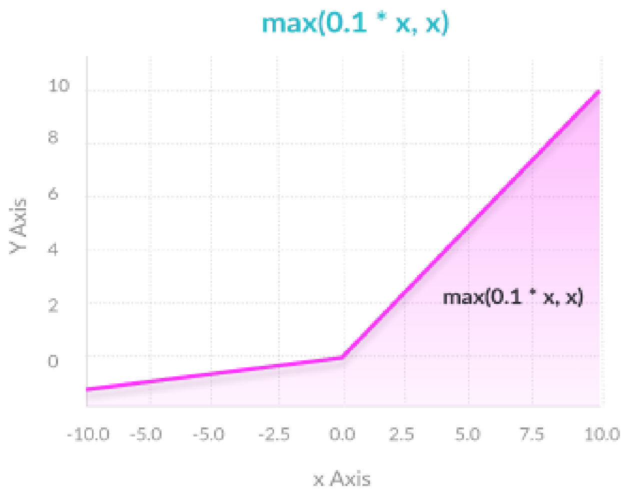


Figure 3.7: ReLu [40].

Disadvantages

- The Dying ReLU problem-when inputs approach zero, or are negative, the gradient of the function becomes zero, the network cannot perform backpropagation and cannot learn.

Leaky ReLU

Advantages

- Prevents dying ReLU problem-this variation of ReLU has a small positive slope in the negative area, so it does enable backpropagation, even for negative input values
- Otherwise like ReLU

Disadvantages

- Leaky ReLU does not provide consistent predictions for negative input values.

Softmax

Advantages

- Able to handle multiple classes only one class in other activation functions-normalizes the outputs for each class between 0 and 1, and divides by their sum, giving the probability of the input value being in a specific class.
- Useful for output neurons-typically Softmax is used only for the output layer, for neural networks that need to classify inputs into multiple categories.

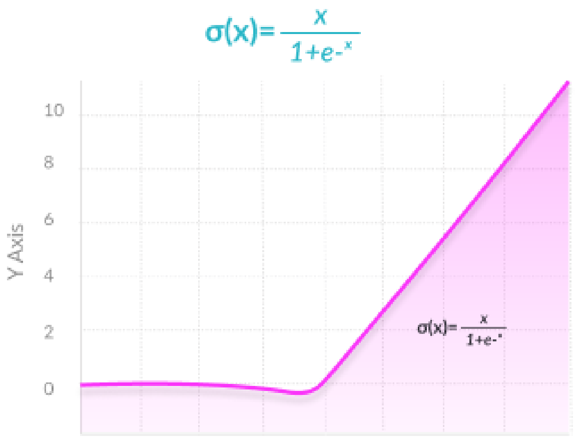


Figure 3.8: Softmax [40].

3.8 Evaluation Metrics

Evaluation metrics are used to measure the quality of the statistical or machine learning model. Evaluating machine learning models or algorithms is essential for any project. It is very important to use multiple evaluation metrics to evaluate a model. This is because a model may perform well using one measurement from one evaluation metric, but may perform poorly using another measurement from another evaluation metric. Using evaluation metrics are critical in ensuring that the model is operating correctly and optimally [42].

Table 3.1: Confusion Matrix for Binary Classification and the Array Representations [43]

	Actual Positive Class	**Actual Negative Class**
Predicted Positive Class	True positive (*tp*)	False negative (*fn*)
Predicted Negative Class	False positive (*fp*)	True negative (*tn*)

Accuracy (acc)

The accuracy metric measures the ratio of correct predictions over the total number of instances evaluated. Formula= $\frac{tp+tn}{tp+fp+tn+fn}$

F1-Score (FM)

The F1-score, is a measure of a model's accuracy on a dataset. The F1-score is a way of combining the precision and recall of the model, and it is defined as the harmonic mean of the model's precision and recall [44]. Mathematically, F1=2 $\times$ $\frac{precision \times recall}{precision+recall}$

Precision (p)

Precision is used to measure the positive patterns that are correctly predicted from the total predicted patterns in a positive class. Precision= $\frac{tp}{tp+fp}$

Recall (r)

Recall is used to measure the fraction of positive patterns that are correctly classified. Mathematically it is, Recall= $\frac{tp}{tp+tn}$

Sensitivity (sn)

This metric is used to measure the fraction of positive patterns that are correctly classified. The formula of it is, Sensitivity= $\frac{tp}{tp+fn}$

Specificity (sp)

This metric is used to measure the fraction of negative patterns that are correctly classified. Mathematically defined as, Specificity= $\frac{tn}{tn+fp}$

Chapter 04

Transfer Learning

Transfer Learning

Transfer learning is a machine learning strategy. For it a model developed for a task is reutilized as the initial point for a model on a different task.
It is a well-known technique in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks. Considering that they reuse the enormous compute and time resources required to develop neural network models. So the provide huge jumps in skill on related problems [45].

4.1 Benefits of Transfer Learning

a. **Higher start**: The initial skill (before refining the model) on the source model is higher than it otherwise would be.
b. **Higher slope**: The rate of improvement of skill during training of the source model is steeper than it otherwise would be.
c. **Higher asymptote**: The converged skill of the trained model is better than it otherwise would be [46].

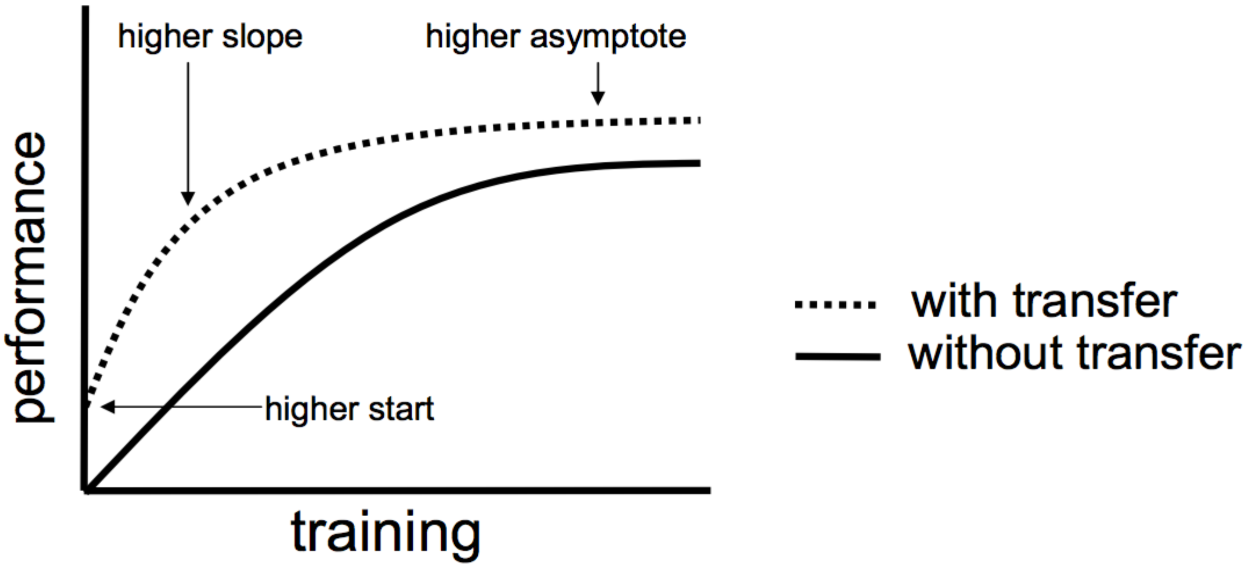


Figure 4.1: Three ways in which model improves learning taken from Transfer Learning [46].

Keras, a widely used open-source library for ANN. It provides a number of pre-trained models to use for prediction, feature extraction, and fine-tuning [47]. Among them VGG-16, VGG-19, InceptionV3 and ResNet50 are used often in brain tumor researches.

4.2 VGG-16

VGG-16 is a convolutional neural network model proposed in the paper titled "Very Deep Convolutional Networks for Large-Scale Image Recognition" authored by K. Simonyan and A. Zisserman from the University of Oxford [48].

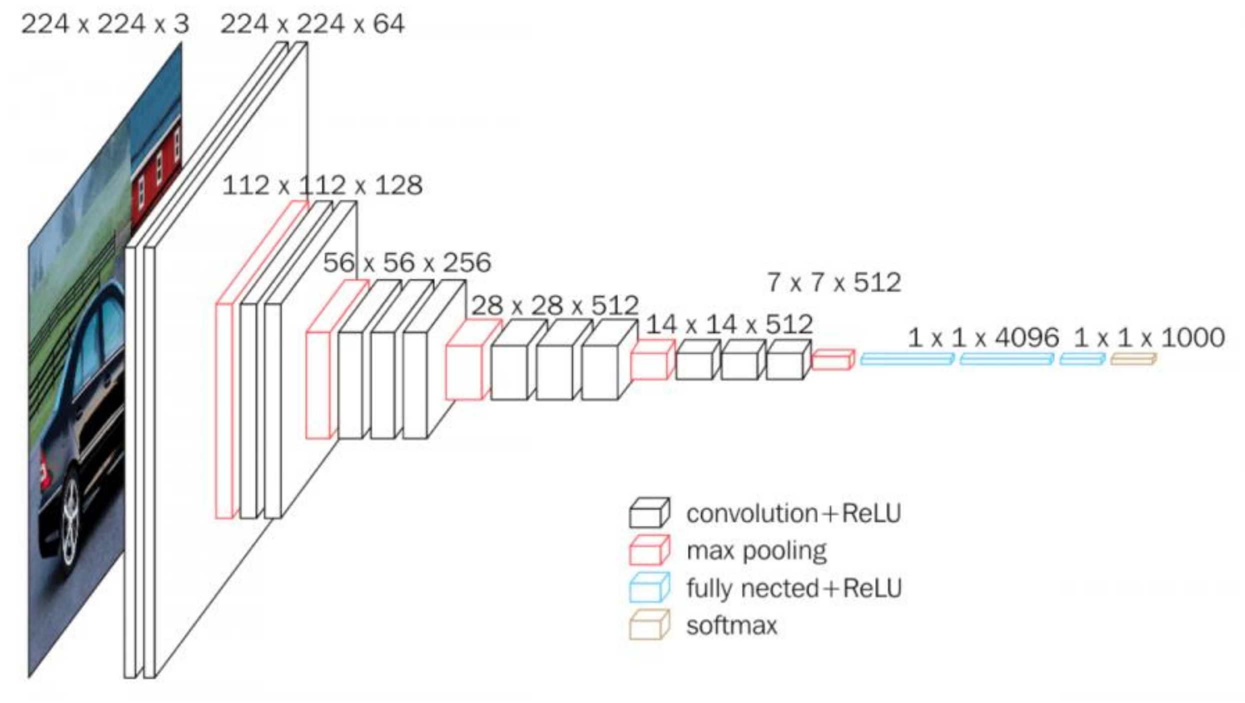


Figure 4.2: The basic architecture of VGG-16 [49].

The input to first convolutional layer is fixed. The size is 224*224*3. The image then goes through a number of convolutional layers. The filter size is as small as 3*3. The stride for convolution is fixed as 1 pixel. The padding of convolutional layer input is designed in such way that the resolution stays same after convolution.

Five max-pooling layers carry out spatial. And it is followed by some of the convolutional layers. But some conv. layers are followed by max-pooling. For max-pooling the window size is 2*2 pixel and it has a stride of 2.

After the convolutional layers comes three Fully-Connected (FC) layers. The first two has 4096 channels each. The third has 1000 channels as it performs 1000-way ILSVRC classification. The last layer is the Softmax layer. All networks have the same configured fully connected layers. The ReLU non-linearity is equipped by all hidden layers .[50]

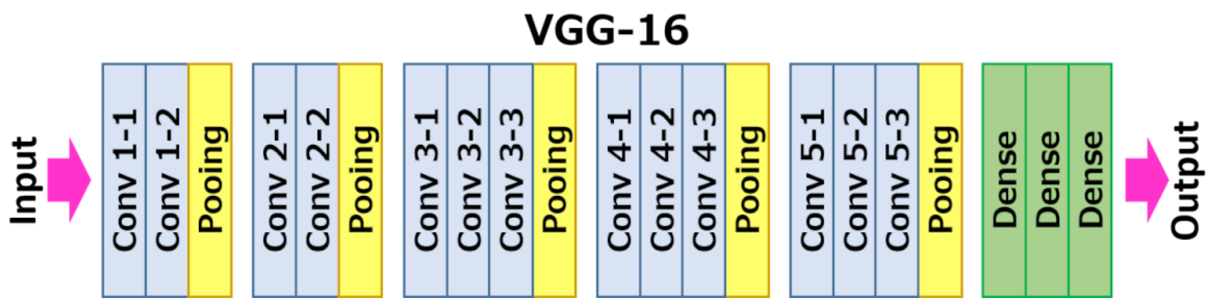


Figure 4.3: Flow diagram of the layers of VGG-16.[3]

4.4 VGG-19

In this network a RGB image of size of (224 * 224) is given as input. So, the shape of the matrix is (224,224,3). Some preprocessing is done, which is the mean RGB value from each pixel is subtracted then the whole training set is computed. The filter is of (3 * 3) size and the stride size is 1 pixel, the entire notion of the image is covered through it. In order to preserve the spatial resolution of the image spatial padding is used. For max-pooling the window size is 2*2 pixel and it has a stride of 2.

After this Rectified linear unit(ReLU) is used to introduce non-linearity. By using it the model can classify better and improve computational time. The previous models used tanh or sigmoid functions and ReLU is proved much better than those. After that comes three Fully-Connected (FC) layers. The first two has 4096 channels each. The third has 1000 channels as it performs 1000-way ILSVRC classification. The last layer is the Softmax layer[51].

[3]"VGG16 - Convolutional Network for Classification and Detection." https://neurohive.io/en/popular-networks/vgg16/

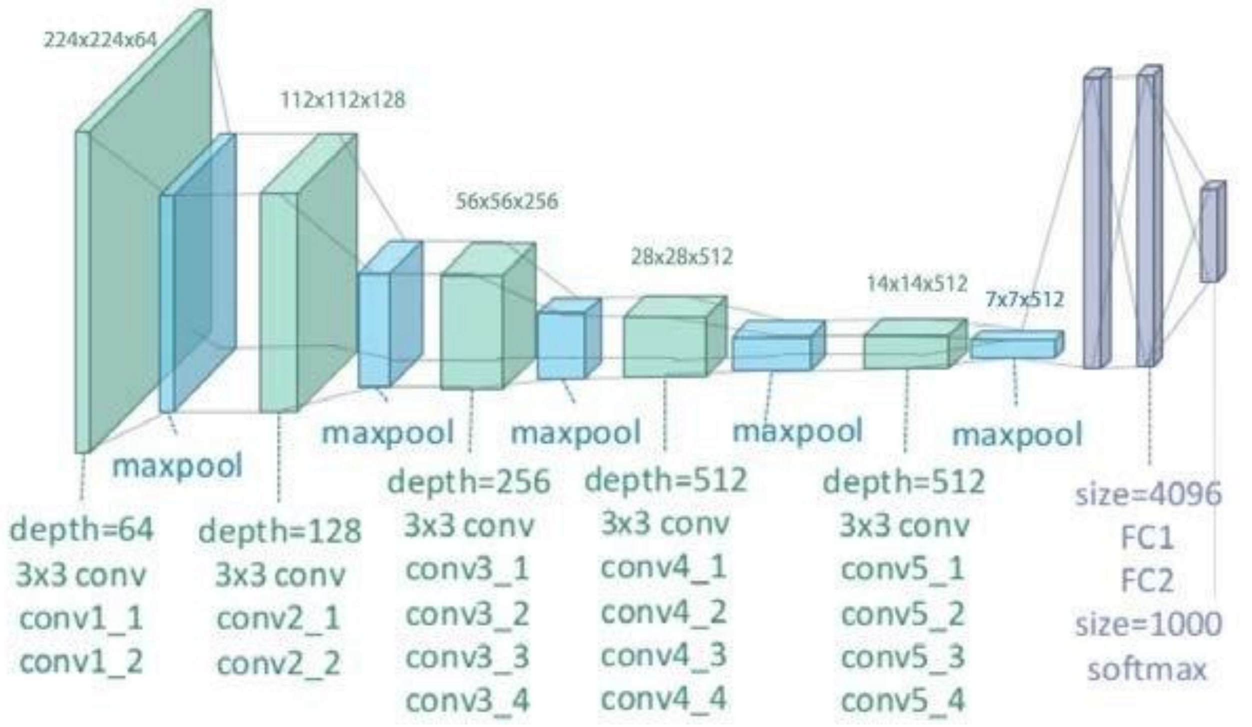


Figure 4.4: Network architecture of VGG-19 model [52].

4.3 InceptionV3

The most recent used inception architecture was first introduced by Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, Zbigniew Wojna in their paper “Rethinking the Inception Architecture for Computer Vision” [53].

The building block of this model is known as an "Inception cell". In an inception call a number of convolutions is performed at the same time and consequently totaled the outcomes. To save calculation, 1x1 convolutions are utilized to decrease the depth of the input channel. For every cell, some 1x1, 3x3, and 5x5 filters are found which can learn to extract features at different scales from the input [54].

In this way, as opposed to picking a clear filter size and working with it, inception works with every option possible. It doesn’t miss out on anything and that is the best advantage of it.

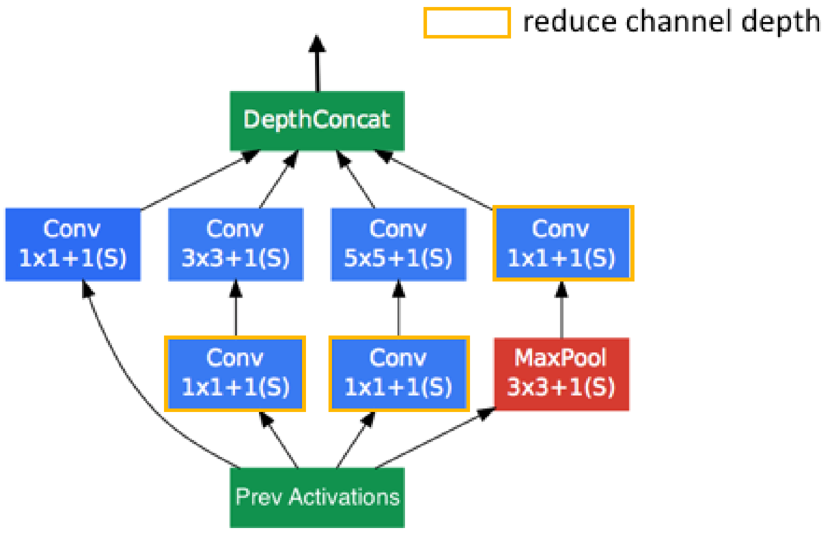


Figure 4.5: Inception Cell [54].

The primary version V1 has 5 million parameters and the latest version of inception which is V3 has 23 million parameters and the layer depth is 159 [55].

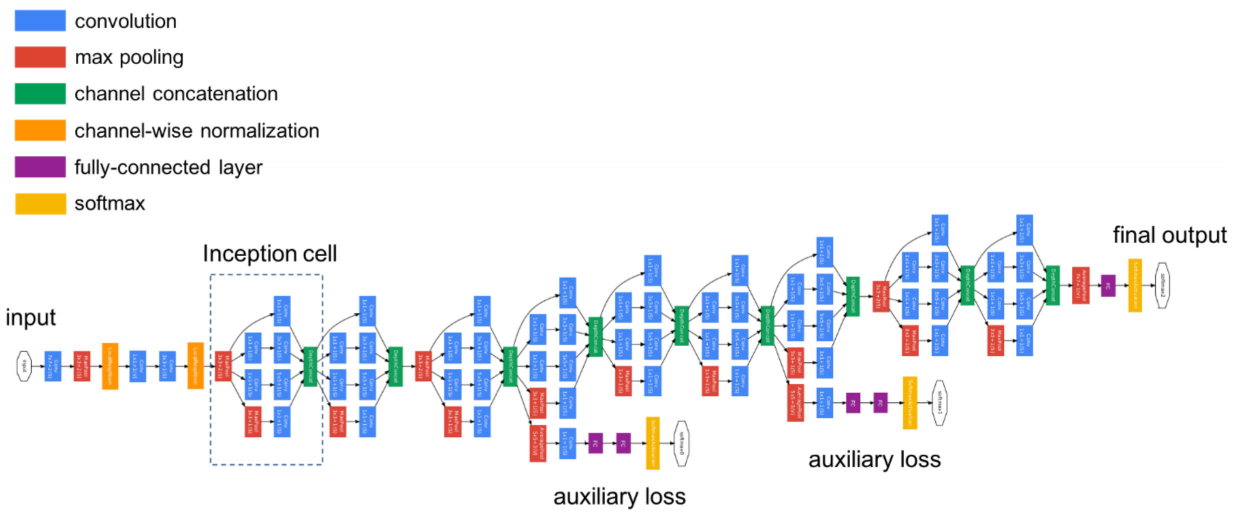


Figure 4.6: Inception Model.[4]

[4] "Inception Network | Implementation Of GoogleNet In Keras." https://www.analyticsvidhya.com/blog/2018/10/understanding-inception-network-from-scratch

4.5 ResNet50

ResNet was introduced in the paper "Deep residual learning for image recognition" [56] and it was a ground breaking work. The basic building block of ResNet is residual blocks. They are not like other neural networks. In traditional neural networks, each layer's output goes into the next layer as an input. But in a network that has residual blocks, each layer feeds into the next also into the layers which are 2-3 jumps ahead of it.

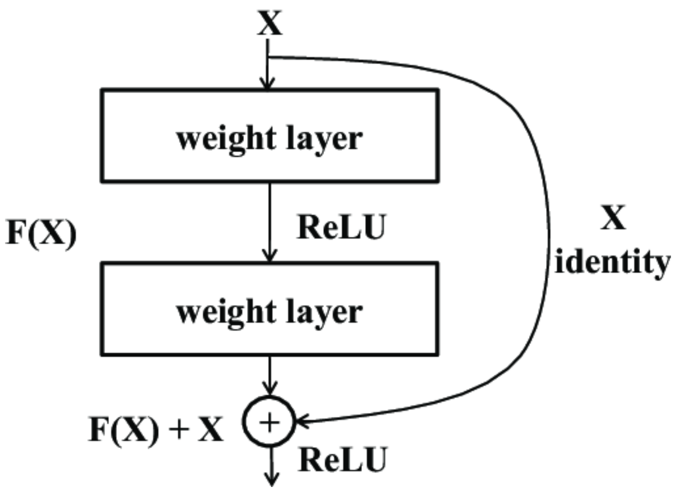


Figure 4.7: Residual Block.[5]

ResNet50 architecture has the following layers:

- A convolution with a filter size of 7 * 7 and 64 different filters. The stride size is 2 pixels. Then comes a max pooling layer which also has a stride size of 2.
- In the next convolution there is a 1 * 1,64 filter prior to a 3 * 3,64 filter. After that a 1 * 1,256 filter. These three layers are repeated 3 time that makes 9 layers in this step.
- Next a filter of 1 * 1,128 after that a filter of 3 * 3,128 and at last a filter of 1 * 1,512. These three layers are repeated 4 time that makes 12 layers in this step.
- After that there is a filter of 1 * 1,256 and two more filters with 3 * 3,256 and 1 * 1,1024 and this is repeated 6 time. So, a total of 18 layer**s**.

[5] "Illustration of a residual block. Residual learning: a building block." https://www.researchgate.net/figure/llustration-of-a-residual-block-Residual-learning-a-building-block_fig3_331840514

- And then again a 1 * 1,512 filter with two more of 3 * 3,512 and 1 * 1,2048 and this was repeated 3 times having a total of 9 layers.
- After that an average pool and it ends with a fully connected layer containing 1000 nodes and at the end a softmax function so this counts as 1 layer.

Without counting the activation functions and the max/ average pooling layers. Here is a (1 + 9 + 12 + 18 + 9 + 1) 50 layers' deep convolutional network [57].

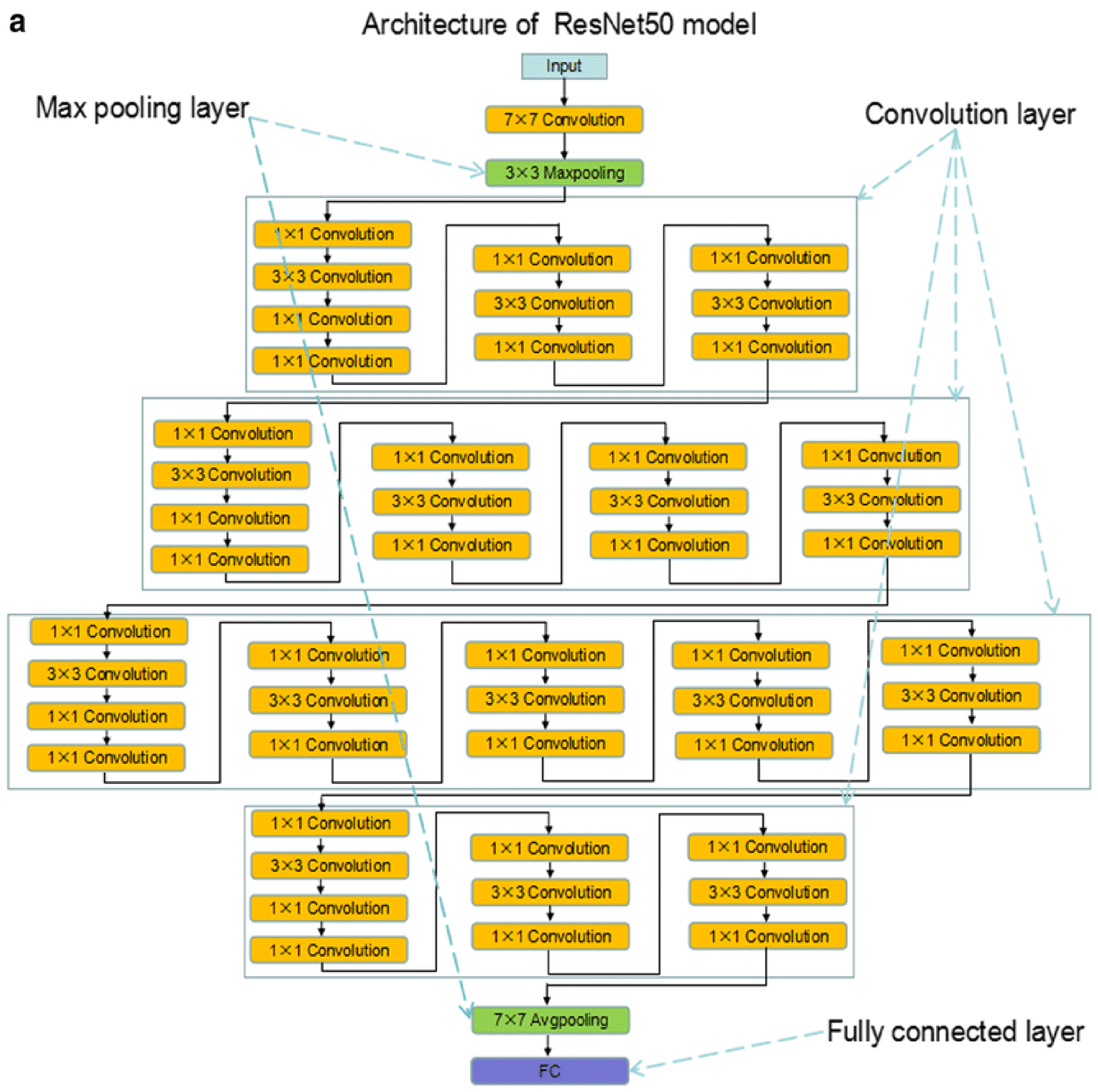


Figure 4.8: A ResNet50 architecture [58].

Chapter 05

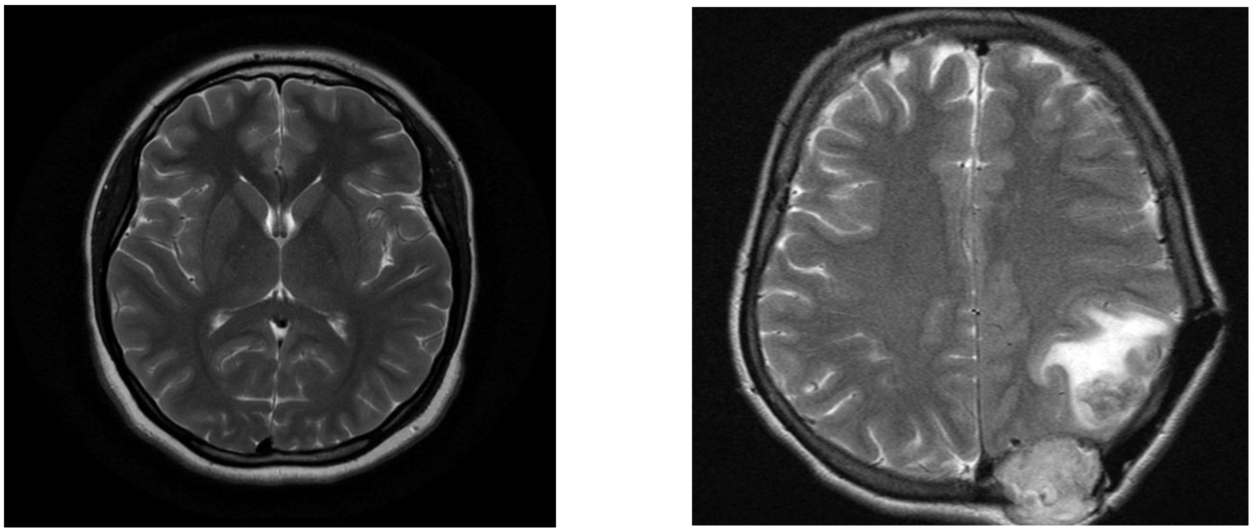
Methodology

Methodology

5.1 Dataset

The title of the dataset used is Brain MRI Images for Brain Tumor Detection. It consists of free accessible MRI images. It has 253 images in two folders named yes and no. The no folder has 98 images and the yes folder has 155 images [59].

Field experts, such as doctors and radiologists collected the images and shared on the internet. Each picture was gotten from the volunteer patients. Accordingly, the dataset has a heterogeneous structure. As, the number of tumor images is 155 while the number of normal samples is 98. The resolution of the images is not steady and the image quality is not good. Sample images from the dataset are as following:



(a) (b)

Figure 5.1: (a) Normal Sample and (b) Tumor Sample.[6]

[6] Brain MRI Images for Brain Tumor Detection | Kaggle.” https://www.kaggle.com/navoneel/brain-mri-images-for-brain-tumor-detection

5.2 Data Preprocess

Data preprocessing can be termed as a data mining technique which involves the transformation of raw data to a format which is more interpretable and makes the images more suitable for further processing. The following pre-processing steps were involved:

5.2.1 Resizing of Images

The input dataset contains images with a different dimensions and with different aspect ratio. Therefore, the images in the dataset are resized to a preset format, which is (240,240,3) = (image_width, image_height, number of channels).

5.2.2 Crop Normalization

It is the approach of determining of extreme points in contours. This method is used to determine the farthest north, south, east, and west (x, y)-coordinates along a given contour. This technique applies to both raw contours and rotated bounding boxes. Using this technique, only the portion of the image containing the brain is cropped out using a Computer Vision (CV) technique given by Adrian Rosebrock [60].

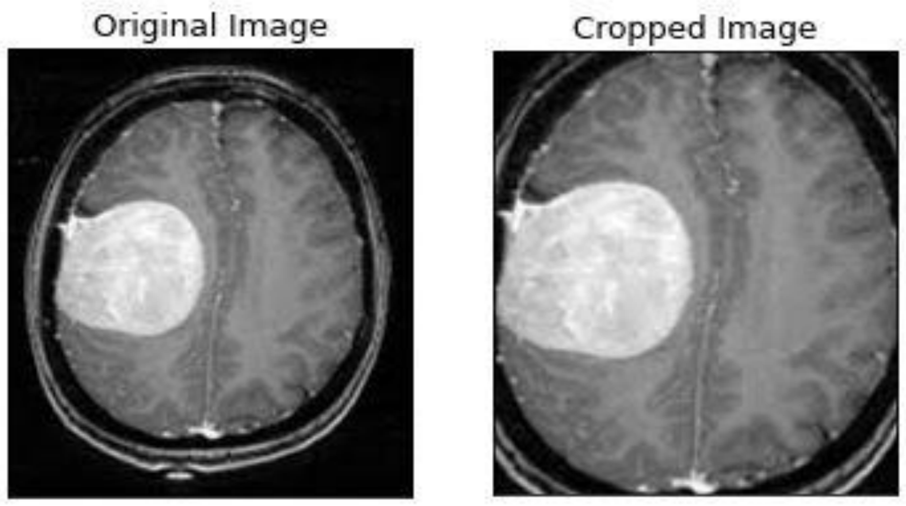


Figure 5.2: Original vs Cropped Image.

5.3 Data augmentation

Deep neural networks provide a significant boost in performance and produce skillful models whenever trained on more data. It is a method to artificially enhance the size of image training data by generating modified images using the original dataset.

By using this technique, variations are introduced in the images which enhance the capability of the model to learn and generalize better on future unseen data. So, by introducing variation in the training dataset, the model becomes generalized and in turn, becomes less prone to overfitting [61] ImageDataGenerator () function is used to augment the data and the arguments used are rotation_range, width_shift_range, height_shift_range, shear_range, brightness_range, horizontal_flip, vertical_flip, fill_mode, preprocessing_function. After augmentation the dataset has 2065 images among which 1085 are positive examples and 980 negative examples. Also, the data is more balanced subsequent augmentation.

Table 5.1: Ratio of Positive and Negative Examples Before and After Augmentation.

	Percentage of positive examples (%)	**Percentage of negative examples (%)**
Before Augmentation	61.26	38.73
After Augmentation	52.54	47.45

5.4 Data Split

In this step, the complete dataset is divided into three segments, namely, Train, Test and Validation. Train data comprises the data sample that is used for fitting the model. Validation data is that sample of the data, which helps in providing an unbiased evaluation of the model which was trained on the training data along with tuning the hyper-parameters of the model. Test data includes those samples of data which provides an unbiased evaluation of a final model that was trained on training data.

So, 70% data is kept as training data to train the model. Another, 15% are kept as validation data and the remaining 15% are used to test the accuracy of the model. That means 1445 images are for the training set, 310 images for the validation set and 310 images for the test set.

5.5 CNN Model

In proposed CNN model, each input x (image) has a shape of (240, 240, 3) and is fed into the neural network. And, it goes through a zero padding layer with a pool size of (2, 2). Then, a convolutional layer with 32 filters, with a filter size of (7, 7) and a stride equal to 1. After that a batch normalization layer to normalize pixel values to speed up computation and ReLU activation layer. Then 2 max pooling one after another f = 4 and s = 4 for both of them. Next, a flatten layer in order to flatten the 3-dimensional matrix into a one-dimensional vector. Finally, a dense fully connected layer with one neuron with a sigmoid activation. The basic model has 1 convolution and 2 max-pool layers. It was then tuned with 2 convolution layers and 3 max-pool layers as well as 3 convolution layers and 4 max-pool layers keeping all the other variables same as before.

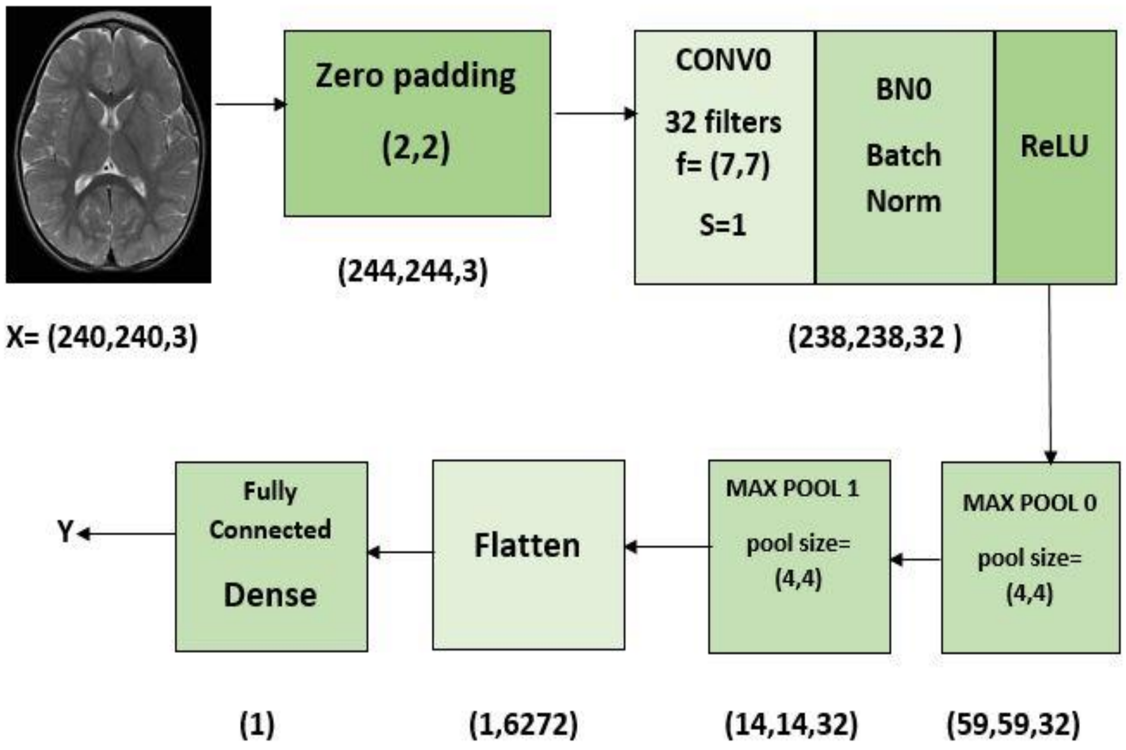


Figure 5.3: Proposed CNN Architecture.

5.5.1 Loss Function

In machine learning loss function is used to calculate the error between the algorithmic predicted values and the true label values. Then this error is minimized by using any optimization method. In this experiment Cross Entropy loss function is used, which was presented by Shie Manor [62]. As binary classification of our MRI Images are done so binary cross entropy is used. In binary cross entropy error rate is calculated between 0 and 1. Mathematically it is represented as:

$$J(z) = [y\, log\, P(y) + (1 - y) \log(1 - P(y)\,)] \tag{5.1}$$

Here y is the actual labels and P(y) is the predicted labels. So when the actual labels y will be 0 then the first term will be zero because y is multiplying with log. And when y will be 1 then second term will be zero (1-y) will be zero and it is multiplied with log. And if y = P(y) then J(y)(Loss) will be zero.

5.5.2 Optimizer

In deep neural networks, different optimization methods are applied by adjusting parameters such as weights and learning rates to reduce the loss. In this experiment, Adaptive Moment Estimation(Adam) optimizer, proposed by Diederik Kingma is used [63]. Adam optimizer is a combination of RMSprop and Stochastic Gradient Descent with momentum.

The Stochastic Gradient Descent method is proposed by Herbert and Sutton [64]. In Stochastic Gradient Descent, the derivative of weights is taken, dW, and derivative of Bias, db for each epoch. And multiplied with the learning rate.

$$W = W - \eta \times dW \tag{5.2}$$

$$b = b - \eta \times db \tag{5.3}$$

While Stochastic Gradient Descent with momentum V is the moving mean of gradients, here is the moving mean between 0 and 1 when dW and db on the current batch is calculated,

$$V_{dw} = \beta \times V_{dw} + (1 - \beta) \times dW \tag{5.4}$$

$$V_{db} = \beta \times V_{db} + (1 - \beta\,) \times db \tag{5.5}$$

Similarly, Root Mean Squared Prop is an adaptive learning rate methodology presented by Geoff, Hinton [65]. In RMSProp, the exponential moving mean square of gradients are taken. So in RMSProp,

$$S_{dW} = \beta \times S_{dW} + (1-\beta) \times dW^2 \tag{5.6}$$

$$S_{db} = \beta \times S_{db} + (1-\beta) \times db^2 \tag{5.7}$$

Beta β, is a hyper-parameter that controls exponentially weighted means. So combining the features of the weighted mean of past gradients and the weighted mean of the squares of the past gradients Adam optimization technique is implemented, So the updated weights and bias in Adam optimizer will be,

$$W = W - \eta \times \left(V_{dW} / \sqrt{S_{dW+\epsilon}} \right) \tag{5.8}$$

$$b = b - \eta \times \left(V_{db} / \sqrt{S_{db+\epsilon}} \right) \tag{5.9}$$

Epsilon ϵ, is a small number that prevents zero division (Epsilon = 10^{-7}) and β is a learning rate with a different range of values. In this experiment learning rate is 0.001.

5.5.3 Activation Function

Relu

Rectified Linear Unit (ReLU) activation function is applied in each convolutional layer. An activation function converts the input weighted sum into that node's output represented by Vinod and Hinton [66]. Rectifier Linear unit function is often used in hidden layers of the convolutional neural network. Mathematically, ReLU is represented by,

$$f(z) = \max\ (0, z) \tag{5.10}$$

Where z is the input when z is negative or equal to 0, it transforms the negative input to 0. When the input is greater than 0, then the output will be 1. So the derivative of ReLU will be,

$$f'^{(z)} = \begin{cases} 1\,, for\ z \geq 0 \\ 0, for\ z < 0 \end{cases} \tag{5.11}$$

So if the input is 0 then that neuron is a dead neuron in ReLU function and it won't be activated.

Sigmoid

Another activation function used is the sigmoid function. It is applied in the fully connected layer and it helps with binary classification as it takes any real-valued input, and outputs a value between zero and one. Mathematically, it is represented as,

$$s\ (x) = \frac{1}{1 + e^{-x}} = \frac{e^{x}}{e^{x} + 1} \tag{5.12}$$

The main advantage of sigmoid function is it exists between zero to one**.** Therefore, it is especially used for models where the probability is predicted as an output. Since probability of anything exists only between the range of 0 and 1**,** sigmoid is the right choice.

5.6 Pre-trained Models

In this experiment four pre-trained models are used. They are VGG-16, VGG-19, InceptionV3, ResNet50. Their Keras pre-trained model is used which was trained on ImageNet dataset. The base model was used and no other flatten layer was added at the end of the model. Every model has its own pre-processing function and input while training them all these criteria's were fulfilled to get the best performance from the models.

As the models are very deep with lots of hidden layers, so dropout is used to avoid over-fitting. The probability value used as dropout is 50%. Other than that, the epochs play a role on the performance of the models. Every model is trained in 24 epochs without InceptionV3. It was trained with only 10 epochs due to hardware drawbacks.

5.7 Parameters of different CNN

Every CNN model has different number of parameters. Some of them are trainable and some are not. The deeper the model the more the number of parameters. The run from as small as 64 non-trainable to as big as millions. A table is given below to get a clear understanding on it.

Table 5.2: Number of Parameters of Different CNN Models

CNN Architecture	Total Parameters	Trainable Parameters	Non-Trainable Parameters
1 convolution layer 2 max-pool layers	11,137	11,073	64
2 convolution layers 3 max-pool layers	31,777	31,713	64
3 convolution layers 4 max-pool layers	116,257	116,193	64
VGG-16	14,739,777	25,089	14,714,688
VGG-19	20,049,473	25,089	20,024,384
InceptionV3	21,933,857	131,073	21,802,784
ResNet50	23,688,065	100,353	23,587,712

5.8 Hyperparameters

Some parameters are not changeable or trainable by the model as they are inputted manually. They are set before training, before optimizing weights and bias. They determine how the network is trained and what is the structure of the model. For example, the number of hidden units, learning rate, dropout rate et cetera.

Table 5.3: Different Hyperparameters used in the Research.

Name of the hyperparameter	Value
Batch Size	32
Dropout Rate	0.5
Learning Rate	0.001
Maximum Epochs	24 (Except InceptionV3)

Chapter 06

Experimental Results and Analysis

Experimental Results and Analysis

6.1 Result Without Data augmentation

The basic convolutional architecture does not perform well without any data augmentation with only the 253 images the dataset offers. It over-fits while training so the train accuracy is quite good but the difference between accuracy of train set and validation set is noticeable.

Accuracy

Train accuracy = 98.87 %

Validation accuracy = 79 %

Test accuracy = 84.21 %

F1 Score

Validation = 84 %

Test = 87.49 %

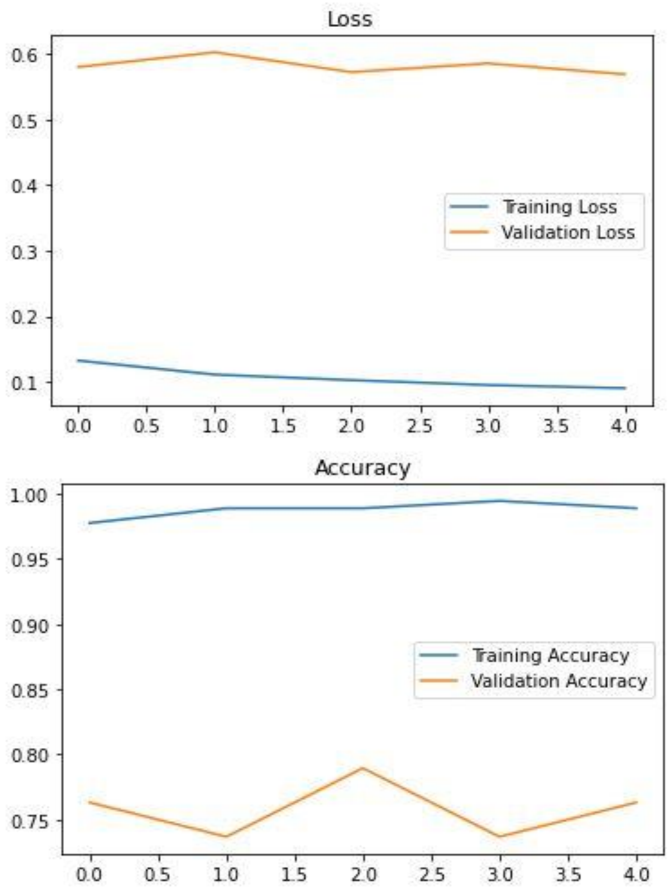


Figure 6.1: Accuracy and Loss Graphs for Original Dataset.

6.2 Result of CNN with 1 Convolutional Layer

After augmentation the same architecture performs better. Though over-fitting still happens the difference between accuracy of test and validation set decreases a lot.

Accuracy

Train accuracy = 96.89 %

Validation accuracy = 87 %

Test accuracy = 88.39 %

F1 Score

Validation = 87.15 %

Test = 88.16 %

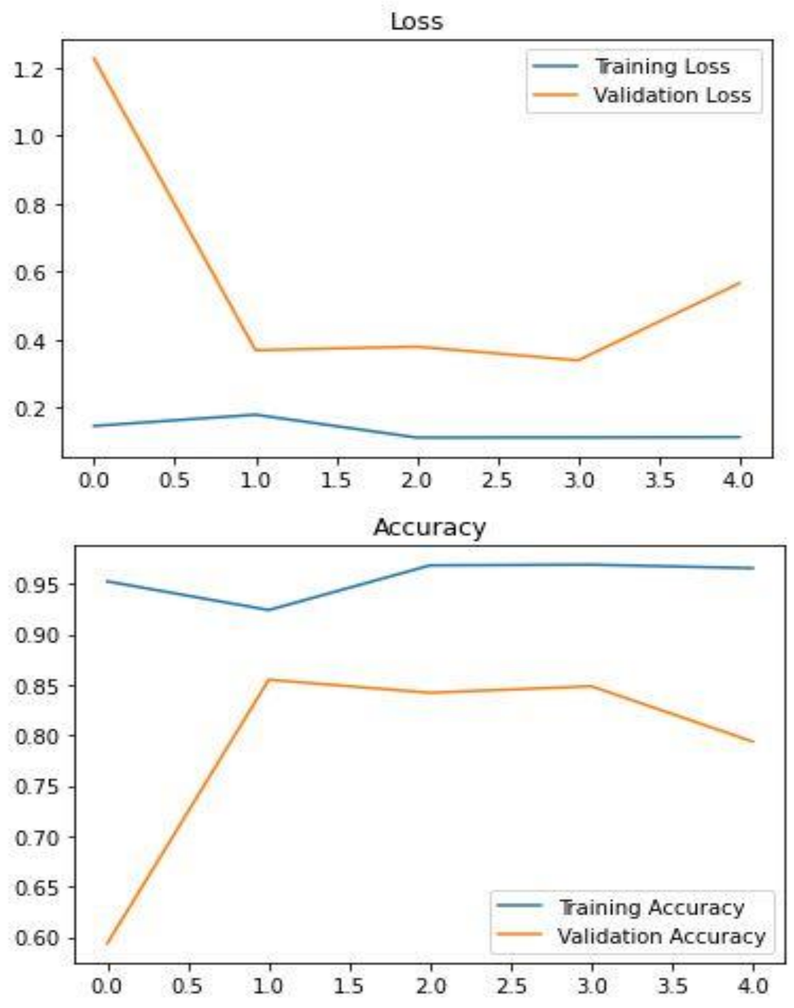


Figure 6.2: Accuracy and Loss Graphs for Augmented Data and Single Conv Layer.

6.3 Result of CNN with 2 Convolutional Layers

In this architecture the data goes through 2 convolutional and 3 max-pooling layers before entering the fully connected layer. The accuracy for training set increases but all the other values drop after this change.

Accuracy

Train accuracy = 100 %

Validation accuracy = 86 %

Test accuracy = 87.10 %

F1 Score

Validation = 86.62 %

Test = 86.58 %

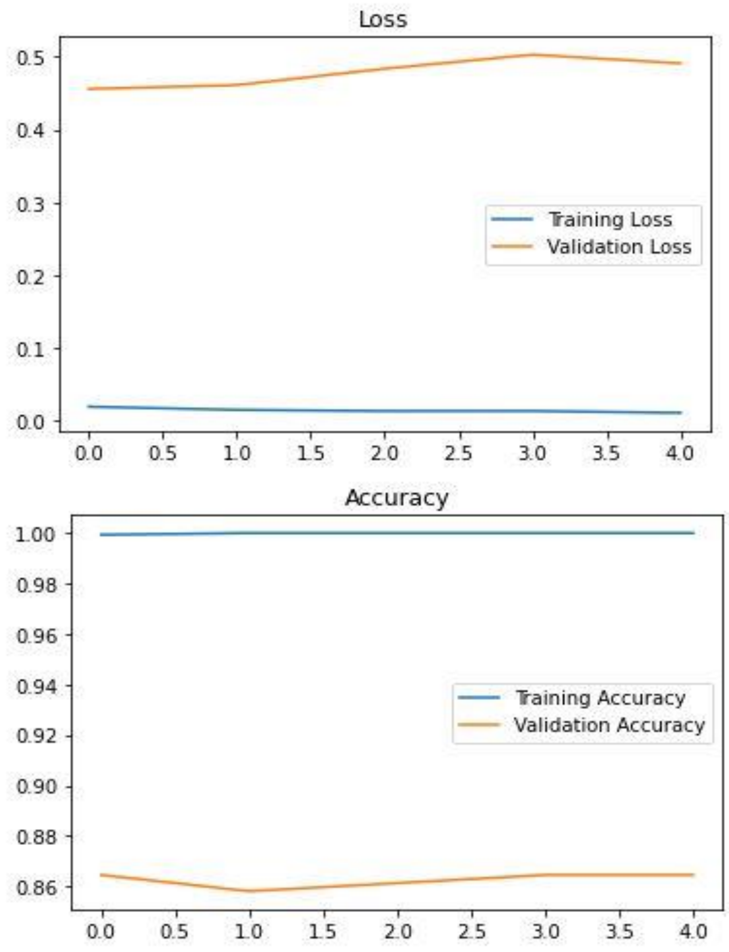


Figure 6.3: Accuracy and Loss Graphs for 2 Conv Layers.

6.4 Result of CNN with 3 Convolutional Layers

Here 3 convolutional and 4 max-pooling layers are used. The accuracy for training set remains same but the other accuracies and also the F-measures decreases.

Accuracy

Train accuracy = 100 %

Validation accuracy = 85 %

Test accuracy = 82.85 %

F1 Score

Validation = 84.62 %

Test = 83.89 %

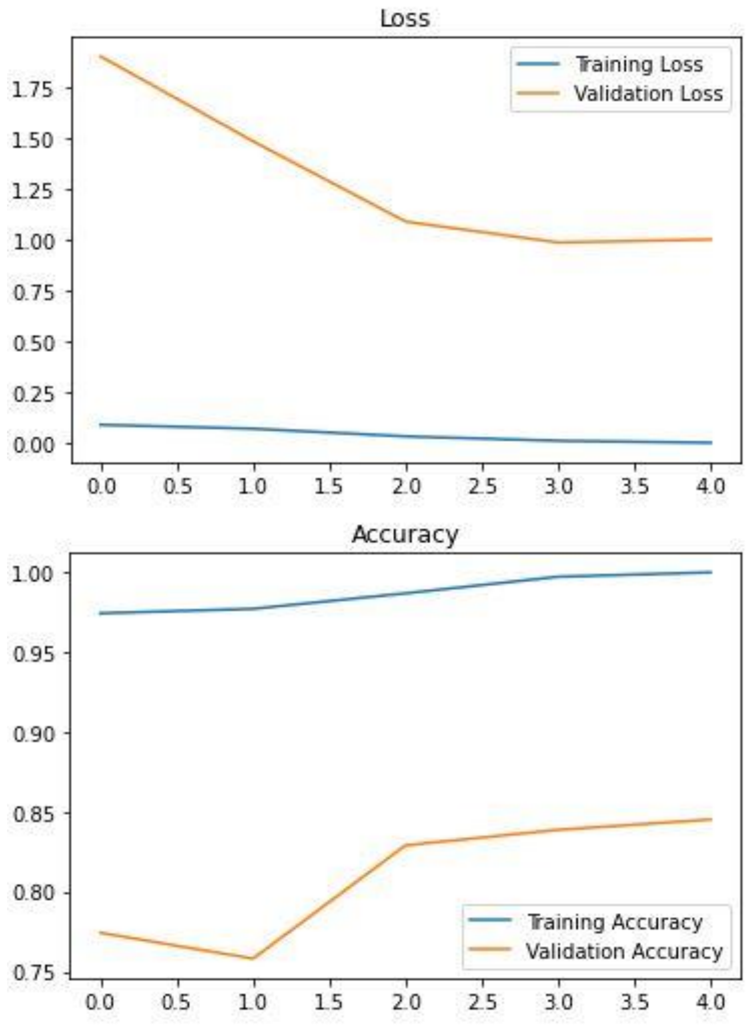


Figure 6.4: Accuracy and Loss Graphs for 3 Conv Layers.

6.5 Result of VGG-16

It showed the best result among the pre-trained models for both train and validation set.

Accuracy

Train accuracy = 99.93 %

Validation accuracy = 98.71 %

Test accuracy = 97.42 %

F1 Score

Validation = 98.72 %

Test = 97.35 %

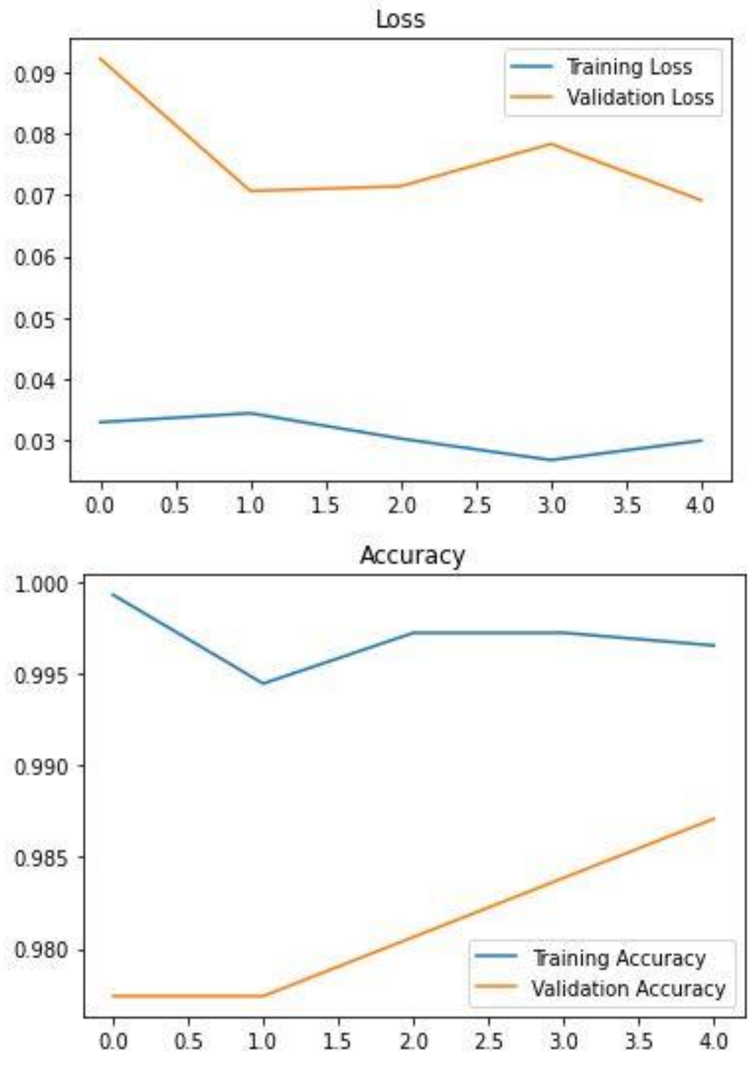


Figure 6.5: Accuracy and Loss graphs for VGG-16.

6.6 Result of VGG-19

This model shows the least difference between validation set and test set accuracy. Though the difference between them and train set is more than VGG-16 model.

Accuracy

Train accuracy = 99.45 %

Validation accuracy = 97.42 %

Test accuracy = 97.10 %

F1 Score

Validation = 97.61 %

Test = 97 %

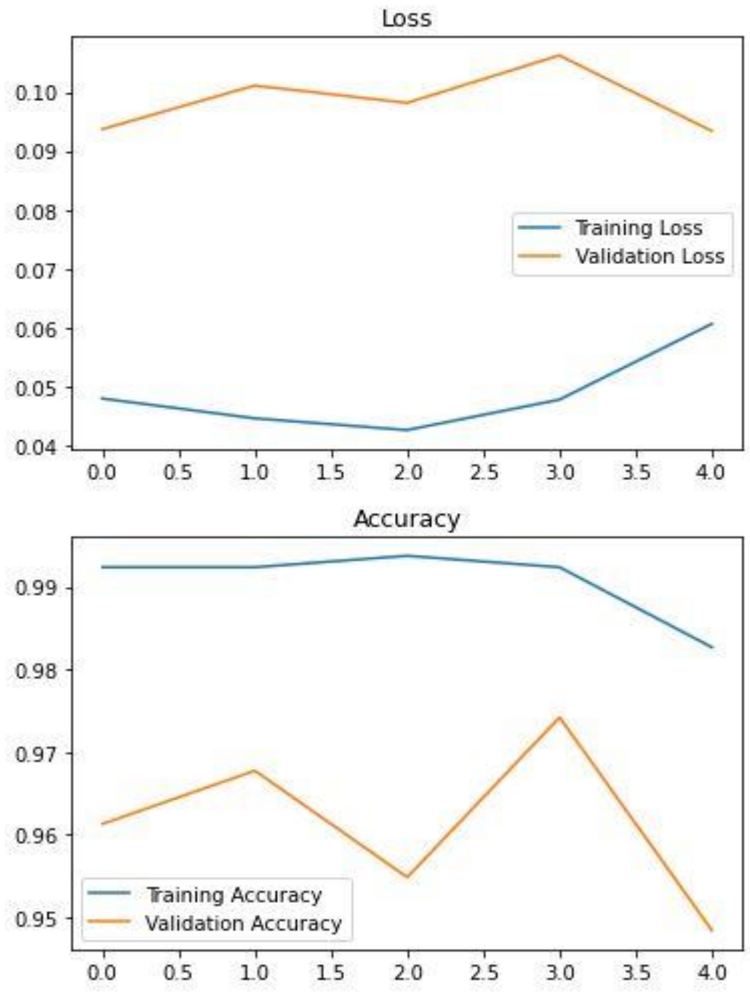


Figure 6.6: Accuracy and Loss graphs for VGG-19.

6.7 Result of InceptionV3

It has the highest train set accuracy among VGG-16, VGG-19 and ResNet50. The other metrics are also quite satisfactory considering only 10 epochs were done due to hardware malfunctions.

Accuracy

Train accuracy = 100 %

Validation accuracy = 97.74 %

Test accuracy = 96.45 %

F1 Score

Validation = 97.79 %

Test = 96.86 %

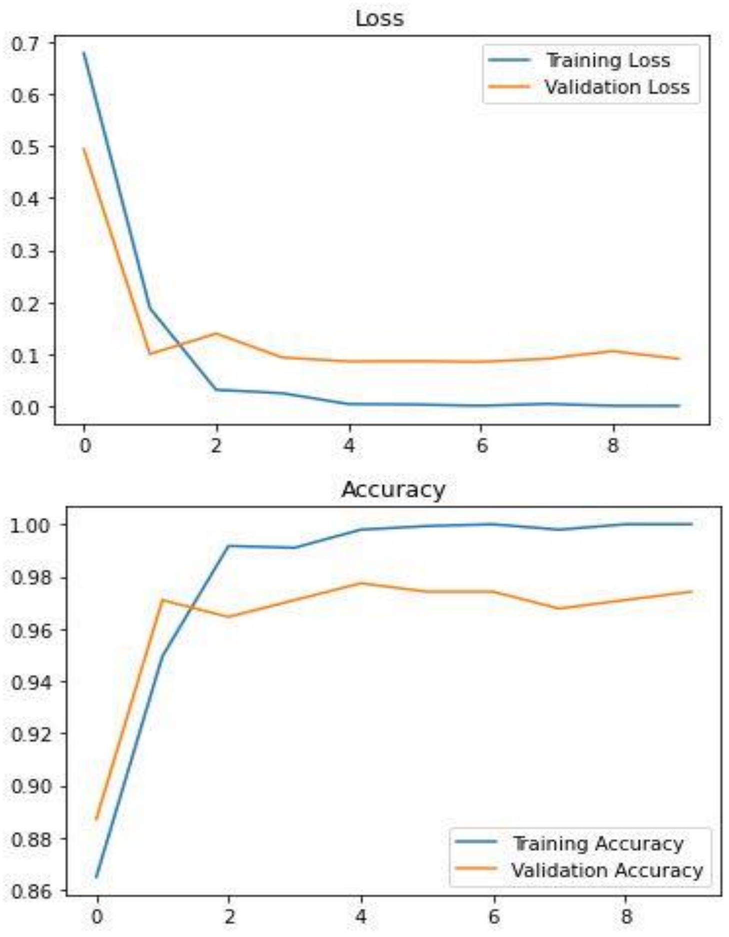


Figure 6.7: Accuracy and Loss graphs for InceptionV3.

6.8 Result of ResNet50

ResNet50 model shows the least amount of over-fitting as the train and validation set accuracy is very close. But it also has the worst result of all CNN architectures.

Accuracy

Train accuracy = 80.25 %

Validation accuracy = 80.32 %

Test accuracy = 82.20 %

F1 Score

Validation = 83.10 %

Test = 85.01 %

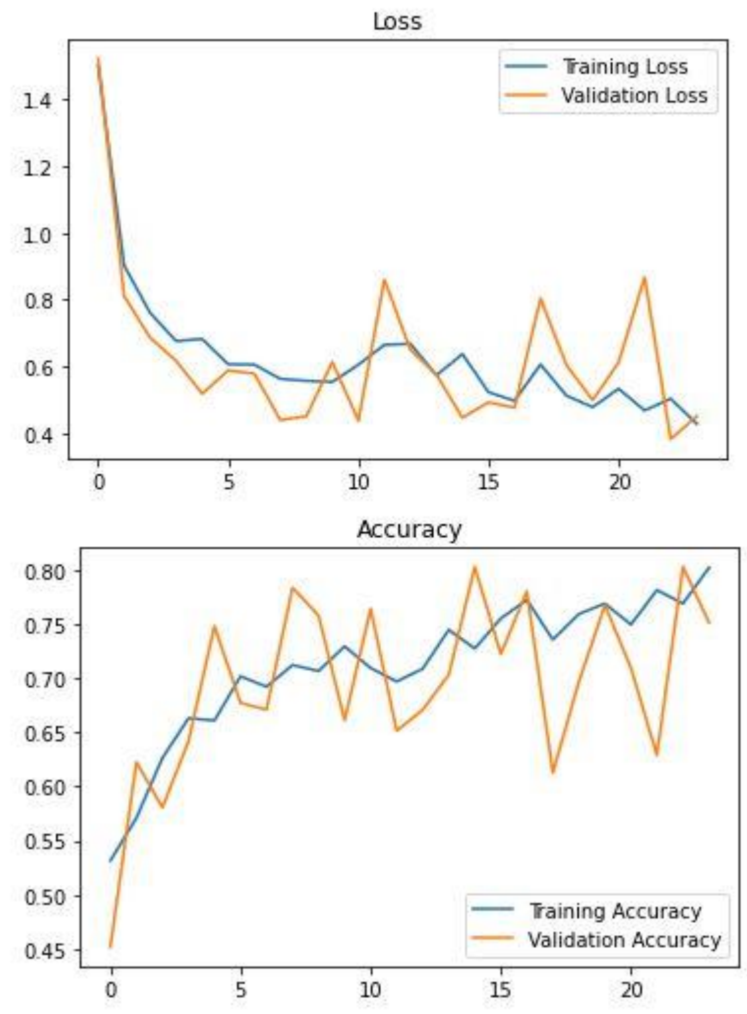


Figure 6.8: Accuracy and Loss graphs for ResNet50.

6.9 Analysis of Results

This experiment covers a huge area. From simple CNN with a single layer of convolution to very deep CNN with 159 layers all are fed the same data and observed. They were evaluated on how they work on unseen test data.

Time and memory consumptions were not taken as an evolution metrics. Although a general insight is the pre-trained models took longer time than the basic architectures. VGG-16 took the most time almost 13.8 minutes for each epoch. On the other hand, InceptionV3 was the fastest among the pre-trained models. It took only 6.6 minutes per epoch.

Every model was trained on 24 epochs in order to get more precise accuracy. But the most memory consuming was InceptionV3 and could not finish all the epochs. The allocated ram was crashing after only 10 epochs and it was stopped. So, in case of InceptionV3 the training was only 10 epochs long and only the accuracies on those epochs were considered.

The batch size was same for every model and the number is 32. It means 32 images from the dataset were processed as a batch before the model is updated to make the learning process faster. Also, another hyper-parameter used was learning rate of the optimizer algorithm ADAM. In order to avoid getting trapped in local minima the learning rate was kept at 0.001.

The deep networks sometimes train on the training set too accurately and does not perform well on unseen test data or validation data. This phenomenon is known as overfitting in deep learning. Here to avoid it dropout is used to numb some of the hidden layers. The dropout rate is 0.5.

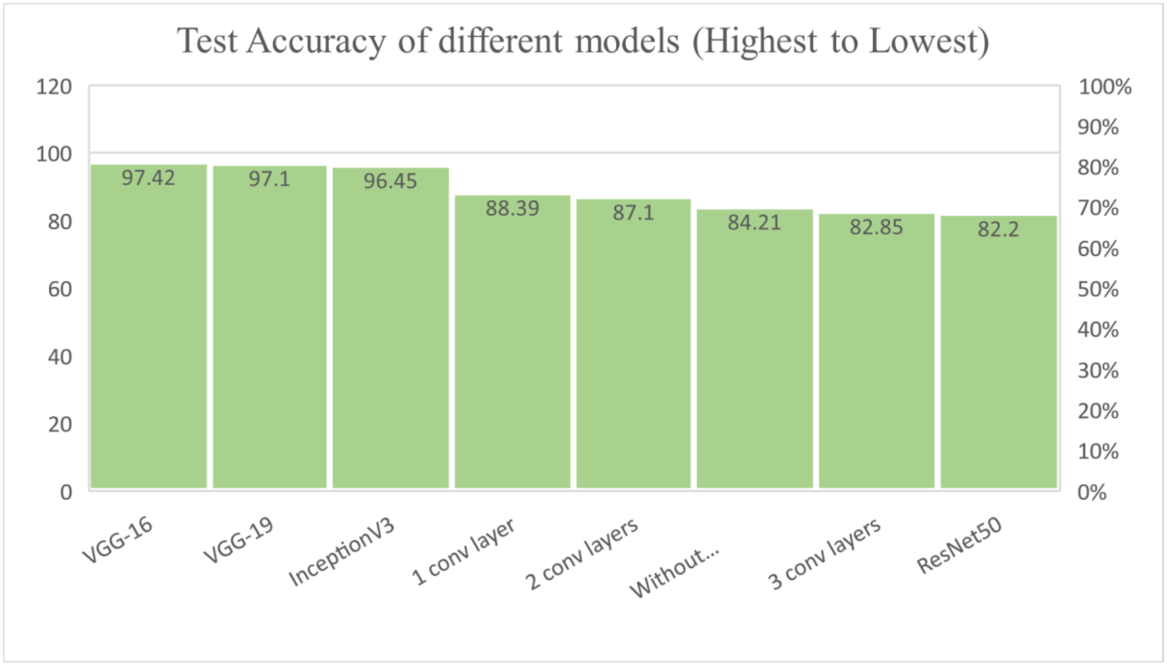
Initially the basic architecture was doing fine but as the layers increased the accuracy decreased. That's why the tuning stopped after adding 3 convolution layers. The accuracies are 88.39%, 87.10% and 82.85% for 1, 2 and 3-layer models respectively.

Rather than them the pre-trained models have better results except ResNet50. The best result was VGG-16 with a test accuracy of 97.42%. But it was very time consuming and computationally expensive. The worst test accuracy was by ResNet50. It only got 82.20% which is lower than 82.85% the result of the single layer architecture and original dataset (without augmentation).

Table 6.1: Test Accuracy of Different CNN Models (Highest to Lowest)

CNN architecture	**Test accuracy (%)**
VGG-16	97.42
VGG-19	97.10
InceptionV3	96.45
1 convolutional layer 2 max-pool layers	88.39
2 convolutional layers 3 max-pool layers	87.10
1 convolutional layer 2 max-pool layers (without augmentation)	84.21
3 convolutional layers 4 max-pool layers	82.85
ResNet50	82.20

A histogram with the test accuracy is drawn below to comprehend the performances in a peek:



Chapter 07

Conclusion and Future Research

Conclusion and Future Research

7.1 Conclusion

The aim of this thesis was to learn how different CNN models both simple architecture and complex architecture behaves on the same dataset. Often small datasets perform much better on basic architectures with a small number of layers than deep neural networks with numerous layers. But in case of this dataset the pre-trained models showed better results.

The importance of data augmentation was also noticed as the un-augmented tiny dataset had a worse performance on the same model. Also, tuning with different number of convolutional layers was done but as the results were dropping so it was stopped after training the model with three convolutional and four max pool layers. As the layers of the model increased, so did the over-fitting. The architecture modeled on the training data so well that it could not perform well on the unseen test data as it picked and learned the noise or random fluctuations in the training data too. The filter size, number of filters these attributes were kept same for all the models.

The best accuracy was achieved by VGG-16 model which is 97.42%. But the time for per epoch was also very high and that is 13.8 minutes. On the other hand, InceptionV3 had obtained 96.45% accuracy by only taking 6.6 minutes per epoch. So, the most convenient model to use among the models trained for this research would be InceptionV3 since it has reached the most accuracy in the shortest time.

The chapters are designed in a manner that it gives a whole perspective on the thesis work as well as the very primary steps of Convolutional Neural Network. The first chapter is dedicated to the introduction of the thesis. So, brain tumor, MRI and its advantages plus disadvantages are described there. Second chapter is about the works of other researchers on the same field. The basic attributes of any convolutional neural network is being introduced in the third chapter. From the layers of CNN to the loss function, activation function, optimizer algorithms, evaluation metrics everything is described on this chapter.

After building the foundation on CNN networks on the third chapter, the fourth is about the pre-trained models. The models used in this thesis is narrated with the help of images. The next two chapters' methodology and experimental results and analysis describe about the work done for this thesis. Finally, the last chapter sums all this up and gives an overall prospect on the thesis. Possible future research directions are also provided in the last chapter of this book.

7.2 Future Research Directions

The best thing about CNN is the endless options that can be taken to get a more accurate result. The models used in this thesis gave a good performance but there is also room for more perfection so future research can be done on the same materials. Some ideas for future work is listed below:

- The basic architecture can be tuned with a bigger dataset to decrease over-fitting.
- The filter size and number can be tweaked to observe how it affects the accuracy curve.
- The dataset can be trained with other pre-trained models such as AlexNet, GoogleNet etc.
- Some models showed a lot of over-fitting and steps can be taken to reduce it.
- Creating a real time dataset and train the proposed model on that dataset.

References

[1] L. M. DeAngelis, “Brain Tumors,” *N. Engl. J. Med.*, vol. 344, no. 2, pp. 114–123, Jan. 2001, doi: 10.1056/NEJM200101113440207.

[2] “Stewart BW, Wild CP, World Cancer Report 2014. 2015. IARC Report - Onco’Zine.” .

[3] “Brain, other CNS and intracranial tumours incidence statistics | Cancer Research UK.” https://www.cancerresearchuk.org/health-professional/cancer-statistics/statistics-by-cancer-type/brain-other-cns-and-intracranial-tumours/incidence

[4] A. Behin, K. Hoang-Xuan, A. F. Carpentier, and J. Y. Delattre, “Primary brain tumours in adults,” *Lancet*, vol. 361, no. 9354. Elsevier, pp. 323–331, Jan. 25, 2003, doi: 10.1016/S0140-6736(03)12328-8.

[5] A. Patel, “Benign vs Malignant Tumors,” *JAMA Oncology*, vol. 6, no. 9. American Medical Association, p. 1488, Sep. 01, 2020, doi: 10.1001/jamaoncol.2020.2592.

[6] “MRI, Magnetic Resonance Imaging | Mayfield Brain & Spine Cincinnati, Ohio.” https://www.nibib.nih.gov/science-education/science-topics/magnetic-resonance-imaging-mri.

[7] “MRI | CancerQuest.” https://www.cancerquest.org/patients/detection-and-diagnosis/magnetic-resonance-imaging-mri.

[8] “Brain Tumor: Statistics | Cancer.Net.” https://www.cancer.net/cancer-types/brain-tumor/statistics.

[9] H. A. Khan, W. Jue, M. Mushtaq, and M. U. Mushtaq, “Brain tumor classification in MRI image using convolutional neural network,” doi: 10.3934/mbe.2020328.

[10] “OpenCV: Canny Edge Detection.” https://docs.opencv.org/master/da/d22/tutorial_py_canny.html

[11] P. Saxena, A. Maheshwari, S. Tayal, and S. Maheshwari, “Predictive modeling of brain tumor: A Deep learning approach,” *Adv. Intell. Syst. Comput.*, vol. 1189, pp. 275–285, Nov. 2019.

[12] Onkar Rajesh Mulay, and Hemprashad Yashwant Patil, “Transfer Learning for Classification of 2D Brain MRI Images and Tumor Segmentation,” doi: 10.35940/ijrte.F8321.038620.

[13] M. Toğaçar, B. Ergen, and Z. Cömert, “BrainMRNet: Brain tumor detection using magnetic resonance images with a novel convolutional neural network model,” *Med. Hypotheses*, vol. 134, p. 109531, Jan. 2020, doi: 10.1016/j.mehy.2019.109531.

[14] R. Chelghoum, A. Ikhlef, A. Hameurlaine, and S. Jacquir, “Transfer learning using convolutional neural network architectures for brain tumor classification from MRI images,” in *IFIP Advances in Information and Communication Technology*, Jun. 2020, vol. 583 IFIP, pp. 189–200, doi: 10.1007/978-3-030-49161-1_17.

[15] M. A. Bakr Siddiaue, S. Sakib, M. M. Rahman Khan, A. K. Tanzeem, M. Chowdhury, and N. Yasmin, “Deep Convolutional Neural Networks Model-based Brain Tumor Detection in Brain MRI Images,” in *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Oct. 2020, pp. 909–914, doi: 10.1109/I-SMAC49090.2020.9243461.

[16] A. Kumar Pandey and K. C. James, “A Review of Different Classification Techniques used in Brain Tumor Detection.”

[17] D. Kim, “BRAIN TUMOR DETECTION: 2 NOVEL APPROACHES,” Aug. 2020, doi: 10.20944/preprints202008.0641.v1.

[18] H. H. Sultan, N. M. Salem, and W. Al-Atabany, “Multi-Classification of Brain Tumor Images Using Deep Neural Network,” *IEEE Access*, vol. 7, pp. 69215–69225, 2019, doi: 10.1109/ACCESS.2019.2919122.

[19] I. Goodfellow, Y. Bengio, and A. Courville, “Deep Learning.”

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998, doi: 10.1109/5.726791.

[21] Yann LeCun, Corinna Cortes and Chris Burges, “MNIST handwritten digit database” http://yann.lecun.com/exdb/mnist

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, Jun. 2017, doi: 10.1145/3065386.

[23] “ImageNet.” http://www.image-net.org

[24] R. Geirhos, D. H. J. Janssen, H. H. Schütt, J. Rauber, M. Bethge, and F. A. Wichmann, “Comparing deep neural networks against humans: object recognition when the signal gets weaker *,” 2018.

[25] T. Hey and A. Trefethen, “The Data Deluge: An e-Science Perspective,” in *Grid Computing*, John Wiley & Sons, Ltd, 2003, pp. 809–824.

[26] R. HECHT-NIELSEN, “Theory of the Backpropagation Neural Network**Based on ‘nonindent’ by Robert Hecht-Nielsen, which appeared in Proceedings of the International Joint Conference on Neural Networks 1, 593–611, June 1989. © 1989 IEEE.,” in *Neural Networks for Perception*, Elsevier, 1992, pp. 65–93.

[27] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533–536, 1986, doi: 10.1038/323533a0.

[28] Phung and Rhee, “A High-Accuracy Model Average Ensemble of Convolutional Neural Networks for Classification of Cloud Image Patches on Small Datasets,” *Appl. Sci.*, vol. 9, no. 21, p. 4500, Oct. 2019, doi: 10.3390/app9214500.

[29] Rachel Lea Ballantyne Draelos, “The History of Convolutional Neural Networks ” https://towardsdatascience.com/a-short-history-of-convolutional-neural-networks-7032e241c483

[30] “What’s the Difference Between a CNN and an RNN? - Edge AI and Vision Alliance.” https://www.edge-ai-vision.com/2018/09/whats-the-difference-between-a-cnn-and-an-rnn.

[31] Kousai Smeda, “Understand the architecture of CNN” https://towardsdatascience.com/understand-the-architecture-of-cnn-90a25e244c7.

[32] “Max Pooling Definition | DeepAI.” https://deepai.org/machine-learning-glossary-and-terms/max-pooling.

[33] “Illustration of Max Pooling and Average Pooling | Download Scientific Diagram.” https://www.researchgate.net/figure/Illustration-of-Max-Pooling-and-Average-Pooling-Figure-2-above-shows-an-example-of-max_fig2_333593451

[34] V. Christlein, L. Spranger, M. Seuret, A. Nicolaou, P. Kral, and A. Maier, “Deep generalized max pooling,” in *Proceedings of the International Conference on Document Analysis and Recognition, ICDAR*, Sep. 2019, pp. 1090–1096, doi: 10.1109/ICDAR.2019.00177.

[35] “Convolutional Neural Network Architecture: Forging Pathways to the Future - MissingLink.ai.” https://missinglink.ai/guides/convolutional-neural-networks/convolutional-neural-network-architecture-forging-pathways-future.

[36] “Fully Connected Deep Networks .” https://www.oreilly.com/library/view/tensorflow-for-deep/9781491980446/ch04

[37] Shiva Verma, “Understanding different Loss Functions for Neural Networks” https://towardsdatascience.com/understanding-different-loss-functions-for-neural-networks-dd1ed0274718.

[38] Satyam Kumar, “Overview of various Optimizers in Neural Networks” https://towardsdatascience.com/overview-of-various-optimizers-in-neural-networks-17c1be2df6d5.

[39] Sanket Doshi, “Various Optimization Algorithms For Training Neural Network” https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6.

[40] “Types of Activation Functions in Neural Networks” https://missinglink.ai/guides/neural-network-concepts/7-types-neural-network-activation-functions-right.

[41] SAGAR SHARMA, “Activation Functions in Neural Networks” https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6.

[42] “Evaluation Metrics Definition | DeepAI.” https://deepai.org/machine-learning-glossary-and-terms/evaluation-metrics.

[43] M. Hossin and Sulaiman, “A REVIEW ON EVALUATION METRICS FOR DATA CLASSIFICATION EVALUATIONS,” *Int. J. Data Min. Knowl. Manag. Process*, vol. 5, no. 2, 2015, doi: 10.5121/ijdkp.2015.5201.

[44] “(PDF) The truth of the F-measure.” https://www.researchgate.net/publication/268185911_The_truth_of_the_F-measure

[45] “A Gentle Introduction to Transfer Learning for Deep Learning.” https://machinelearningmastery.com/transfer-learning-for-deep-learning

[46] “Handbook Of Research On Machine Learning Applications and Trends | Guide books.” https://dl.acm.org/doi/book/10.5555/1803899

[47] “Keras Applications.” https://keras.io/api/applications/.

[48] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” Sep. 2015, Available: http://www.robots.ox.ac.uk/.

[49] “VGG-16 neural network architecture. | Download Scientific Diagram.” https://www.researchgate.net/figure/VGG-16-neural-network architecture_fig1_327070011

[50] “VGG16 - Convolutional Network for Classification and Detection.” https://neurohive.io/en/popular-networks/vgg16

[51] “Understanding the VGG19 Architecture.” https://iq.opengenus.org/vgg19-architecture/

[52] “Illustration of the network architecture of VGG-19 model: conv means... | Download Scientific Diagram.” https://www.researchgate.net/figure/llustration-of-the-network-architecture-of-VGG-19-model-conv-means-convolution-FC-means_fig2_325137356

[53] C. Szegedy, V. Vanhoucke, S. Ioffe, and J. Shlens, “Rethinking the Inception Architecture for Computer Vision,” 2016.

[54] “Common architectures in convolutional neural networks.” https://www.jeremyjordan.me/convnet-architectures/#inception

[55] “InceptionV3.” https://keras.io/api/applications/inceptionv3/

[56] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” 2016.[Online]. Available: http://image-net.org/challenges/LSVRC/2015/.

[57] Sabyasachi Sahoo, “Residual blocks — Building blocks of ResNet” https://towardsdatascience.com/residual-blocks-building-blocks-of-resnet-fd90ca15d6ec

[58] “The architecture of ResNet50 and deep learning model flowchart. a, b... | Download Scientific Diagram.” https://www.researchgate.net/figure/The-architecture-of-ResNet50-and-deep-learning-model-flowchart-a-b-Architecture-of_fig1_334767096

[59] “Brain MRI Images for Brain Tumor Detection | Kaggle.” https://www.kaggle.com/navoneel/brain-mri-images-for-brain-tumor-detection

[60] “Finding extreme points in contours with OpenCV - PyImageSearch.” https://www.pyimagesearch.com/2016/04/11/finding-extreme-points-in-contours-with-opencv/

[61] M. A. Tanner and W. H. Wong, “The calculation of posterior distributions by data augmentation,” *J. Am. Stat. Assoc.*, vol. 82, no. 398, pp. 528–540, 1987, doi: 10.1080/01621459.1987.10478458.

[62] S. Mannor, B. Peleg, and R. Rubinstein, “The cross entropy method for classification,” in *ICML 2005 - Proceedings of the 22nd International Conference on Machine Learning*, 2005, pp. 561–568, doi: 10.1145/1102351.1102422.

[63] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” Dec. 2015. Available: https://arxiv.org/abs/1412.6980v9.

[64] “Robbins, Herbert, and Sutton Monro. ‘A Stochastic Approximation Method.’ The Annals of Mathematical Statistics, vol. 22, no. 3, 1951, pp. 400–407. JSTOR, www.jstor.org/stable/2236626” https://www.jstor.org/stable/2236626?seq=1

[65] G. Hinton, N. Srivastava, and K. Swersky, “Neural Networks for Machine Learning Lecture 6a Overview of mini---batch gradient descent.”

[66] V. Nair and G. E. Hinton, “Rectified Linear Units Improve Restricted Boltzmann Machines,” Jan. 2010.