FRANKFURT UNIVERSITY OF APPLIED SCIENCES

3D BRAIN TUMOR IMAGE SEGMENTATION USING DENSE U-NET ARCHITECTURE BASED ON DEEP CONVOLUTIONAL NEURAL NETWORKS

by

Kalaichelvi Rajendran

Under the guidance of

Prof. Dr. Peter Thoma

Prof. Dr. Egbert Falkenberg

A thesis submitted in partial fulfilment for the
degree of Master of Science
in the
High Integrity Systems
FB 2: Informatik Ingenieurwissenschaften

February 2020

Declaration of Authorship

I, Kalaichelvi Rajendran, declare that this thesis, titled '3D Brain Tumor Image Segmentation Using Dense U-Net Architecture Based On Deep Convolutional Neural Networks', and the work presented within it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this university.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work. I have acknowledged all main sources of help.

Signed: _____

Date: _____

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Abstract

High Integrity Systems

Master of Science

by Kalaichelvi Rajendran

Brain tumors result from abnormal cell growth within or around the brain. Diagnosis of the tumor region in the brain is a challenging task for medical professionals. MRI (Magnetic Resonance Imaging) images are helpful in brain tumor analysis, where most classifications are performed by radiologists. Manual identification of tumors with the help of MR images is a tedious process that requires experienced radiologists or specialists. This method is also impractical, as delays can arise from classification of huge volumes of data. Accuracy is very important in medical image classification. MR images may contain noise that can lead to incorrect results. This paper aims to propose a model that can improve accuracy in detecting and segmenting brain tumors using U-Net-based deep convolutional neural networks.

Hyperthermia (HT) is a method of treatment that increases the temperature of the tumor tissue to 39–44 C, significantly enhancing the effectiveness of radiotherapy and chemotherapy. This process essentially increases the temperature of the tumor tissue to kill the cancerous cells. Personalised hyperthermia treatment planning has become a effective tool to improve treatment quality, aided by enhanced growth in computational techniques and computing power. This treatment planning requires patient-specific data, such as accurate data detailing the tumor region's location. Utilizing this data, tumor cells can be destroyed by increasing their temperature using thermal redistribution of energy.

Accurate detection of the tumor region is a challenging, but necessary task that is achieved through image segmentation. Without data detailing the tumor region, Hyperthermia (HT) cannot be performed. 3D MRI brain tumor image segmentation is the method of dividing a digitalized 3D image into multiple smaller portions, or segments. The key idea is to find out whether a tumor is present in the MRI brain image scan and where exactly it is located. Digital images of abnormal tissue regions are segmented so that they can be accurately diagnosed. The method for medical image segmentation proposed in this paper uses deep convolutional neural networks based on the U-Net approach for 3D image segmentation.

Acknowledgements

My sincere thanks to my academic supervisors, Prof. Dr. Peter Thoma and Prof. Dr. Egbert Falkenberg for their continuous support. Their patience, motivation, and vast knowledge and experience was of immeasurable value and greatly appreciated throughout the entirety of this endeavor.

I would like to express my deepest gratitude towards my husband, Raja Jasper, who undoubtedly has been my greatest supporter in my drive to complete this thesis and, in the end, my Masters degree. I am also grateful to have a close friend, Andrew Lacey, who helped me proofread all of the work I put into my thesis.

Lastly, I would like to thank my family, who guided and supported me in every way they could while I pursued a Masters in High Integrity Systems at Frankfurt University of Applied Sciences.

Most Sincerely,
Kalaichelvi Rajendran.

Contents

List of Figures

List of Tables

Abbreviations

MRI	**M**agnetic **R**esonance **I**maging
AI	**A**rtificial **I**ntelligence
ML	**M**achine **L**earning
DL	**D**eep **L**earning
CNN	**C**onvolutional **N**eural **N**etwork
CNS	**C**entral **N**ervous **S**ystem
CT	**C**omputed **T**omography
CE	**C**ontrast **E**nhanced
CAD	**C**omputer **A**ssisted **D**iagnostic
WM	**W**hite **M**atter
GM	**G**ray **M**atter
CSF	**C**erebro**s**pinal **F**luid
GD	**G**a**d**olinium
TE	**T**ime to **E**cho
TR	**R**epetition **T**ime
FLAIR	**F**luid **A**ttenuated **I**nversion **R**ecovery
DWI	**D**iffusion **W**eighted **I**maging
HT	**H**yper**t**hermia
WHO	**W**orld **H**ealth **O**rganization
ANN	**A**rtificial **N**eural **N**etwork
DNN	**D**eep **N**eural **N**etwork
RNN	**R**ecurrent **N**eural **N**etwork
RBM	**R**estricted **B**oltzmann **M**achine
DBN	**D**eep **B**elief **N**etwork
MLP	**M**ulti**l**ayer **P**erceptron

GPU	**G**raphical **P**rocessing **U**nit
RGB	**R**ed **G**reen **B**lue
ReLU	**Re**ctified **L**inear **U**nit
AF	**A**ctivation **F**unction
ET	**E**nhancing **T**umor
TC	**T**umor **C**ore
WH	**W**hole **T**umor
ED	**Ed**ema
BraTS	**Bra**in **T**umor **S**egmentation
NCR	**N**e**cr**otic
NET	**N**on-**E**nhancing **T**umor

Chapter 1

Problem Statement

One of the deadliest diseases in the world is the brain tumor (also called brain cancer), as evidenced by its low survival rates. The National Brain Tumor Society (2019) has reported that the chances of surviving a primary malignant brain tumor over 5 years is 5.6% [35]. Research scientists around the world have pursued and continue to pursue finding a cure for brain cancer. A crucial factor in the treatment of cancer is early detection.

Analyzing a tumor within the human brain is a difficult process because of its complicated structure. Magnetic Resonance Imaging (MRI) is one of several screening methods used to detect the presence of tumors, inflammation, and injury in different parts of the body. MRI images are used by radiologists to manually detect tumors and analyze the severity to determine which treatment procedure should be defined for a patient. One such treatment is called Hyperthermia (HT). It is a method of treatment that increases the temperature of the tumor tissue to 39–44 C, essentially killing the tumor tissue. This is a powerful tool for patient-specific tumor treatment that cannot be performed without accurate tumor segmentation.

In this personalised treatment planning, accurate detection of the tumor region is a challenging, but necessary task. Combined with growing technologies, such as Machine Learning (ML) and Deep Learning (DL), MRI imaging can be more effectively utilized to detect tumors efficiently and accurately. Various deep learning architectures are utilized in the areas of medical image segmentation. The intention of this thesis is to research and propose a deep learning model that further increases both efficiency and accuracy

in the detection of brain tumors. Accurate detection of tumors plays a vital role in determining and providing optimal treatments for patients.

1.1 Objectives

This thesis work aims to research and identify an appropriate deep learning model capable of efficiently and accurately detecting brain tumors via image segmentation. Using this model, a novel method for the detection of brain tumors will be proposed.

To find an appropriate deep learning model, the following questions must be answered:

- How are tumor regions differentiated from normal brain tissues?
- What are the different MRI modalities and how can they be used with deep learning models?
- What deep learning models are currently available and how are they different from one another?
- What research already exists in the field of brain tumor detection via medical imaging?

1.2 Contributions

In this paper, a 3D brain tumor image segmentation model is proposed by combining two deep learning models; namely DenseNet and U-Net. This combination yields several advantages in medical image segmentation. A block diagram is presented to understand the different phases involved in the proposed brain tumor segmentation model. MRI image preprocessing and patch extraction are among the primary methods of cleansing corrupted images for better segmentation results. 64 x 64 x 64 patches are extracted from the MRI image of total size 240 x 240 x 155 to concentrate on the significant tumor area and avoid the non-tumor region. This will allow us to reduce training time and efficiently segment the tumor region. The proposed Dense U-Net model utilizes the advantages of DenseNet and U-Net to provide an efficient image segmentation algorithm.

1.3 Structure of the Document

This thesis report is structured in the following manner.

Chapter 2: This chapter gives information regarding the Human Brain, Different types of tumors, Introduction to MRI, and different modalities related to this thesis.

Chapter 3: This chapter is an introduction to Artificial Intelligence (AI), Machine Learning (ML), types of Machine Learning (ML), and Deep Learning (DL).

Chapter 4: Deep Learning Models - CNN, U-Net, ResNet, DenseNet in image segmentation are explained in this chapter.

Chapter 5: In this chapter, the theoretical Background related to image segmentation in the field of medical image processing is explained.

Chapter 6: Proposed architecture and pseudocode of the thesis are discussed in this chapter.

Chapter 7: This chapter provides a summary of the thesis and future enhancements.

Chapter 2

Brain Tumors and MRI

2.1 Introduction

The brain is the most complex organ in the human body. It is made up of more than 100 billion nerves (neurons) that manage the activities of our nervous system. Together with the spinal cord, it is an important part of the Central Nervous System (CNS), controlling most functions of the body and the mind [36]. The brain controls our personality, vital body functions, senses, and movement. It is capable of much more than a machine and is the foundation of human intelligence [37].

The brain's importance makes it all the more critical that brain tumors are detected and treated quickly. Tumors develop from unchecked cell growth within or around the brain, disrupting normal brain functions. Given that the brain is housed within the skull, there exists only so much room for this cell growth to expand. Compression and displacement of brain tissues resulting from cancerous tumor expansion can cause serious, and in many cases fatal damage.

In order to help combat this threat, scanning and imaging techniques have been developed over the years that offer a means to detect brain tumors, allowing medical professionals to plan and provide targeted treatment. In this chapter, the anatomy of the brain will be explained in detail, which provides an overview of the different types of tumors, and discuss various methods of scanning and imaging, each with their advantages and disadvantages.

2.2 The Human Brain

The human brain is composed of three main parts: the cerebrum, cerebellum, and brainstem (as shown in Figure 2.1).

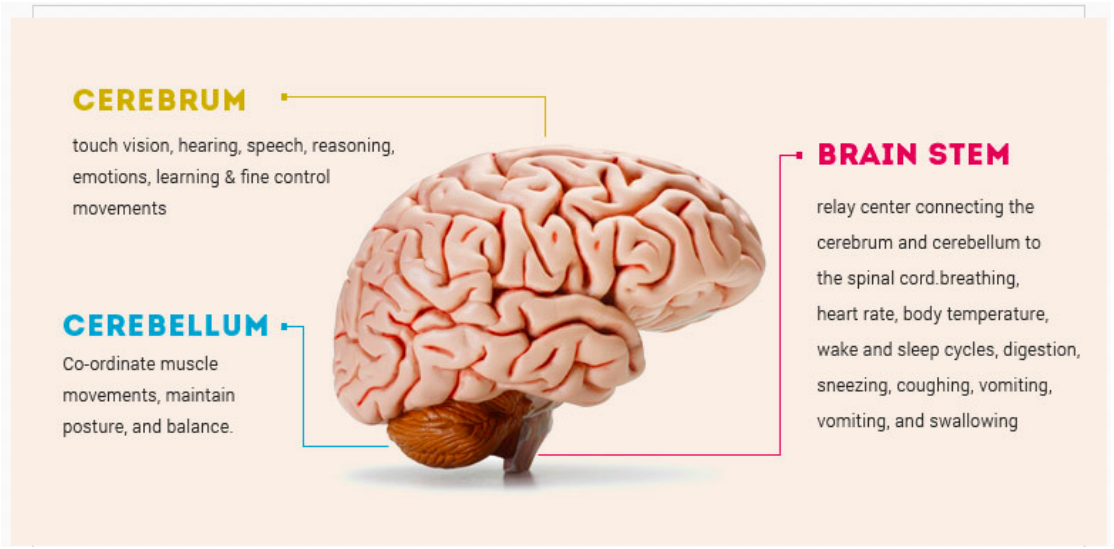

FIGURE 2.1: The Human Brain - The Human brain is composed of three main parts the cerebrum, cerebellum and brain stem [1]

The cerebrum is divided into two halves: the left hemisphere and the right hemisphere. The hemispheres are joined by the corpus callosum, which is a bundle of fibers that transmits messages from one side to the other. Each hemisphere controls the side of the body opposite from it. As such, if the right side of the brain is affected by a stroke, then the left leg or left arm may become paralyzed or weak (as shown in Figure 2.2) [2].

The cerebral hemispheres are further divided into lobes. Each hemisphere (the left and the right brain) has four lobes (as shown in Figure 2.3) [2].

- The Parietal lobe: Language interpretation, sense of touch, memory, pain, etc.
- The Frontal lobe: Personality, speaking, writing, intelligence, behavior, etc.
- The Temporal lobe: Memory, hearing, sequencing and organization, understanding language.
- The Occipital lobe: Vision interpretation (color, light).

The structure of the brain is complicated and can be evaluated only by skillful, expert physicians and radiologists. Brain tumor extraction is a challenging task. Detection of

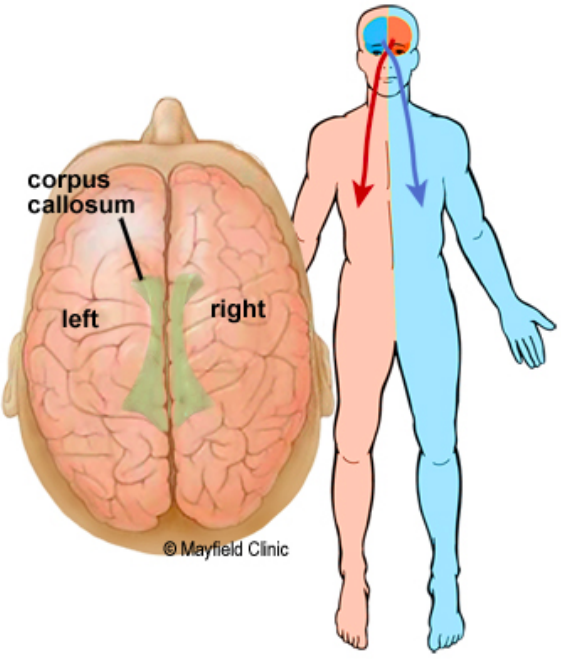

FIGURE 2.2: The Cerebrum - The cerebrum is divided into left and the right hemispheres, each connected by the nerve fibres called the corpus callosum [2].

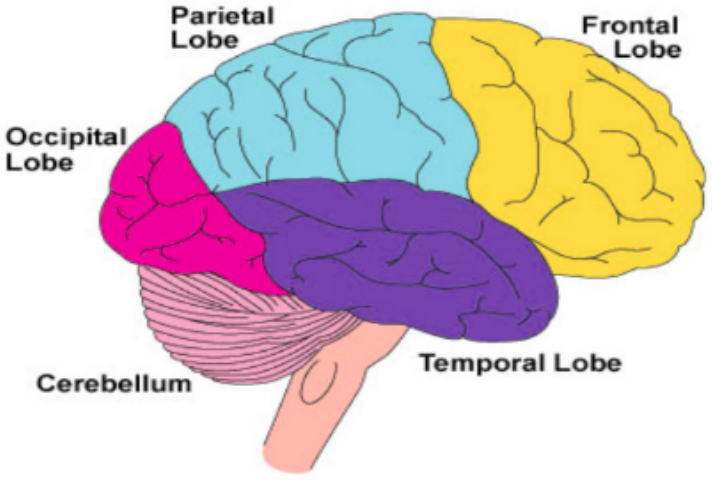

FIGURE 2.3: Lobes of the brain - The cerebrum is divided into four lobes: the parietal lobe, the frontal lobe, the occipital lobe, and the temporal lobe [3]

brain tumors and analysis of images is time-consuming. In fact, it is the most time-consuming task in medical image processing. Image segmentation is a crucial job in medical image processing that will be further discussed in this paper. There are several clinical diagnostic tools (X-Ray, Magnetic Resonance Imaging (MRI), Computed Tomography CT, Endoscopy, etc.) which could be used to capture medical images in order to diagnose disease, injury, and tumors [38].

2.3 Brain Tumor

Normally in adults, a new cell is formed whenever it is required to replace old or damaged cells. A tumor is developed when a normal or abnormal cell multiplies when it is not required [4]. An abnormal cell or tissue growth that develops within or around the brain is called a brain tumor and can disrupt normal brain functions. It is hazardous to a healthy brain because it can damage normal brain tissues by compressing and displacing them. The brain has a fixed amount of space inside the skull. As such, a growing brain tumor can cause serious brain damage [39]. There are two main categories of brain tumor:

- **Primary Brain Tumor:** The tumor starts in the brain and tends to stay there [40].
- **Secondary or Metastatic Brain Tumor:** The tumor starts in another part of the body and spreads to the brain [4].

2.3.1 Primary Brain Tumors

Primary brain tumors originate and stay in the brain itself or in the tissues around it, such as the brain-covering membranes (see: Figure 2.4). When normal cells acquire errors or alterations (also called mutations) in their DNA, a primary tumor begins. These mutations cause the cells to multiply at rapid rates and continue to live while healthy cells die. This results in the formation of abnormal cells, causing a tumor. Primary brain tumors are least frequent in adults when compared to secondary brain tumors [40].

There are more than 120 different kinds of brain tumors, each of which falls under one of two groups: benign tumors (non-cancerous) and malignant tumors (cancerous) [40].

- **Benign brain tumors** usually have distinct borders consisting of very slow-growing cells (non-cancerous) that rarely spread into other tissues. These cells almost appear as normal cells when viewed under a microscope. Benign tumors can be effectively removed through surgical treatment. However, they may become life-threatening when situated in a crucial area of the brain that affects the brain's

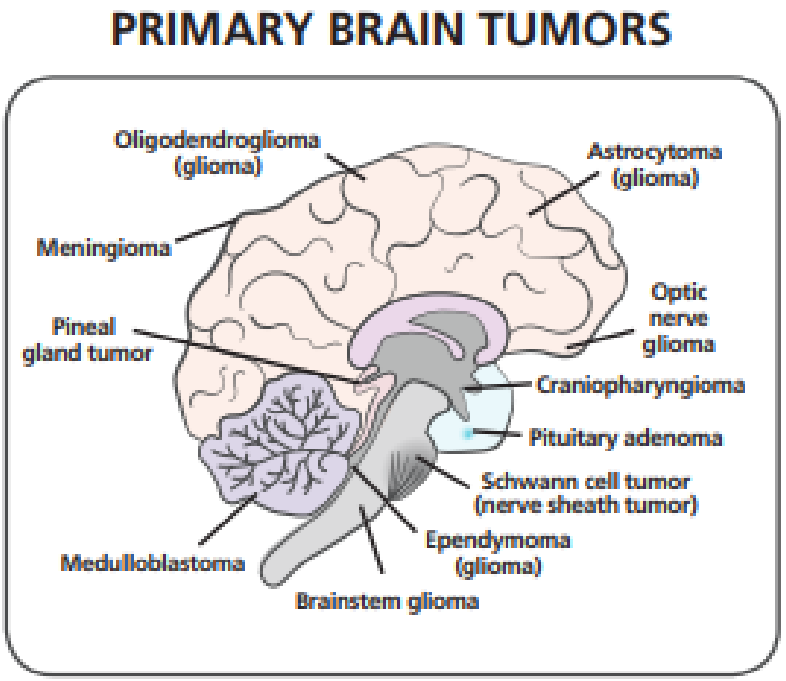

FIGURE 2.4: Primary Brain Tumors - The tumor begins in the brain and tends to stay there [4].

ability to work normally, although the tumor cells have not been categorized as malignant [4].

- **Malignant brain tumors** lack clear borders, are invasive (spreads very quickly), and contain cancer cells that typically grow at a rapid pace. This sort of brain tumor is life-threatening because it invades into the surrounding brain tissue. A malignant brain tumor is a type of primary brain tumor and therefore does not spread to different parts of the body. However, they can spread throughout the brain or to the spine [5]. Malignant brain tumors have the tendency to send "roots" to the nearby normal tissues. These "roots" are cells that shed while traveling to different parts of the spine and brain through the CSF (Cerebrospinal Fluid). Such tumors are sometimes referred to as brain cancer [4]. Malignant tumors can be treated with chemotherapy, radiation, and surgery, though they may still recur after treatment [5].

Primary brain tumor diagnoses in 2019 show an estimated "700,000 people in the United States are living with a primary brain tumor, and approximately 86,000 more will be diagnosed in 2019" [35]. Nearly one third of diagnosed primary brain tumors are malignant

(as show in Figure 2.5)

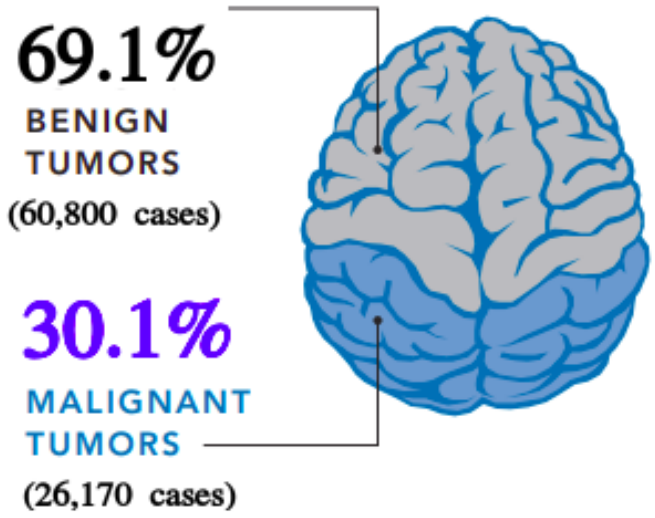

FIGURE 2.5: Primary Brain Tumor diagnosis in the United States (2019) - In the United States, an estimated 700,000 people are living with a primary brain tumor, and over 86,000 more will be diagnosed in 2019 [5]

2.3.2 Metastatic or Secondary Brain Tumors

Metastatic tumors begin in another part of the body, then spread to the brain. In adults, this kind of tumor is more common when compared to primary brain tumors. They usually occur in people who have a hereditary history of cancer. Tumors are named after the part of the body in which they begin. For example, Breast cancer, Colon cancer, Kidney cancer, Lung cancer, etc. [5] (such as in Figure 2.6).

2.4 Tumor Grading

Tumors are identified (diagnosed) and classified based on the World Health Organization (WHO) classification system [4].

Doctors and medical professionals assign "grades" to the tumors, making it easier for healthcare team members to communicate clearly about the tumor, define the treatment options (to plan), and predict the outcomes. Tumors are classified into four grades (I, II, III, IV) based on their cancerous properties. Grade I tumors develop at a slower pace and are able to be treated easily, whereas Grade IV tumors are the most malignant and are extremely hard to treat [4].

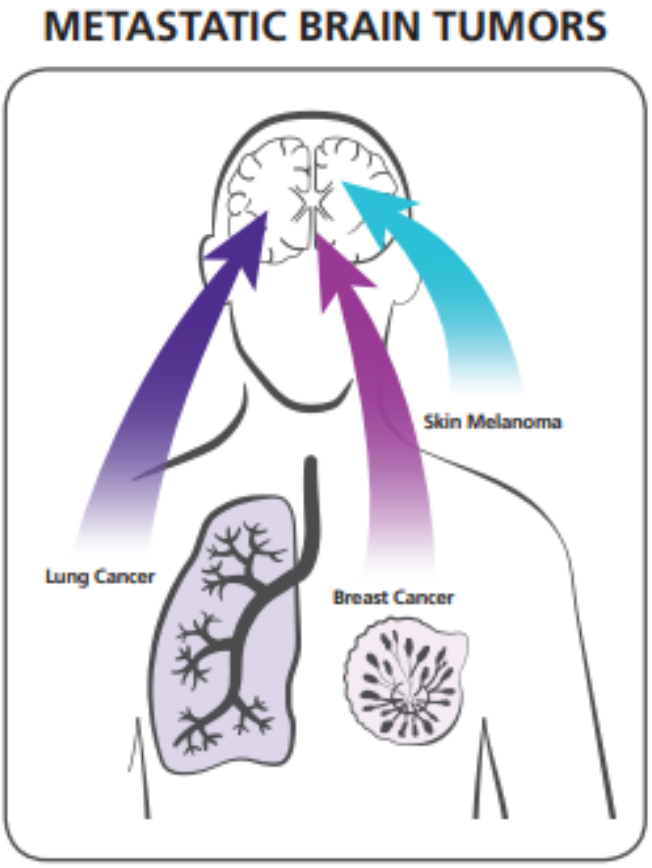

FIGURE 2.6: Metastatic Brain Tumors - The tumor starts in another part of the body and spreads to the brain [4].

2.4.1 World Health Organization (WHO) Grading System

1. **Grade I**
   (a) The least cancerous tumors and are usually associated with long term survival.
   (b) Tumor cells grow slowly and appear normal when viewed under a microscope.
   (c) Surgery is an effective treatment for Grade I tumors.
2. **Grade II**
   (a) Relatively slow-growing cells that appear slightly abnormal under the microscope.
   (b) Some tumors are invasive. They can spread into nearby normal tissues and become a higher-grade tumor.
3. **Grade III**

(a) Abnormal cells are reproduced actively into nearby normal tissues.

(b) These tumors tend to recur and change into a Grade IV tumor.

4. **Grade IV**

(a) The most malignant brain tumors. The tumor cells multiply with rapid speed and can have a "bizarre cellular appearance" [4] through a microscope [4].

(b) Tumor cells grow at a rapid speed around the ordinary brain tissues. This rapid growth is maintained by forming new blood vessels [4].

(c) Glioblastoma is a Grade IV tumor.

2.5 Scans and Imaging Techniques

The latest advancements in medical imaging from methods such as CT and MRI can provide comprehensive data on diseases and can recognize many pathological conditions for precise diagnosis. Specialists utilize brain MRI scans to diagnose and treat patients with brain tumors. Still, the specialist must manually analyze the MRI scan. This is a tedious process and the accuracy of the result depends on the specialist's experience. Findings may vary from one physician to another. These challenges can be overcome by automating or robotizing the investigative methods used to diagnose brain tumors via digital MRI images. Biomedical image processing methods are applied to MRI scans for this purpose. Consequently, the image segmentation and further characterization of brain tumors from MRI scans remains a wide-ranging area of research in the medical science field. A Computer-Assisted Diagnostic (CAD) system for automatic brain tumor detection through MRI was developed in the initial stages. This is a major objective for doctors because early detection of diseases is the main factor in effective and successful treatment [38].

Segmentation is an essential and mandatory task in MRI image processing. It is a method in which the scanned image is divided into several parts based on different tumor tissues, such as solid tumors, edema, necrosis, and normal brain tissues. These parts are White Matter (WM), Gray Matter (GM), and Cerebrospinal Fluid (CSF) (as shown in Figure 2.7). In addition, the characteristics of tumor cells, such as their complicated form, heterogeneous intensity distribution, tumor position variability, and

tumor artifacts also have an important effect on diagnosis. The heterogeneity of tumors explains the observation that different tumor cells have distinct phenotypic and morphological profiles, including metabolism, gene expression, proliferation, cellular morphology, metastatic potential, and motility. There are significant challenges in designing effective treatment strategies for the heterogeneity of cancer cells. MRI images such as T1-weighted and T2-weighted are used for various segmentation methods. T1-weighted images are widely tested for different segmentation methods due to their more favorable contrast levels [38].

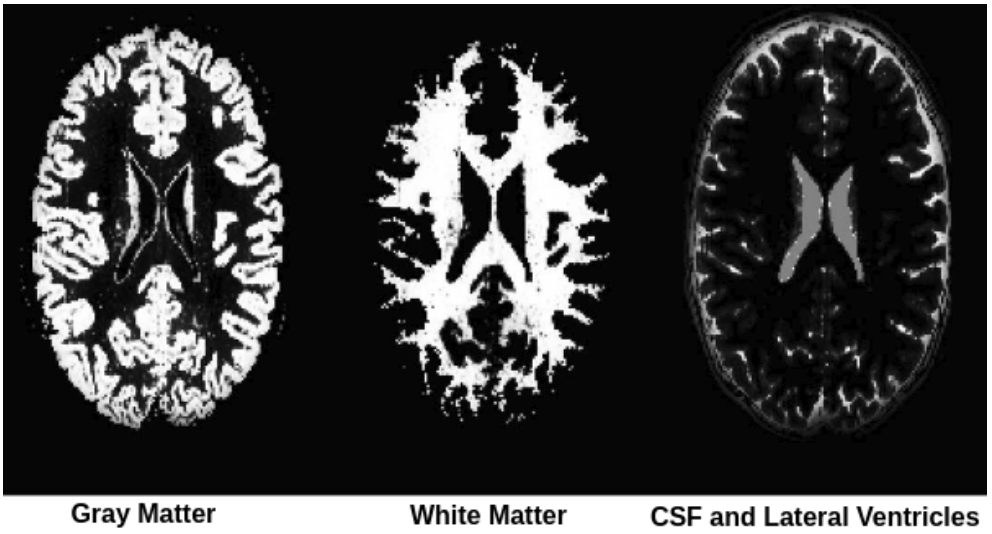

FIGURE 2.7: Appearance of Grey Matter (GM), White Matter (WM), and Cerebrospinal Fluid (CFS) in MRI images [6].

2.6 MRI Basics

The assessment of tumors with medical imaging procedures is now one of the main aims of radiology departments. **Computed tomography (CT)**, **Magnetic Resonance Imaging (MRI)**, and various advanced MRI techniques play a crucial role in brain tumor assessment. CT imaging is often used in initial tumor assessments, as it is minimally invasive, low-cost, and extensively accessible in medical practice. CT is an effective method of screening that helps to illustrate supratentorial abnormalities. However, MRI provides additional sequences that are mandatory for improved distinction of anatomy with regards to judgment of suitable surgical procedures. The reports provided by MRI in the assessment of brain tumors are essential for accurate diagnosis, remedial intervention, and prediction [41].

2.6.1 What is Magnetic Resonance Imaging (MRI)?

Magnetic Resonance Imaging (MRI) is a radiological scanning procedure that is more often used in neurology and neurosurgery tests. MRI generates images of the brain, spinal cord, and other organ structures in exquisite detail by generating signals from the body using a strong magnetic field and radio-frequency waves. The MRI scanner is typically shaped like a tunnel with a giant circular magnet surrounding it. The patient lies down on a movable table that slides into the tunnel. A strong magnetic field is created that aligns hydrogen atom protons, which are then exposed to radio-frequency waves. The receiver portion of the MRI scanner detects the faint signal that spins off of various protons existing in the body. A radio antenna senses these signals and a computer processes them to create detailed images of the inside of the patient's body. MRI scanners produce images in highly detailed resolution and can detect minute structural changes inside the body. Contrast agents, such as gadolinium, are used in some procedures to improve image accuracy [42].

2.6.2 Physics of MRI

Magnetic Resonance Imaging (MRI) is based upon the magnetic properties of atomic nuclei. Uniform and strong magnetic fields are utilized to line up the randomly appearing protons within the water nuclei of the tissues. Next, an external Radio Frequency (RF) energy is introduced to disrupt the magnetic field (or the proton alignment). The nuclei are then returned to their resting positions using numerous relaxation processes. After the initial RF is given, the discharged signals are measured over a period of time. Frequency information present in the signal from every location in the capture area is converted into equivalent intensity levels using Fourier transformation. This is then viewed as gray-shaded pixels in a matrix arrangement. By applying different levels of RF energy and collecting the emitted signals, several kinds of scan images are created. The time taken between the transmission of RF energy and the receiving of the echo signal is called **Time to Echo (TE)**. The time taken between consecutive energy sequences applied to the same image slice is called **Repetition Time (TR)** [7].

There are two different types of relaxation time through which the tissues are characterized: Longitudinal (T1) and Transverse (T2). **Longitudinal relaxation time, or T1**,

determines the speed at which the excited protons return back to equilibrium. That is to say, T1 is the measurement of the time taken to realign the spinning protons with the external magnetic field. **Transverse relaxation time, or T2,** is the constant of time that determines the speed at which the excited protons attain the equilibrium state. That is to say, T2 is the measurement of time taken for the proton spin to lose phase coherency amongst the nuclei spinning perpendicular to the main field [7].

2.6.3 MRI Imaging Sequences

The most frequently used MRI sequences are T1-weighted and T2-weighted scans (as shown in Figure 2.8). The images that are produced by utilizing short TR and TE times are called T1-weighted images (as shown in Table 2.1). The T1 properties of the tissue primarily determine the contrast and brightness of T1-weighted images. Conversely, the images that are produced utilizing long TR and TE times are called T2-weighted images (as shown in Table 2.1). The T2 properties of the tissue primarily determine the contrast and brightness of T2-weighted images [7].

	T1-Weighted (shortest TR and TE)	**T2-Weighted (longest TR and TE)**	**Flair (much longer TR and TE**
TR (msec)	500	4000	9000
TE (msec)	14	90	114

TABLE 2.1: Repetition Time (TR) and Time to Echo (TE) for T1-weighted, T2-weighted and FLAIR images in msec [7]

Usually, T1-weighted images and T2-weighted images can easily be distinguished at a glance by looking at the Cerebrospinal Fluid (CSF). CSF looks brighter on T2-weighted images and darker on T1-weighted images [7].

The third type of MRI sequence that is used frequently is the **Fluid Attenuated Inversion Recovery (FLAIR)**, (as shown in Figure 2.8). The FLAIR sequence is very similar to a T2-weighted image, excluding that it takes a very long TR and TE time (as shown in Table 2.1). By taking a longer TR and TE time, the FLAIR sequence causes abnormalities to remain brighter while the normal CSF appears dark. This sequence is highly sensitive to pathology and differentiates between an abnormality and CSF more easily [7].

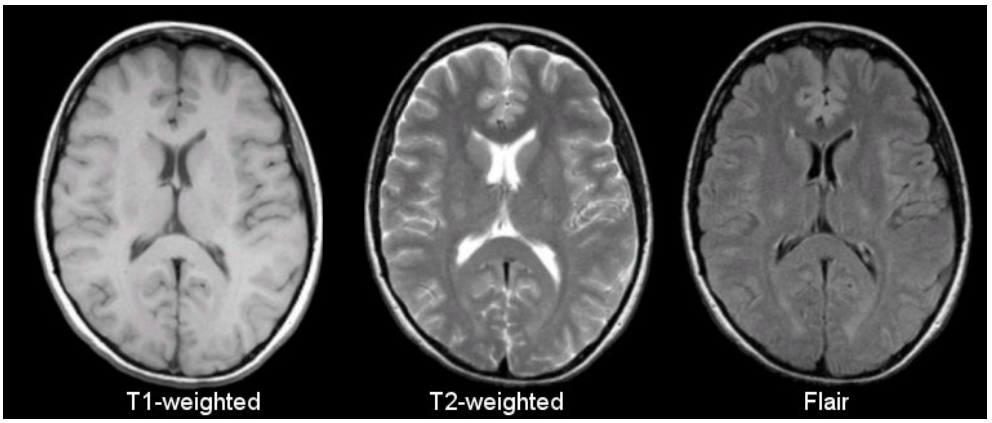

FIGURE 2.8: T1-weighted vs T2-weighted vs FLAIR (Brain images) [7].

2.6.4 What is the gadolinium contrast medium (MRI contrast agent)?

MRI contrast media are sometimes called **Gadolinium (GD)** contrast, agents, or 'dyes'. Each refers to chemical substances used in MRI scans. When the gadolinium contrast medium is injected into the patient's body, it enhances the quality of the MRI images. This allows for greater accuracy in reporting on how the patient's body is working and detection of any abnormalities or diseases present. The radiologist can then investigate the images and review the case with specialized medical practitioners. Gadolinium contrast media are composed of chemical bonds of atoms held together and complex molecules. The chemical bonds occur between a carrier molecule, also called a chelating agent, and a gadolinium ion. The chelating agent can prevent the toxic effects of gadolinium without negating its contrasting properties. Different types of gadolinium contrast media use a variety of chelating molecules. A contrast medium is injected in the vein as part of MRI scanning procedures and removed from the body through the kidneys [43].

Among the benefits of the gadolinium contrast injection is that it provides improved diagnostic accuracy in a number of conditions; namely infectious diseases and inflammations in the brain, spine, bones, and soft tissues. By providing a clearer image to the radiologist, the gadolinium contrast injection helps them better understand what the issue is and where it is located. There is a range of cancers and benign tumors that are best observed and evaluated after a gadolinium contrast dosage. Real-time scans showing the functions of blood vessels can be performed using a gadolinium contrast medium. In fact, there are several heart malfunctions that can only be evaluated this way [43].

MR imaging is superior when compared to CT for differentiating between tumors and perifocal edema, for defining the level of a tumor, and for showing the correlation of a tumor to critical nearby structures. Heavily T2-weighted sequences are crucial to discovering the tumor and edema extent, however the edema surrounding is not well separated from the tumor focus. Hence, this is usually achieved by using T1-weighted MRI with contrast enhancement (CE). This provides enhanced localization of the tumor focus and diagnostic information regarding the grade of the tumor, hemorrhage, necrosis, edema, and blood-brain barrier breakdown. The tumor and edema are effectively distinguished from the adjacent cerebrospinal fluid using proton density images. This is similar to the high-signal areas on heavily T2-weighted images. Such information is necessary for planning surgical procedures [43].

Human anatomy can be envisioned in all different planes: the axial, the sagittal, and the coronal. This is the main advantage of using an MRI (as shown in Figure 2.9) [7].

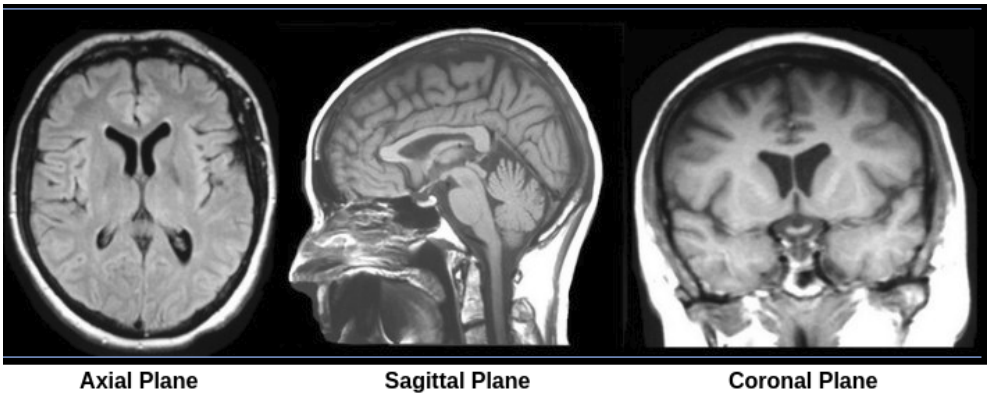

FIGURE 2.9: MRI Brain Images - The three planes of MRI scan: axial, sagittal and coronal (from left to right) [7]

Gadolinium (GD) can also be infused while performing T1-weighted imaging. GD is a contrast enchancing agent that is non-toxic and has paramagnetic properties. GD modifies the signal intensity by shortening T1 when injected at the time of scanning. Hence, GD appears very bright on T1-weighted images (as shown in Figure 2.10). GD enhanced images are particularly helpful in viewing vasular structures, tumors (breakdown in the blood-brain barrier), etc [7].

To identify the random movements of water protons, another type of MRI imaging has been designed, called **Diffusion-Weighted Imaging (DWI)**. The movement of water molecules is considerably restricted in the intracellular space, whereas they are relatively free in the extracellular space. This spontaneous movement, also referred to as

FIGURE 2.10: Comparison of T1-weighted vs T1-weighted with Gadolinium [7].

diffusion, quickly becomes restricted in ischemic (restriction in blood supply to tissues) brain tissue. At the time of ischemia, the sodium-potassium pump turns off, leading to sodium accumulation intracellularly. Water moves to the intracellular space from the extracellular space because of the osmotic gradient [7]. Intracellular restriction of water movement leads to an extremely bright signal when using DWI. As such, DWI is also an extremely sensitive imaging technique for detecting an acute stroke [7].

2.7 Conclusion

As discussed in this chapter, there are several types of brain tumors, classified based on varying behaviors, growth rates, and degrees of severity. While some are relatively slow-growing and can be treated effectively with surgery, others spread rapidly and are much more difficult to successfully treat. The more cancerous classes of tumors may still recur even after treatment and are very commonly fatal. It is paramount that tumors are detected early in order to increase the effectiveness of treatments and chances of the patients' survival [5] [4].

Developments in scanning and imaging techniques have become invaluable in the detection of tumors. Techniques such as Computed Tomography (CT) and Magnetic Resonance Imaging (MRI) are used to generate images of anatomical structures, such as the brain, providing a means to detect and analyze tumors present in patients' bodies. While CT is useful for initial screening assessments, MRI scans are crucial to further

diagnosis and determination of suitable treatments. Imaging procedures are aided by different dyes or agents, such as Gadolinium, that are injected into patients in preparation for scanning. These contrast media enhance the quality of MRI images and therefore improve diagnostic accuracy [38].

While these techniques are undeniably useful in diagnosing patients for tumor treatment, properly analyzing the images they produce can be a daunting task. Manual analysis of MRI images is tedious, time-consuming, and produces results of which accuracy is greatly dependent on the experience of the specialist performing analysis [38]. As early and accurate tumor detection is crucial to successful treatment, delays and errors in diagnosis can have severe consequences [38].

Computerized automation has long been used to tackle problems such as these. However, MRI image segmentation and tumor detection are complex tasks that require advanced computing techniques. In the next chapter, Artificial Intelligence (AI), Machine Learning (ML), and Deep Learning (DL) will be discussed. These concepts underlie more recent advancements in image segmentation and form the basis for the model proposed in this thesis.

Chapter 3

Artificial Intelligence (AI), Machine Learning (ML) and Deep Learning (DL)

3.1 Introduction

Artificial Intelligence (AI) encompasses any technique that enables computers to behave and make decisions as if they were human. It first entered the field in the mid-twentieth century and has been implemented in various ways since its inception. Two such implementations, considered to be subsets of AI, are Machine Learning (ML) and Deep Learning (DL).

ML centers around the ability of a system to learn "on its own" in place of receiving explicit instruction by programmers. There are varying forms of ML, such as unsupervised learning, regression, and clustering. Each of them are used to tackle the challenge of learning from data in different ways and are chosen by researchers based on the type of task at hand. They may be used to predict an outcome from a series of events or detect anomalies amongst vast amounts of data that might not feasibly be discovered by human examination. However, not all forms of ML are completely self-reliant, depending on researchers for guidance in particular areas or tasks.

DL is an extension of ML that adds the ability of systems to learn different features of data from that data. Previously, features were engineered by researchers using feature extraction algorithms, then incorporated into ML implementations. DL is able to extract these features automatically. This important capability brings DL ever closer to true AI. This chapter, provides an overview of how various forms of ML and DL work, their use cases, and what makes them such powerful tools.

3.2 Understanding the difference between AI, ML and DL

Machine Learning (ML) is a class of Artificial Intelligence (AI) that enables a model to learn from data instead of relying on explicit programming [44]. That said, Machine Learning (ML) is not an easy process. Tom Mitchell is a computer scientist, Machine Learning researcher, and professor at Carnegie Mellon University. The following definition of Machine Learning (ML) comes from his book, published in 1997 [8]:

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E" [8].

Experience (E) refers to the kind of data that needs to be collected, Tasks (T) refers to the kind of decisions the software needed to take, and Performance (P) is a measure for evaluating the results [45] (as shown in Figure 3.1).

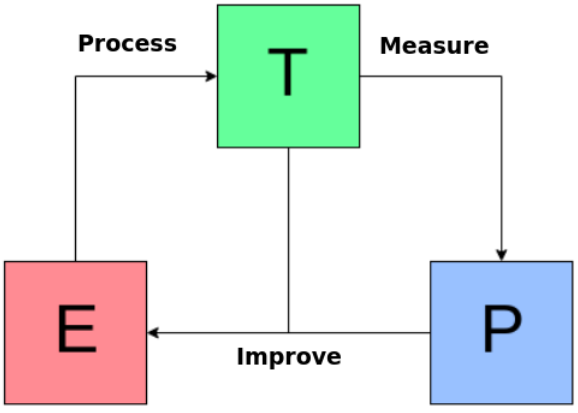

FIGURE 3.1: The Mitchell Paradigm, visualized [8]

Machine Learning (ML) uses a range of algorithms that continually learn from input data to improve, describe the data, and predict outcomes. More accurate models can be

built based on data ingested by Machine Learning (ML) algorithms. This type of data is referred to as training data. When the Machine Learning (ML) algorithm is trained with data, the output generated is called the "Machine Learning (ML) Model". When a Machine Learning (ML) model is provided with an input (the data), it outputs a prediction based on that data. Machine Learning (ML) is also necessary for the creation of analytical models [44].

In our day to day life, we work together with Machine Learning (ML) applications without even knowing it. Consider, for example, online shopping. When we use an e-business site to purchase products and begin to view them, the website automatically displays other, similar product recommendations that might possibly be interesting to you [44]. These recommendations are provided to the website by means of a Machine Learning (ML) model. The model will ingest the browsing history from your browser to learn what products were searched, then use this data along with purchasing data from other shoppers to display related products you might want to buy [44].

These models are well trained on the datasets prior to being deployed. There are two broad categories of Machine Learning (ML) models:

- **Online Machine Learning** models continuously adapt themselves as new data is acquired. The continuous learning process results in an improved accuracy of predictions in accordance with new data elements [44]. For example, a weather forecasting model can utilize the rapidly changing variables, namely the sensor information, weather details, and time, to quickly predict severe weather events (e.g. tornados) and trigger warning systems [44].
- **Offline Machine Learning** models include all other models. In contrast with Online models, these do not become adapted as new data is acquired. Once the model is deployed, it does not change [44].

Machine Learning (ML), Deep Learning (DL), and Artificial Intelligence (AI) are frequently mentioned terms when discussing analytics, big data, and advanced technology. AI can be understood as a way to describe intelligent machines that can "think" [44]. AI uses Machine Learning (ML) to imitate human intelligence. Speech recognition is the most widely used AI technology. For example, apple iPads and iPhones use the

popular personal assistant "Siri", which helps users make calls, find directions, and send messages, among other useful tasks. Siri is smart and capable of understanding natural language requests using Machine Learning (ML) technology. Other examples include the predictive capabilities of Tesla automobiles, Netflix recommendations, and the personal assistant "Alexa" used in Amazon's Echo [46].

Machine Learning (ML) and Deep Learning (DL) both are related to Artificial Intelligence (AI). AI was first introduced in the 1950s [10]. Machine Learning (ML) then came into the picture, followed by Deep Learning (DL), which is now used for solving complex real life problems [10]. As discussed earlier, a machine should learn the ability to act like a human being from experience, rather than explicit programming. Deep Learning (DL) is a subset of Machine Learning (ML) [47] (see: Figure 3.2). To summarize the difference between AI, ML, and DL:

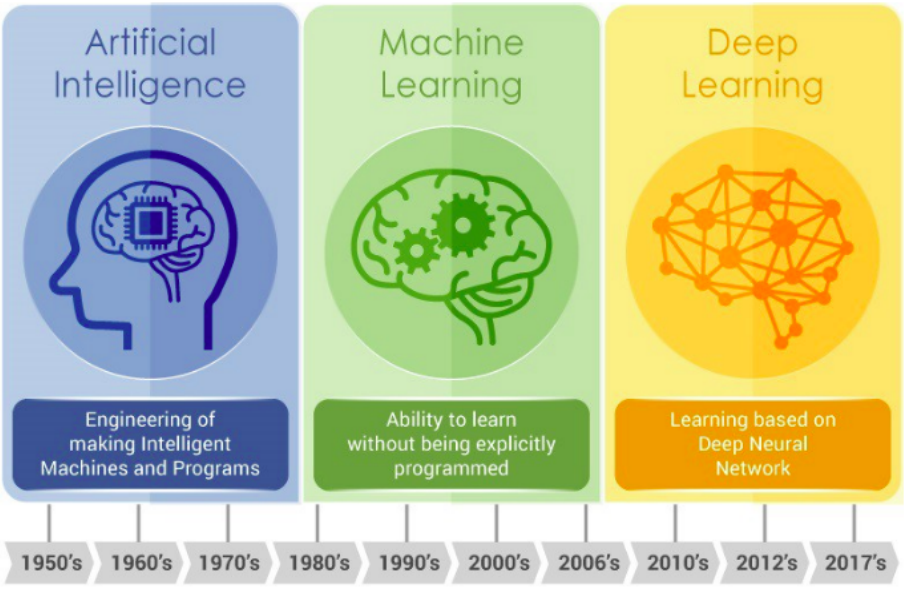

FIGURE 3.2: Artificial Intelligence (AI) vs Machine Learning (ML) vs Deep Learning (DL) - AI was introduced in the 1950s, followed by ML, then finally DL, which is used to solve complex real life problems [47].

1. Machines mimic human intelligence using Artificial Intelligence (AI) [48].

2. Artificial Intelligence (AI) can be achieved by Machine Learning (ML), which is a subset of AI [48].

3. Machine Learning (ML) can be implemented using Deep Learning (DL) techniques, which are a subset of ML [48] (as shown in Figure 3.3).

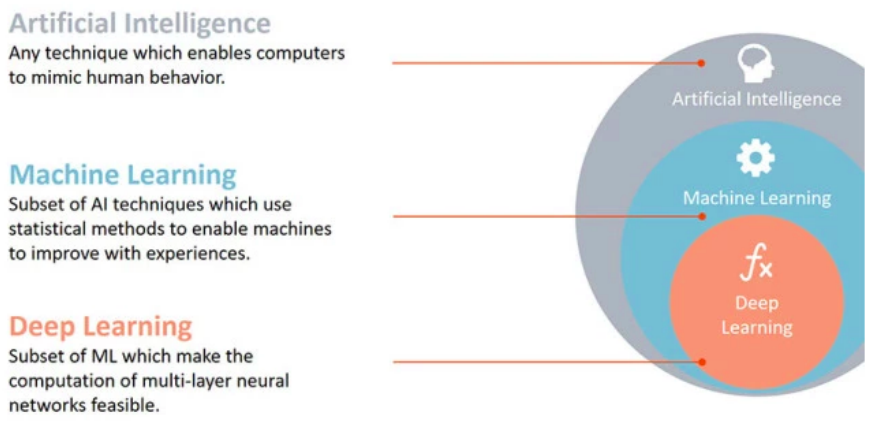

FIGURE 3.3: Machine Learning (ML) and Deep Learning (DL) are subsets of Artificial Intelligence (AI) [9].

3.3 Types of Machine Learning (ML)

At a high level, Machine Learning is essentially the process of teaching a computer program or an algorithm how to gradually improve on a given task. It is interesting to learn, understand, and recognize the forms of Machine Learning that we may experience in a world in wherein it became ubiquitous [10]. This could take the form of simply knowing what types of Machine Learning an average computer user may come across and how Machine Learning can be identified in such applications [10]. It is also important for professionals who create these applications to know the different types of Machine Learning. This will enable them to create the best learning environment for any given task a user may experience and understand why they have succeeded. There are three main recognized categories of Machine Learning: Supervised learning, Unsupervised learning, and Reinforcement learning [10] (as shown in: Figure 3.4).

3.3.1 Supervised Learning

The most popular paradigm of Machine Learning (ML) is supervised learning. It is a straightforward learning approach that is relatively simple to implement. Supervised learning is comparable to the way children are taught with the use of flashcards [10]. It generally starts with a defined set of data and some knowledge about how the data is classified. These kinds of Machine Learning (ML) models are designed to recognize patterns in data which could be applied on an analytical process. The data have labeled features that determine the significance of that data [44]. For instance, if we have millions

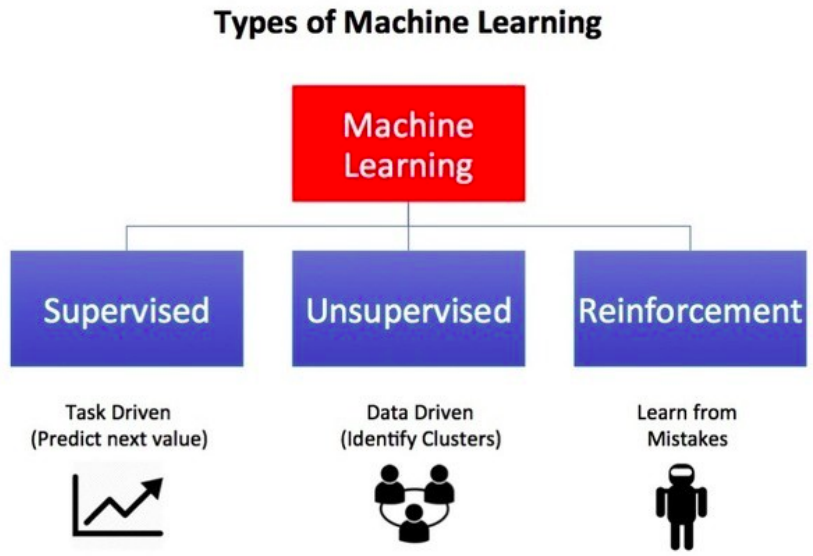

FIGURE 3.4: Types of Machine Learning (ML) [10].

of photos of animals along with descriptions of those animals as data, a Machine Learning (ML) application can be created that is able to distinguish one animal from another. With the help of labeled information about the types of animals, we can have hundreds of different species of animal as categories. By identifying its attributes and meaning, the animal data becomes well-understood by the users training the modeled data. This allows them to more easily fit the data to the details of the labels [44].

The following algorithm is used in supervised learning to study the mapping function in accordance with the variable input (x) to the variable output (y) [49]:

$$y = f(x)$$

The purpose of this algorithm is to evaluate the mapping function (f) as precisely as possible so that the variable output (y) for the dataset could be predicted anytime when there is new input data (x) [49]. The learning algorithm (model) is fed with labelled data (as shown in Figure 3.5) and the model is allowed to predict the label. Feedback must be given to the model for every prediction it makes, whether the right answer is predicted or not. for a certain period of time, the model will learn the connections (link) between the data and the label, also understands the nature of the labelled data. The model will be able to inspect any new data (never seen before) and predict an appropriate label for it. Examples of supervised learning performed in real-time include spam email classification and the facial recognition software utilized by Facebook [10].

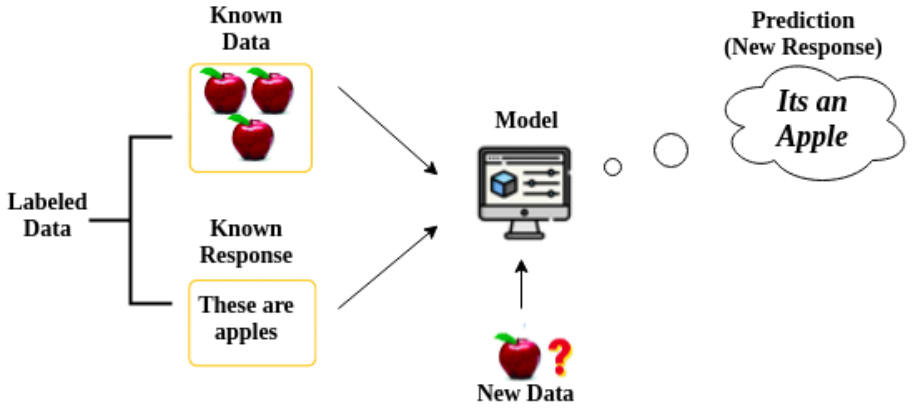

FIGURE 3.5: Supervised Learning [10]

3.3.2 Types of Supervised Learning

Supervised learning is further classified into two types: regression and classification. The key difference between them is that the output parameter is quantitative (or continuous) in regression, while it is categorical (or discrete) in classification [11] (as shown in Figure 3.6).

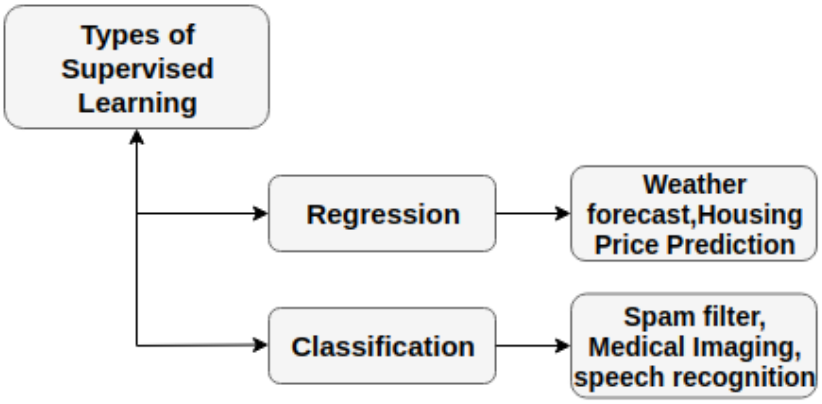

FIGURE 3.6: Types of Supervised Learning [11].

3.3.2.1 Regression

In this type of supervised learning, labelled datasets will be used to make predictions in a continuous form [49]. A regression model is a mapping function (f) that is calculated by an input variable (x) to produce a numerical output (y) [49]. Regression can examine the correlation between the dependent output variable and the independent input variable because it is a form of predictive modeling technique [11]. This is the technique behind such tasks as forecasting the weather, predicting housing prices, etc. Among the most

significant examples of this technique is housing price prediction, wherein the dataset (details of houses) is provided to the model and the housing price is predicted based on the input variables. Input variables in this scenario include the area of a house, the year it was built, nearby transport facilities, number of rooms, etc. As this is a regression task, the output will be continuous [49], [11].

3.3.2.2 Classification

In this type of supervised learning, labeled datasets will be used to make predictions in a categorical or discrete form. Classification involves predicting a class label. A classification algorithm estimates the mapping function (f) with an input variable (x) to produce a categorical output variable (y), where the mapping function predicts the category (y) [49]. In this technique, the classification algorithm learns from the given input data, then utilizes this learning to categorize the new observation (input data) [49], [11]. For example, one such classification problem is the categorization of a tumor as benign or cancerous. Another example is a spam filter, which checks whether an email is genuine or spam. As shown in Figure 3.6, spam filters utilize supervised learning algorithms that classify emails as spam or not spam using labeled emails (example emails with labels) previously fed (or inputted) into the model. The model learns to filter out any new email from the learning experience accordingly. In some cases, the user can also provide new labels, allowing the model to learn from user preference [10].

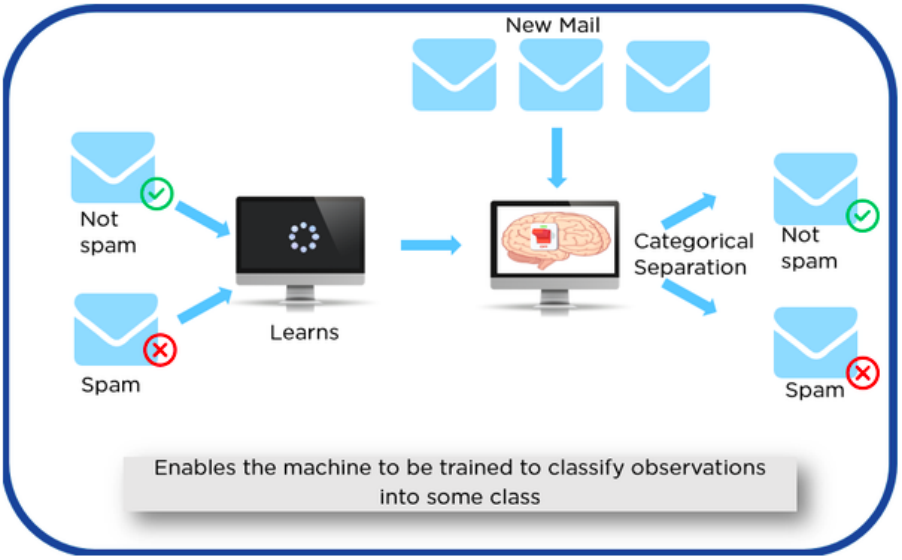

FIGURE 3.7: Spam Filter – Classification algorithm [10].

3.3.3 Unsupervised Learning

Unsupervised learning is a type of Machine Learning (ML), wherein raw data is used as input without any labels (referred to as unlabeled data) to train the model. As there are no labels associated with the data in this kind of learning, the algorithm tries to find patterns in the provided data. The objective of unsupervised learning is to investigate the data and find some patterns within. As shown in Figure 3.8, the unsupervised learning model is fed with unlabeled input data, builds an understanding of the properties of the input data, then finds a pattern to group similar data together [10]. Unsupervised learning is a fascinating area because the majority of the data available in this world is unlabeled [10]. Utilizing such intelligent algorithms that can process these unlabeled data, many industries can boost their productivity in a number of fields, raising the potential for increased profits [10].

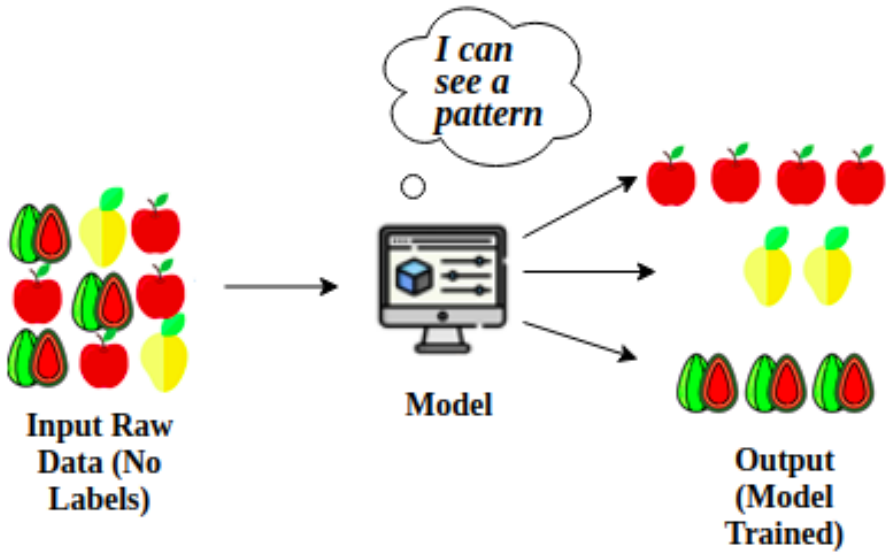

FIGURE 3.8: Unsupervised Learning [10].

For instance, if we had an unsupervised learning algorithm and provided as input a large database of all published research papers, the algorithm could learn how to group the papers in such a manner that we are always aware of current progress in a specific field of research [10]. While performing our own research, the unsupervised learning algorithm is fed input data, learning from that data, and can then make recommendations for related papers and works available for review and citation. As a result, productivity is highly boosted, helping to push that field of research forward. Unsupervised learning is described as data-driven because it is based upon raw input data and its properties [10].

The most common type of unsupervised learning technique is clustering.

3.3.3.1 Clustering

The method of grouping similar entities together is called clustering [11]. The objective of this technique is to group the data together to form clusters by identifying similarities in the data points [11]. Thus, when any new data is given as input, the algorithm makes a determination as to which cluster the data belongs [11]. For example, consider a list of transactions with details that include the transaction date, authentication type, customer name, and so forth (as shown in Figure 3.9). There are no specific labels present in the data set to indicate which of these is a fraudulent transaction [12].

date	customer	account	auth	class	zip	amount
Mon	Bob	3421	pin	clothes	46140	135
Tue	Bob	3421	sign	food	46140	401
Tue	Alice	2456	pin	food	12222	234
Wed	Sally	6788	pin	gas	26339	94
Wed	Bob	3421	pin	tech	21350	2459
Wed	Bob	3421	pin	gas	46140	83
Thr	Sally	6788	sign	food	26339	51

FIGURE 3.9: List of transactions (unlabeled data) [12].

If we use a clustering algorithm for this dataset, similar transactions will be grouped together to form a cluster. As shown in Figure 3.10, transactions with the same authentication type (pin), transaction day (wed), and class (gas) with an amount less than 100 will be grouped together [12].

date	customer	account	auth	class	zip	amount
Mon	Bob	3421	pin	clothes	46140	135
Tue	Bob	3421	sign	food	46140	401
Tue	Alice	2456	pin	food	12222	234
Wed	Sally	6788	pin	gas	26339	94
Wed	Bob	3421	pin	tech	21350	2459
Wed	Bob	3421	pin	gas	46140	83
Thr	Sally	6788	sign	food	26339	51

FIGURE 3.10: Forming clusters in a list of transactions [12].

3.3.3.2 Anomaly Detection

Anomaly detection is another type of unsupervised learning technique that identifies a rare event or item that is significantly different from the majority of the data. This

type of algorithm is used to identify unusual patterns in the data (also called outliers). Considering the list of transactions in the above-mentioned example, we can see that the transaction for customer Bob with an amount greater than 2000 is an unusual entry [12] (as shown in Figure 3.11).

date	customer	account	auth	class	zip	amount
Mon	Bob	3421	pin	clothes	46140	135
Tue	Bob	3421	sign	food	46140	401
Tue	Alice	2456	pin	food	12222	234
Wed	Sally	6788	pin	gas	26339	94
Wed	Bob	3421	pin	tech	21350	2459
Wed	Bob	3421	pin	gas	46140	83
Thr	Sally	6788	sign	food	26339	51

Unusual

FIGURE 3.11: Anomaly detection in a list of transactions [12].

Unsupervised training data has no specific outcome. It is not as easy as supervised learning in terms of evaluating the output and comparing it with the ground truth. As mentioned previously, unsupervised learning uses unlabeled data. We do not know the category it belongs to, however we can still find patterns and similarities amongst the data [12].

3.3.4 Reinforcement Learning

Reinforcement learning is a type of Machine Learning (ML) that is different from supervised learning and unsupervised learning [50]. It is an important type of Machine Learning that requires an agent, an environment, actions, and rewards [50]. By interacting with an environment and carrying out actions, then receiving rewards for those actions, the agent is able to learn how to behave [50] (as shown in Figure 3.12).

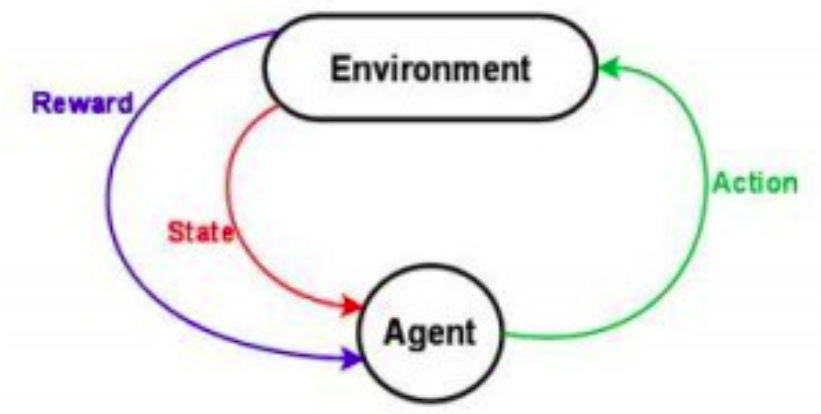

FIGURE 3.12: Reinforcement Learning [10].

In order to understand this concept, let us first consider how humans learn through interaction. For instance, suppose a child is approaching a fireplace, which feels warm and is considered a positive reward because the warmth feels good to the child. As a result, the child understands that fire to be a positive thing. However, if the child tries to touch the fire, it burns their hand, which is a negative reward. Thus the child understands that fire is positive when there is a sufficient distance, but is negative (burns) when getting too close to it. A computational approach to learning from actions is called reinforcement learning [50].

3.3.4.1 Reinforcement learning in the real world

1. **Video Games**

   The most common reinforcement learning from a human perspective is learning to play games. Super Mario Bros. is a widely-known example, where the environment is the game (a specific level) and the agent is the learning algorithm. The button states are the set of actions that the agent has. The updated state would be the frame change as the time is passing and the reward would be the score change. Hence, the connection between all of these components together provides a reinforcement learning scenario [10] (as shown in Figure 3.13).

FIGURE 3.13: Reinforcement learning process – Video Game [10].

2. **Industrial Simulation and Resource Management**

   Reinforcement learning can be used in robotic applications, where it is useful for the machines to learn to do their tasks instead of having the process hardcoded into them. This can be a safer option and less expensive. The machines can also be trained to use less electricity, which will save money. For example, Google's

data centers use reinforcement learning to balance resource needs against the need to satisfy power requirements [10]. Overall, reinforcement learning is an iterative process, where several rounds of feedback make the agent's strategy better. Such an approach can be applied to train robots to make decisions in tasks such as steering an autonomous vehicle or managing stocks in a warehouse [51] [13].

3.4 Deep Learning (DL)

Deep Learning (DL) is a new field of Machine Learning (ML) that has garnered huge success in different application domains. The main purpose of Deep Learning is to move Machine Learning closer to Artificial Intelligence (AI). Deep Learning typically uses Artificial Neural Networks (ANNs) to learn the data using multiple layers and different levels of abstraction [52]. This type of learning has shown rapid growth and is being applied to the majority of traditional application domains, along with some new areas that present greater opportunities. Deep Learning (DL) shows state-of-the-art performance results over conventional Machine Learning methods, particularly in the areas of image segmentation, computer vision, medical image processing, cybersecurity, bioinformatics, robotics, and others [13].

Conventional Machine Learning (ML) is differentiated from Deep Learning (DL) based on feature extraction. Handcrafted engineering features were used in conventional Machine Learning (ML) methods where the features are extracted using various algorithms [13]. Moreover, other "boosting methods" were often used on the ML algorithms during processing of features from the dataset [13]. In Deep Learning (DL), however, feature learning is an automatic process and the representation is hierarchical in multiple layers [13]. This is the main difference between Deep Learning and traditional Machine Learning approaches. Currently, Deep Learning is being implemented in almost every field; hence, why it is known as a "universal learning method" [13].

3.4.1 Why Deep Learning (DL)?

- **Universal learning approach**, as DL is applicable to nearly any field of study [13].

- **Robust**, because Deep Learning (DL) does not require precisely designed features. Instead, DL automatically learns the features that are required for the task [13].
- **Generalizability**, as different applications with different datasets can use the same DL approach. This is called transfer learning. This method is beneficial when the subject or issue is lacking sufficient data [13].
- **Scalability**, such as with ResNet (a deep network). ResNet was invented by Microsoft and contains 1202 layers; usually "implemented at a supercomputing scale" [13]. Development of such networks benefits from Deep Learning (DL) methods, which perform well at adapting to scale [13].

3.4.2 Performance vs Amount of Data

Traditional Machine Learning (ML) approaches show better performance results than Deep Learning (DL) when working with a relatively smaller amount of input data. However, as the input data volume increases past a given point, the performance of conventional Machine Learning (ML) methods becomes stagnant and the system is unable to learn further [13]. At the same time, DL approaches continue to increase in performance with increases in the data volume [13] (as shown in Figure 3.14).

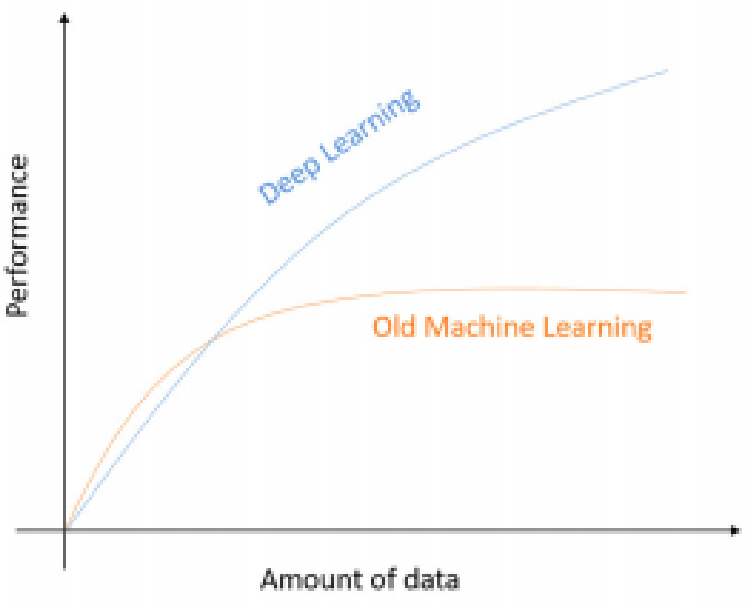

FIGURE 3.14: Performance of Deep Learning (DL) and traditional Machine Learning (ML) with respect to the amount of data [13].

3.4.3 Three Classes of Deep Learning (DL) Networks

Deep Learning (DL) refers to a wide class of Machine Learning (ML) architectures and techniques, which process the information using multiple layers. Depending on the architecture and technique used, it can be broadly classified into three major classes [52]:

3.4.3.1 Deep Networks for Supervised Learning or Discriminative Learning

In this type of Deep Learning (DL), the visible or available data will always have the target labels available in direct or indirect forms, thus providing a discriminative power for pattern classification purposes. Deep Neural Networks (DNNs), Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs) are used in supervised deep networks [52].

3.4.3.2 Deep Networks for Unsupervised Learning or Generative Learning

In this type of Deep Learning (DL), the visible or available data will have no target class labels. The unsupervised Deep Learning (DL) algorithm will analyze the patterns and find the purpose of the data. There exists a multitude of different deep network unsupervised learning algorithms, such as Restricted Boltzmann Machines (RBMs), Deep Belief Networks (DBNs), and others [52].

3.4.3.3 Hybrid Deep Networks

Hybrid Deep Networks is the third category of Deep Learning (DL) architecture. It makes use of both discriminative and generative model components. In existing hybrid architectures, generative learning models are mostly used with the help of discriminative learning models [52]. Recently, a hybrid Deep Learning framework using CNN for registration and integrated segmentation has been proposed [53].

3.5 Conclusion

In this chapter, two prevalent forms of Artificial Intelligence (AI): Machine Learning (ML) and Deep Learning (DL) were discussed . ML is used in a variety of different applications, oftentimes in ways of which its users are entirely unaware. From online shopping to digital personal assistants, fraud detection to weather prediction, ML has become ingrained in much of modern technology. Its differing methods of implementation, each learning from data in distinct ways, have proven adept at accomplishing tasks once thought to be exclusive to human intelligence.

Since the introduction of AI decades ago, numerous advancements have been made to increase the capabilities of its various applications. Most recently, developments in ML have given rise to DL, a new class of AI. DL models improve upon ML in a number of ways, not least of which is their ability to learn features of data instead of having them designed by those who implement them [13]. DL has also proven to be more generalizable, scalable, and better performing with increasingly large datasets than ML.

The performance benefits of DL over ML have been recognized in various use cases. For the purposes of this thesis, the cheif concern is its state-of-the-art performance results in image processing; particularly in the field of medicine. A number of DL models have been proposed and implemented in recent years. Their use in medical image segmentation has become increasingly popular among researchers and medical professionals. In the next chapter, several DL architectures will be covered and discuss the implementations of DL for image segmentation, which plays an important role in tumor detection and diagnosis.

Chapter 4

Deep Learning (DL) in Image Segmentation

4.1 Introduction

Deep Learning (DL) plays an important role in areas concerning computer vision, image classification, face recognition, object identification, and others [13]. Deep Learning (DL) algorithms are comprised of multiple layers of representation and abstraction, which helps in understanding input data, such as images, text, and audio. The idea of Deep Learning (DL) comes from Artificial Neural Networks (ANNs) and the Multilayer Perceptron (MLP), which includes a large number of what are called "hidden" layers. All of these layers together make up the structure of a Deep Learning (DL) algorithm [54].

Medical image segmentation and processing are fields of study in Deep Learning (DL) to which much of current research has turned its focus. Deep Learning (DL) is used to read input images using pixel values and predict the output using different Deep Learning (DL) architectures. One of the primary Deep Learning (DL) approaches used in areas of image processing is the Convolutional Neural Network (CNN) [13]. To reduce the training and execution computation times of a CNN model, special Graphical Processing Units (GPUs) are required [55].

In this case, a medical MRI image is given as input to the Deep Learning (DL) model to detect the tumor region. The Deep Learning (DL) model will process the image in

multiple layers using feature extraction, segmenting the tumor region as output. This chapter covered several Deep Learning (DL) models in detail and discuss how these techniques can be applied to medical image segmentation [55].

4.2 What is a neural network?

Perceptron is an example of a single layer neural network. A perceptron is an artificial neuron, introduced by scientist Frank Rosenblatt in 1958. It is a binary classifier function that takes several binary inputs and produces a single binary output. As shown in Figure 4.1, $x_1, x_2, \ldots, x_m$ are the inputs. To compute the output, Frank Rosenblatt proposed a simple rule. Weights $w_1, w_2, .., w_m$ were introduced as real numbers, showcasing the importance of the respective input to the output. Y is the output node, which calculates the scores by comparing the sum of the inputs multiplied by their respective weights ($\sum w_j x_j$) with the threshold value (parameter of the neuron; a real number) (Equation:4.1) to predict the binary output 0 or 1 [14].

$$Output = \left\{ 0 \quad \text{if} \quad \sum w_j x_j \leq \text{threshold} \quad \text{and} \quad 1 \quad \text{if} \quad \sum w_j x_j > \text{threshold} \right\} \tag{4.1}$$

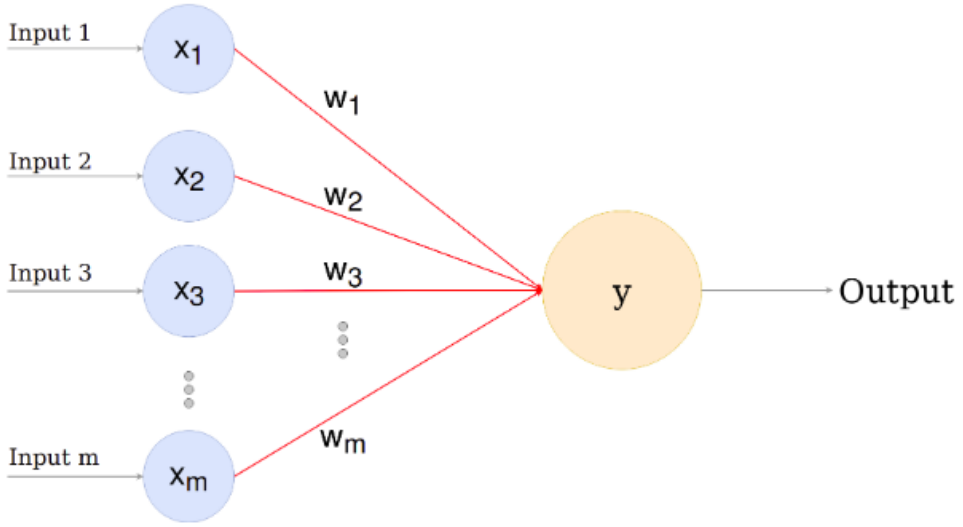

FIGURE 4.1: Perceptron – A perceptron is a single layer neural network [14].

4.3 The architecture of the neural networks

Perceptron is not a complete model for decision making. However, it is reasonable to conclude that a complex network of perceptrons can help make much better decisions than a single perceptron. The architecture of such neural networks is shown in Figure 4.2. In this architecture, the neurons within the input layer are called input neurons and the neurons within the output layer are called output neurons. In the case of Figure 4.2, there is only a single output neuron. The layers in the middle are called hidden layers, as the neurons in this layer are neither inputs nor outputs. This type of multilayer network is referred to as a Multilayer Perceptron (MLP) [15].

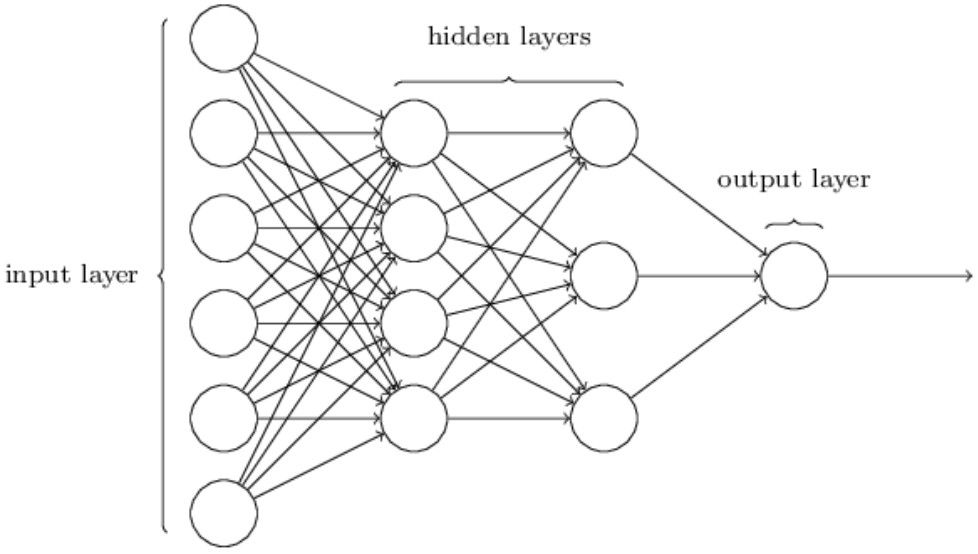

FIGURE 4.2: The architecture of Neural Networks - A four-layer network with two hidden layers [15].

4.4 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a sort of feedforward neural network. A feedforward neural network or Multilayer Perceptron (MLP) consists of numerous hidden layers in an Artificial Neural Network (ANN), typically described as Deep Neural Networks (DNNs) [32]. Research performed by Hubel and Wiesel in the 1960s [56] found that a special network structure using neurons can effectively diminish the complexity of Feedback Neural Networks. Hubel and Wiesel proposed the Convolutional Neural Network (CNN), which is an effective algorithm used in pattern recognition and image processing [54]. The main advantage of a CNN is the structure, which is very simple

and thus reduces the complexity of the network model [54]. The operation of a CNN and the elements involved in segmentation are shown in Figure 4.3. At a high level, they interact in the following manner: the image is fed as input to the CNN, which processes the image and outputs it in segmented form [16].

FIGURE 4.3: Elements involved in segmentation [16].

4.4.1 CNN Architecture

A CNN architecture takes a 2D (grey-scale) or 3D (RGB) image as input. CNNs treat each input image as three tensors with H rows, W columns, and D channels corresponding to RGB (red, green, blue) color channels for a 3D image [57]. The input is then sent through an iterative series of steps for processing (specifically feature learning). As shown in Figure 4.4, these processing steps are layers that alternate between "convolution + ReLu (Rectified Linear Unit)" and "pooling". The convolution, ReLu, and pooling layers are significant components of the CNN architecture [57]. Further explanation of each component of the CNN architecture will be provided in this chapter.

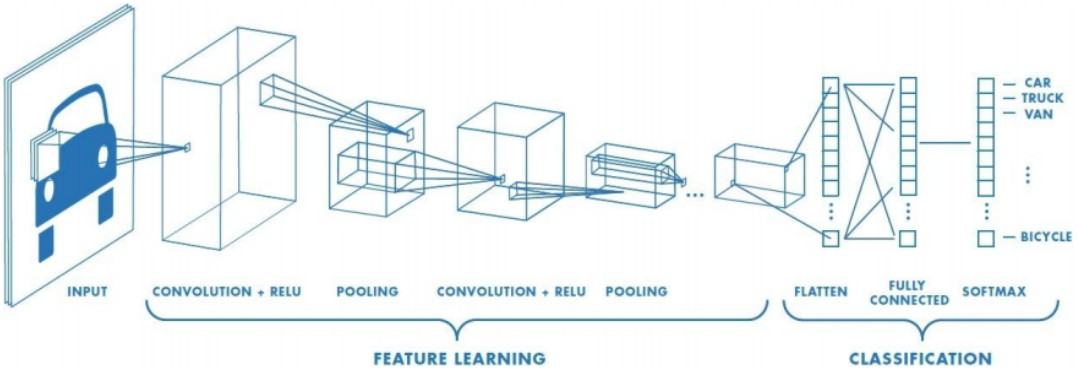

FIGURE 4.4: Convolutional Neural Network (CNN) – Sample Architecture [17].

4.4.2 Representation of 2D and 3D images in a CNN

Convolutional Neural Networks (CNNs) scan images in the form of pixels. The way that black and white (2D) images are scanned differs from colored (3D) image scanning (as shown in Figure 4.5). That said, 2D and 3D images share several fundamental similarities [16].

In a 2D or 3D image, colors are usually represented on a scale from 0 to 255, where 0 represents the minimum possible value and 255 represents the maximum possible value. Given that machines cannot learn colors, every color value is represented in binary 0s and 1s. This is the case for 2D images and 3D images both. Each pixel value in the image will be indicated as an 8-bit binary value, where 0 is represented as "00000000" and 255 is represented as "11111111". In the case of pixel representation, 0 signifies pitch black and 255 signifies pure white. The values in between represent different shades of grey [16].

As their names suggest, where 2D and 3D images differ is in their dimensionality. Pixels in 2D black and white images are represented by a single number in the 0 to 255 range. 3D RGB images, however, consist of pixels represented on three levels (as shown in Figure 4.5): one for red, one for green, and one for blue. Any color can be represented by different combinations of each of these. As such, a pixel from a 3D image would be presented to the neural network as three corresponding values, each ranging from 0 to 255 to indicate color concentration (e.g. 255, 105, 180 = "hot pink") [16].

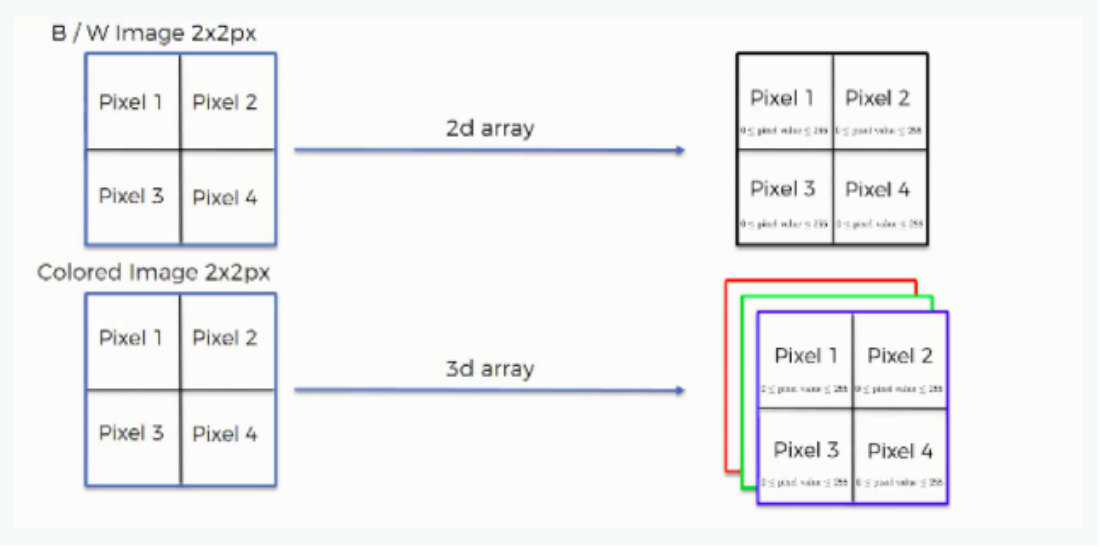

FIGURE 4.5: 2D and 3D images – Pixel representation [16].

4.4.3 The components of a CNN Architecture

The components of a CNN architecture consist of the convolution layer, activation function, pooling layer, flattening layer, fully connected layer, loss layer, and dropout. Each of these will be explained further in this section.

4.4.3.1 Convolution Layer

Convolution is an efficient method of feature extraction. The dataset produced from this process serves as a feature map [32]. Each filter or kernel works to identify a feature by filtering out where it exists in the original image. Use of convolutions reduces the need for manual operations and helps to fully describe the characteristics of the input [58]. A convolution operation is illustrated in Figure 4.6, wherein the dot products of the weights and the input size are integrated across channels. The number of filters is the same as the number of input volume channels and the output volume has the same depth as the number of filters. The number of pixels shifted over an input matrix is called the stride [17].

The input volume accepted by a convolution layer is of size **W1 x H1 x D1** (input size), where W is the width (columns in the pixel matrix) of the image, H is the height (rows in the pixel matrix) of the image, and D is the number of channels (3D or 2D). The convolution operation is performed on the input using the following parameters: K, F, S, and P. K portrays the number of filters used. In the example shown in Figures 4.6, 4.7, 4.8, there are 2 filters, W0 and W1. F represents the receptive field size. The receptive field is a part of the original input image that can potentially impact the output. In this Figures 4.6, 4.7, 4.8 example, the violet box size in the input volume is the receptive field size, which is the same as the kernel size (3 x 3). S represents the stride, which narrates the number of pixels the filter should shift over an input matrix [17]. P represents the amount of zero padding, which helps to control the size of the output feature map and guards the features that exist at the edges of the original matrix [59]. As shown in Fig: 3.6, the matrix with size 5 x 5 x 3 has a zero padding of 1, which changes the size to 7 x 7 x 3.

The size of the output volume **W2 x H2 x D2** (output size) in a convolution operation is calculated using the parameters detailed above. W2 is calculated using the formula

W2 = [(W1 − F + 2P) / S] + 1, H2 is calculated using the formula **H2 = [(H1 − F + 2P) / S] + 1**, and **D2** takes its value from **K**. As mentioned previously, K represents the number of filters [17].

In the example given below (shown in Figures 4.6, 4.7, 4.8), an input size of 5 x 5 x 3 (W1 x H1 x D1) + padding 1 yields a volume of size 7 x 7 x 3 (W2 x H2 x D2). The other parameters in this example are F = 3 (3 x 3 kernel), K = 2 (two filters), and S = 2 (two pixel stride). Factoring in these parameters, the output volume is calculated as follows: **W2 = [(5 − 3 + 2) / 2] + 1 = 3**, **H2 = [(5 − 3 + 2) / 2] + 1 = 3**, and **D2 = 2**. Thus, the **output volume W2 x H2 x D2 = 3 x 3 x 2**.

The stride (S = 2) movement is shown in Figures 4.6, 4.7, 4.8.

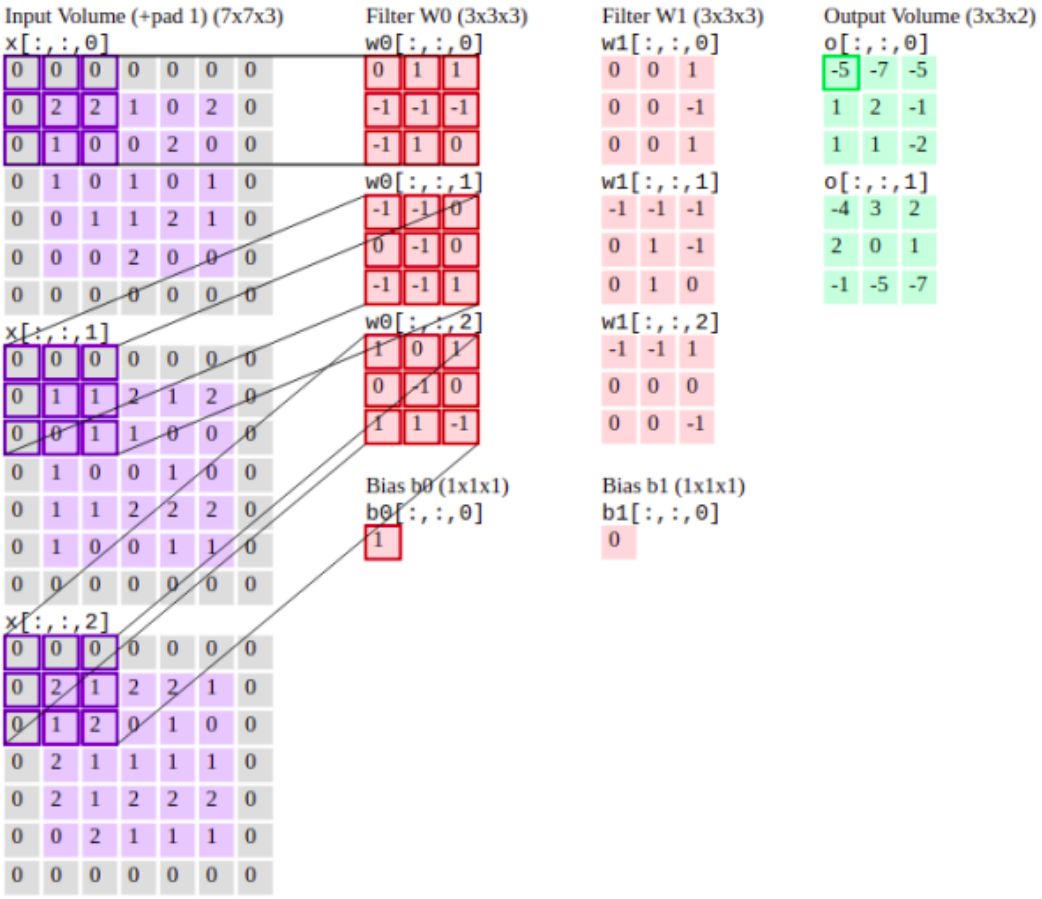

FIGURE 4.6: Convolution Layer - Stride movement 1 [17].

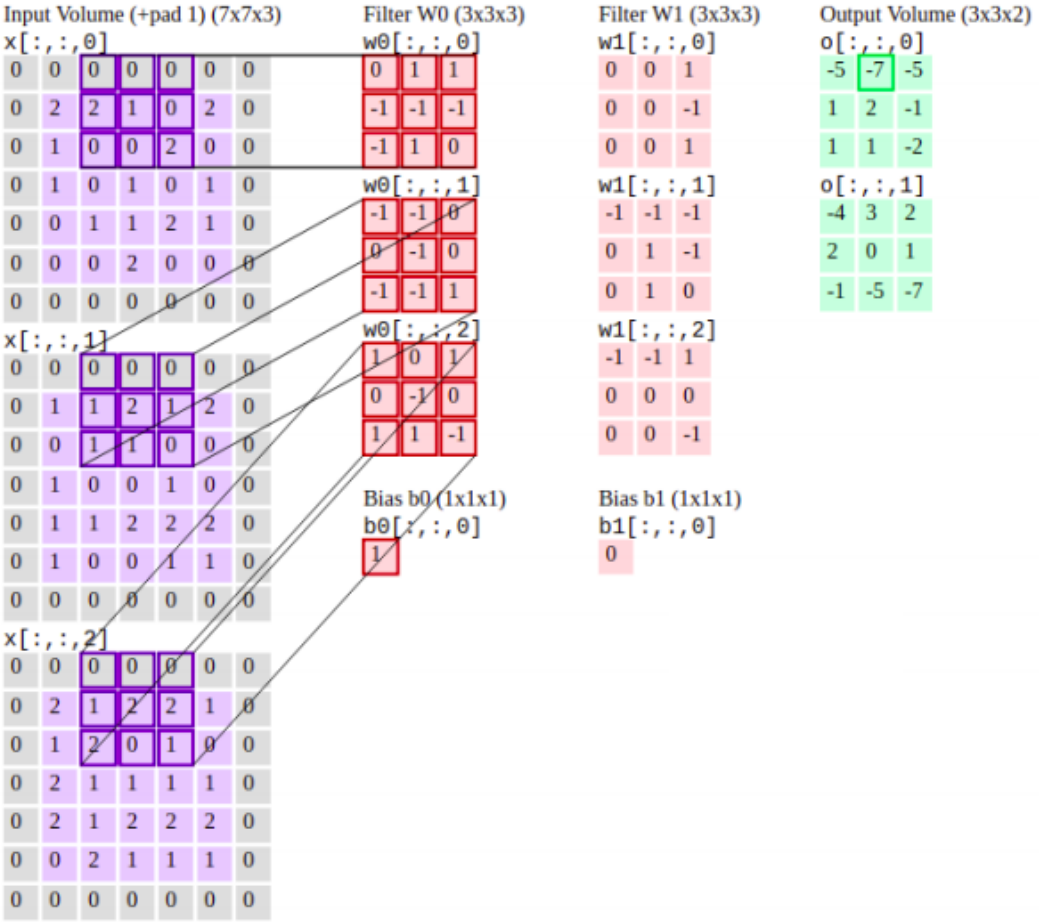

FIGURE 4.7: Convolution Layer - Stride movement 2 [17].

4.4.3.2 Convolution filters in Machine Learning (ML)

There are few convolution filters in machine learning-based image processing. These filters are used to blur an image, sharpen an image, or detect the edges of an image, among other operations [18] (as shown in Figure 4.9).

In CNNs, however, convolution filters are not predefined. The network model learns each filter value during the training process. The input dataset is labelled, allowing the CNNs to find more meaningful filter values from it. Each filter (as shown in Figures 4.6, 4.7, 4.8) in the convolution layer is randomly initialized. Over time, these filters eventually learn to detect different features in the image [18].

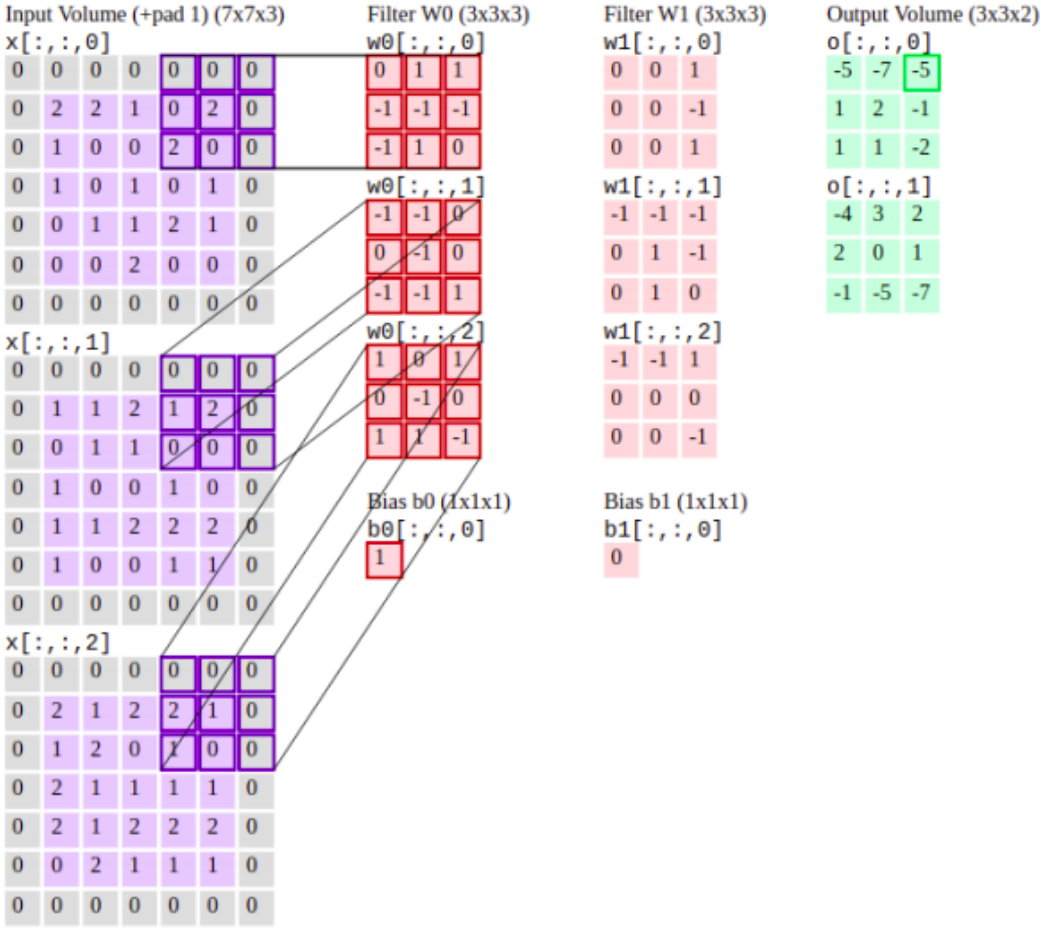

FIGURE 4.8: Convolution Layer - Stride movement 3 [17].

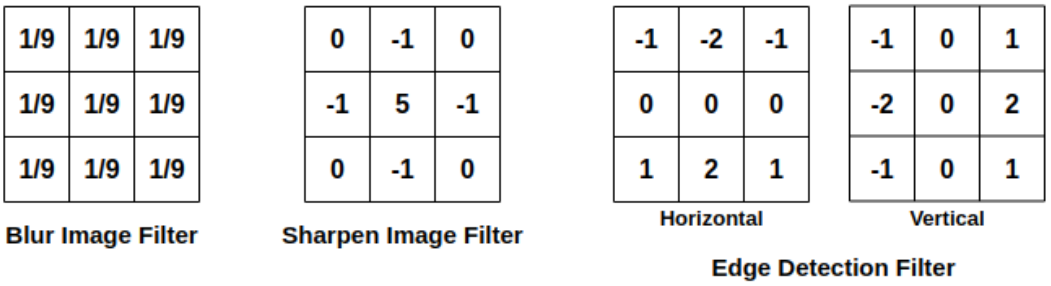

FIGURE 4.9: Convolution filters in Machine Learning (ML) - Blur image filter, Sharpen image filter and Edge detection filter [18].

4.4.3.3 Activation Function (AF)

A general issue with most learning-based systems is gradient flow. Gradients can be sharp in particular directions and gradual (if not zero) in some other directions. This can be problematic for technique selection with regards to learning parameters. To solve this gradient issue, activation functions have been introduced. An Activation Function (AF) is one of the parameters used in the computations of neural networks [60]. It is

used to calculate the sum of weighted inputs (cost of a neuron) and biases, which is then used to decide if a neuron may be released or not. Activation functions (AF) in certain instances are referred to as transfer functions, as they can be either "linear or non-linear" and are utilized to manage the outputs of the network model among various fields of computer vision [60]. Activation functions maintain the gradient values between 0 and 1 with the help of various mathematical functions [60]. If the gradient value is greater than 1, and in successive steps tends to infinity, the gradient will explode. If the gradient value is less than 1, and in successive steps tends to zero, then the gradient will vanish. Thus, choosing an appropriate activation function improves the performance of neural network computing. There are several kinds of Activation Functions (AF), each of which has undergone various evolutions over past years [60]. The sigmoid, softmax, and Rectified Linear Unit (ReLu) functions will be discussed in this section.

1. **Sigmoid Function**

   The sigmoid activation function is nonlinear and mostly used in feedforward neural networks. This is a finite differentiable real function that provides a degree of smoothness [60]. The sigmoid function is mathematically represented as [60] (4.2):

   $$f(x) = \frac{1}{1 + e^{-x}} \tag{4.2}$$

   In Deep Learning (DL) networks, the sigmoid function is generally applied in the output layers, which predict probability-based outputs [60]. The sigmoid activation function is applied in binary classification problems, in some neural network domains, and in logistic regression. Advantages of the sigmoid function include it being easy to understand and its applicability to shallow networks. The disadvantages of the sigmoid function include saturation of gradients, sharp damp gradients, and non-zero centered output, leading the gradients to propagate in different directions [60]. The sigmoid function is shown in Figure 4.10 below.

2. **Softmax Function**

   The softmax activation function computes the probability distribution of real numbers through a vector and is used in the computation of neural networks [60]. Softmax activation outputs the probability distribution as values ranging from 0

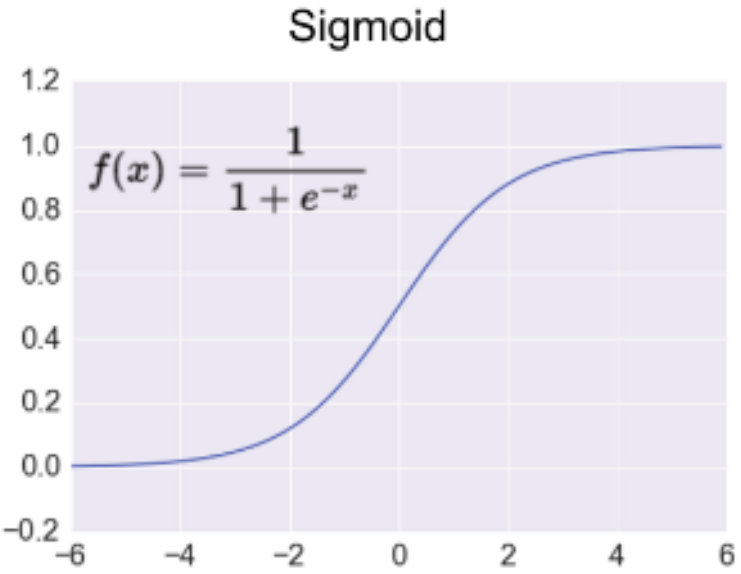

FIGURE 4.10: Sigmoid Function [19].

to 1. As this is a probability distribution, the total sum of these values will always be equivalent to 1 [60] (as shown in Figure 4.11). The softmax function is mathematically represented as (4.3):

$$f(x_i) = \frac{e(x_i)}{\sum_j e(x_j)} \tag{4.3}$$

This type of activation function is used in multiple class models, where the model returns the probabilities of all the classes and from which the highest probability is chosen. In Deep Learning (DL) methods, the softmax Activation Function (AF) is utilized mostly in the output layers [13]. The major difference between the softmax and sigmoid functions is that the softmax function is used for multi-layer classification tasks, whereas the sigmoid function is used in binary classification [60].

FIGURE 4.11: Softmax activation function [19].

3. **Rectified Linear Unit (ReLu) Function**

   The Rectified Linear Unit (ReLu) Activation Function (AF) was suggested in the year 2010 by Hinton Geoffrey and Nair Vinod [61]. It is the most commonly used Deep Learning (DL) activation function and produces state-of-the-art results. This activation function is a fast learner and provides better performance while comparing it with the sigmoid Activation Function (AF) [60]. The linear models characteristics are preserved with gradient-descent techniques when the ReLu Activation Function (AF) is utilized, as ReLu is itself a linear function (as shown in Figure 4.12). How the ReLu activation function operates is illustrated in Figure 4.13. ReLu carries out a boundary check on all input elements. As a result of this boundary check, the negative values (less than 0) are assigned zero values [60]. The ReLu function is mathematically represented as [60]:

$$f(x) = max(0, x) \begin{cases} 0, & \text{if } x_i < 0 \\ x_i, & \text{if } x_i \geq 0 \end{cases} \tag{4.4}$$

Changing the negative input values and forcing them to be zero eliminates the vanishing gradient problem. This ReLu function is used in combination with another Activation Function (AF) in the hidden layers of deep neural networks. Typical examples of ReLu activation function usage include classification and speech recognition applications [60].

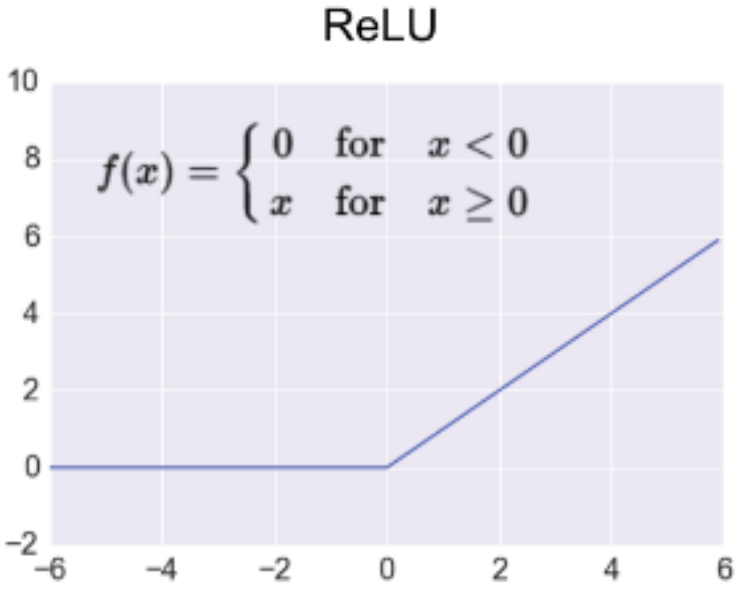

FIGURE 4.12: ReLu Function [19].

The ReLu function makes no attempts to compute complex mathematical calculations, in contrast with other activation functions. This assures faster computation, promoting the overall computation speed of the network model as a result. The primary benefit of using the ReLu function is speed enhancement [60]. The dropout technique is used along with ReLu activation functions to minimize the effect of overfitting, thereby enhancing the performance of neural networks.

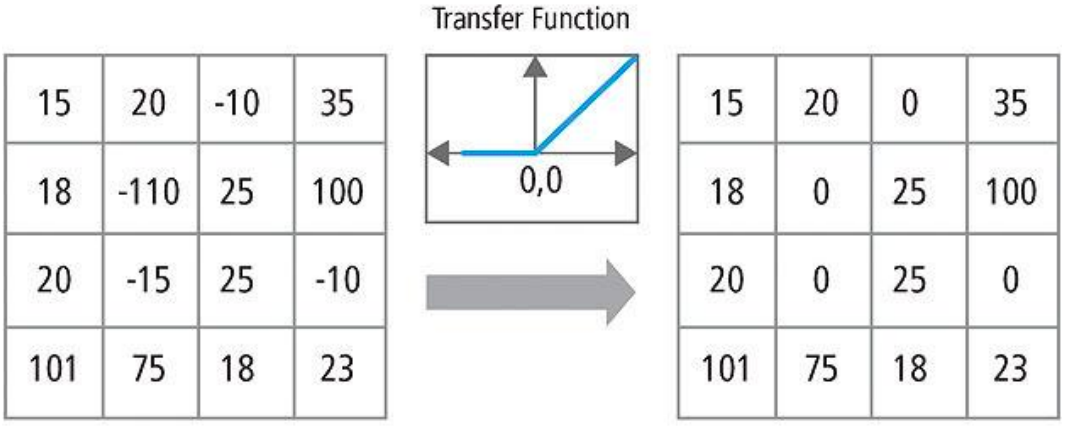

FIGURE 4.13: Operation of the ReLu activation function [20].

4.4.3.4 Pooling Layer

Pooling operations perform non-linear downsampling on activation maps provided by the convolution layer [17]. The pooling layer is an intermediary between multiple convolutional layers. There are two different types of pooling operations [13]: max-pooling and average-pooling. Max-pooling is frequently utilized by CNN pooling operations. Pooling prevents overfitting and reduces data dimension by focusing on local data points using a pooling window. Reducing the data dimension helps to decrease the number of calculation steps involved in the CNN [58]. The remainder of this section will cover how the pooling size is calculated.

Just as with the convolution layer discussed previously, the pooling layer also accepts input data of size **W1 x H1 x D1**. To calculate the pooling size, the receptive field size F and the stride S are required parameters. The output size **(W2 x H2 x D2)** of the pooling operation is calculated using formulas similar to those detailed in the section on the convolution later: $\mathbf{W2 = [(W1 - F) / S] + 1}$, $\mathbf{H2 = [(H1 - F) / S] + 1}$, and $\mathbf{D2 = D1}$ [17].

In the example given below (as shown in Figure 4.21), the input is of size **4 x 4 x 1 (W1 x H1 x D1)**, **F = 2 (2 x 2 kernel)**, and **S = 2** (shift by two). Taking these parameters into account, **W2 = [(4 − 2) / 2] + 1 = 2**, **H2 = [(4 − 2) / 2] + 1 = 2**, and **D2 = 1**. Therefore, the pool size **W2 x H2 x D2 = 2 x 2 x 1**.

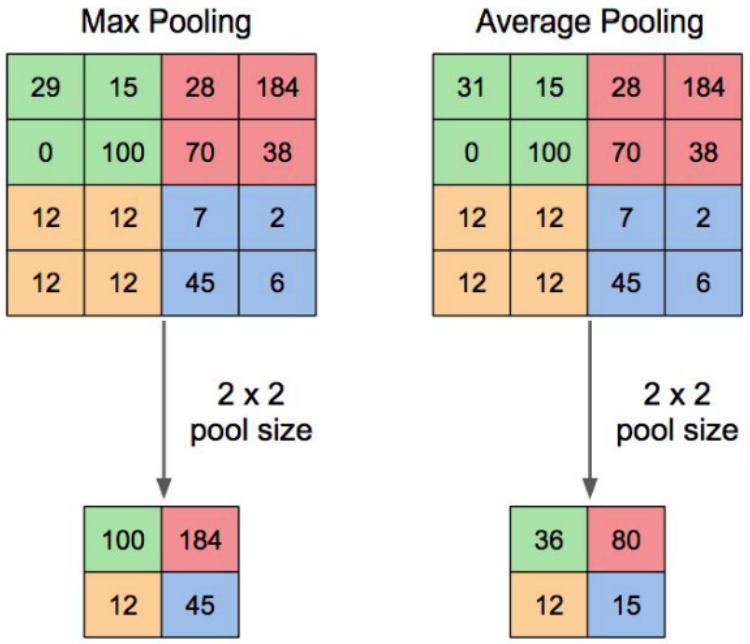

FIGURE 4.14: Pooling Operations – Max-pooling and Average pooling [17].

In max-pooling, the maximum value from the pool window is chosen. In average-pooling, the average round off value is chosen from each pool window (as shown in Figure 4.21).

4.4.3.5 Flattening Layer

Following completion of pooling operations, the pooled feature maps are used as input by the flattening layer. This layer flattens each pooled feature map into long vector maps (columns) (as shown in Figure 4.15). Flattening is done to prepare the data for input into an Artificial Neural Network (ANN) [16] (as shown in Figure 4.16).

The flow of data in a CNN starts with the initial input image, then proceeds to the convolution layer, the activation layer, the pooling layer, then the flattening layer, which provides a long vector input layer for a future ANN [16].

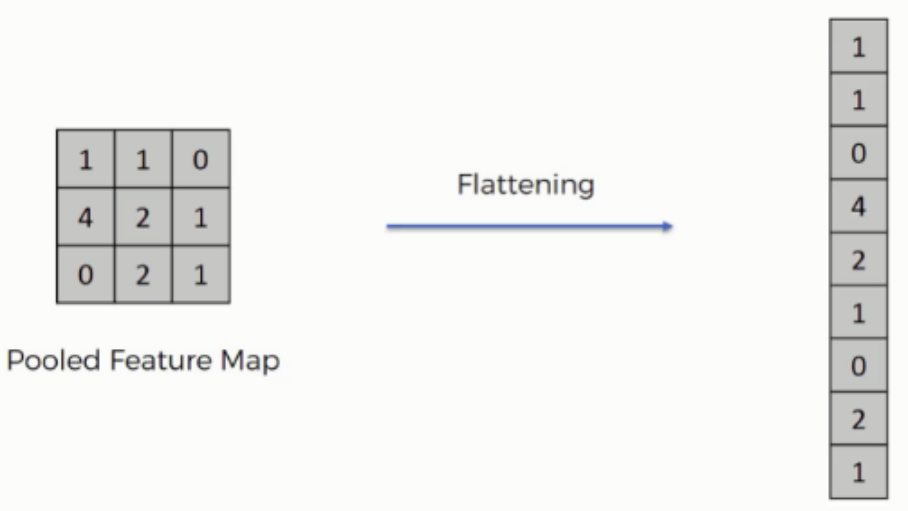

FIGURE 4.15: Flattening Layer [16].

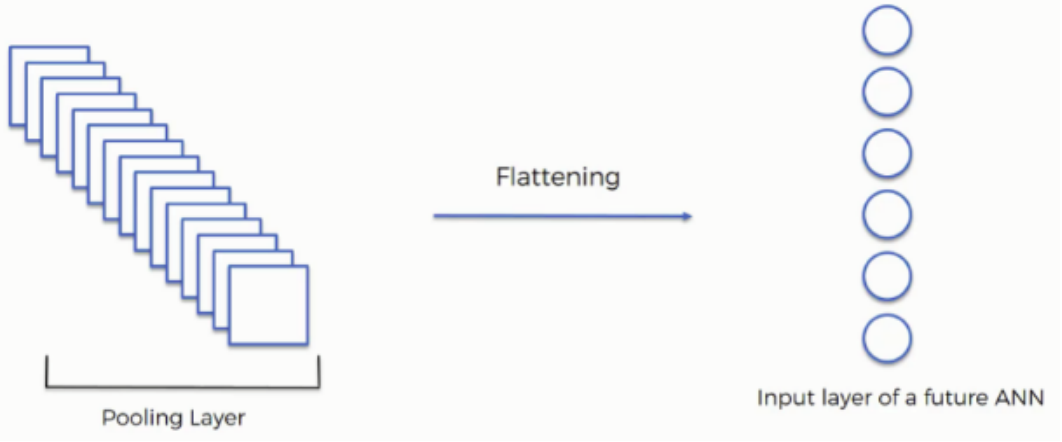

FIGURE 4.16: Input layer of a future ANN [16].

4.4.3.6 Fully Connected Layer

A fully connected layer is a regular neural network. This is the final learning phase in a CNN. Fully connected layers are mostly used in classification problems to map the extracted features to the desired outputs. The output of this layer is a vector, which is passed through a softmax activation layer to classify the output based on probability [17].

Consider the example shown in Figure 4.17. The CNN has two classes: dog and cat. The image provided as input into the CNN is a dog image. Using the above-mentioned layers (the convolution, activation function, pooling, and flattening layers), the neural network will have learned the feature maps. The fully connected layer uses the feature maps produced from the previous layers to detect the features of the classes. Using

this information, it calculates the probabilities of the dog and cat classes matching the detected features. The fully connected layer predicts that the dog image below does in fact show a dog based on its calculation that the dog class fits these features at a probability of 0.95, which is higher than the other class [16].

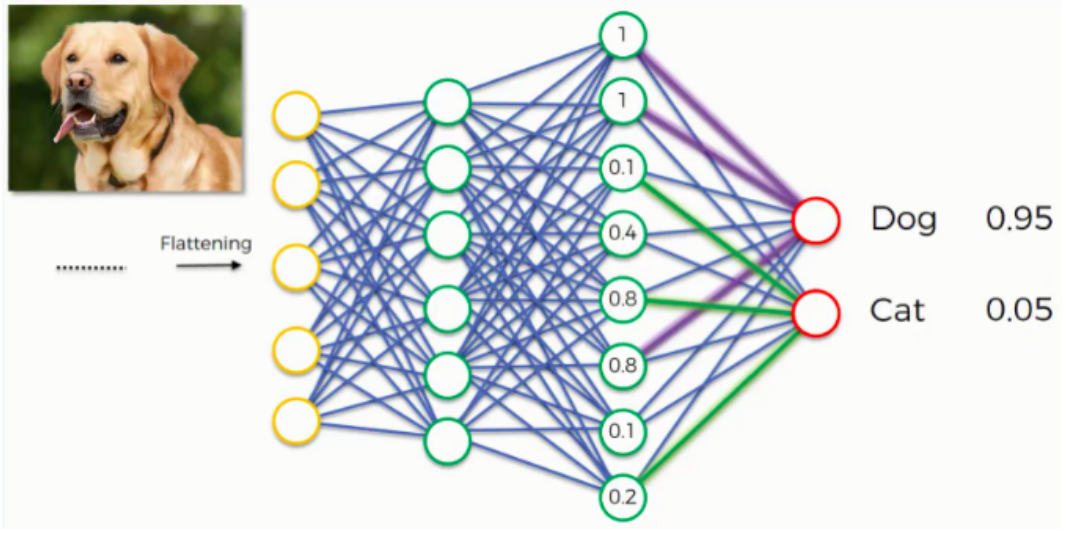

FIGURE 4.17: Fully connected layer – Class recognition [16].

4.4.3.7 Loss Layer

The loss layer is used when the network predicts the wrong class. For instance, consider the example discussed above. If the network predicts the image of the dog based on a probability of 78%, but the actual input image turns out to be a cat, an error must be calculated. In CNNs, this is often handled by what is referred to as a loss function [16].

Every neural network will require different types of loss functions, as the output format will vary from one network to another. In simpler terms, a loss function can be defined using two parameters: predicted output and true output (also called ground truth) [21], as shown in Figure 4.18.

By comparing the ground truth with the model's predicted output, the loss function will essentially compute the performance of the model (good or bad). If the predicted output (Y_pred) value is close to the true output (Y) (as shown in Figure 4.17), then there is little loss and the model has performed well. On the other hand, if the predicted output (Y_pred) value and the true output (Y) value are considerably different from each other, then there is a high amount of loss, indicating that the model needs more training [21].

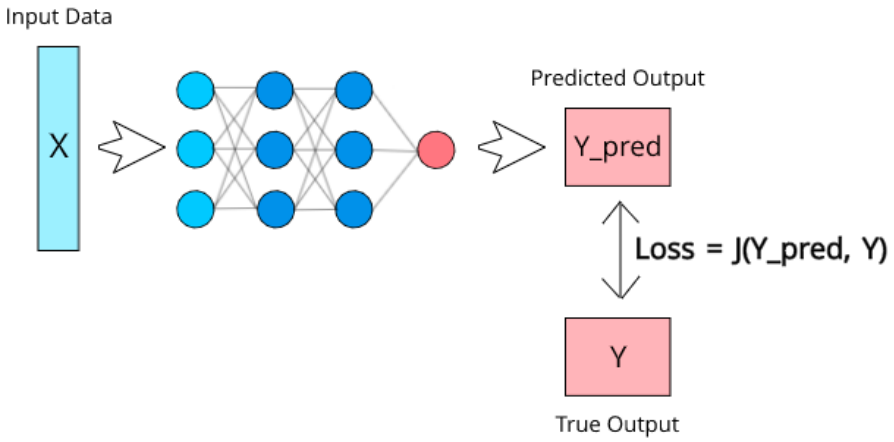

FIGURE 4.18: Loss calculation - By comparing the predicted output with the true output [21].

4.4.3.8 Dropout

Dropout refers to the process of choosing a random set of neurons during the training phase and dropping them out of the network (as shown in Figure 4.19). Dropout is used to effectively prevent overfitting with a large sized training dataset. In a fully connected layer (as shown in Figure 4.17), there will be multiple neurons connected to each other. Each neuron becomes codependent on the rest, controlling the power of individual neurons. This leads to overfitting of training data. To regularize the neural network, the dropout approach is used. Dropout reduces the interdependency among neurons [22].

Dropout is carried out in two phases: the training phase and the testing phase. In the training phase, dropout randomly ignores activations that have a probability p. In the testing phase, the dropout uses all the activations and scales them using a probability p. A sample result from applying dropout in a neural network is shown in Figure 4.17 [22].

The development of artificial intelligence in recent years has helped in addressing problems and challenges such as image segmentation, speech recognition, self-driving cars, and others. In particular, the CNN architecture has played a vital role in the field of image segmentation. CNNs have shown good performance in simple image segmentation problems, but have not yet shown any improvements with complex images. Medical images are quite complicated. Even expert physicians find identifying tumors or differentiating between different organs in the body to be difficult tasks. Therefore, for

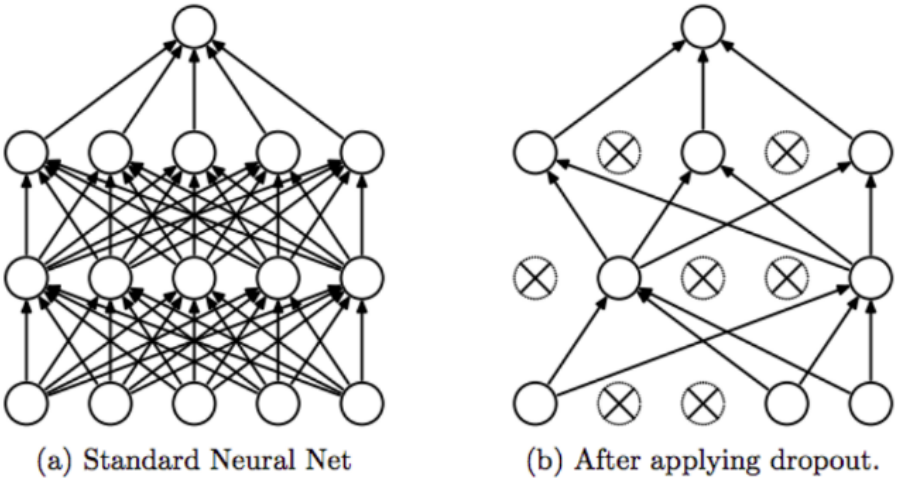

FIGURE 4.19: Dropout: a) Standard Neural Network - with fully connected neurons. b) After Applying Dropout - which reduces interdependency among neurons [22].

the purpose of medical image segmentation, a new model was designed. This model is known as U-Net. In contrast with the CNN architecture, U-Net performs pixel-based image segmentation, making it effective in this context.

4.5 U-Net

U-Net, a medical image segmentation method, was originally introduced by Ronneberger, Fischer, and Brox at the University of Freiburg, Germany in 2015 [23]. U-Net is a fully connected model that includes two symmetrical paths. The contracting path (also called the encoder) captures the context in the image. The expansive path (also called the decoder) provides exact localization and boundary detection using transposed convolutions. The U-Net model does not contain any dense layers. As such, it can accept input images of any size [62].

4.5.1 U-Net Architecture

In [23], the U-Net architecture was proposed as shown in Figure 4.20. The name U-Net is justified, as the architecture looks like the letter "U" and is symmetrical on both sides. The architecture consists of three key elements: the encoder (downsampling) path, the bottleneck layer, and the decoder (upsampling) path.

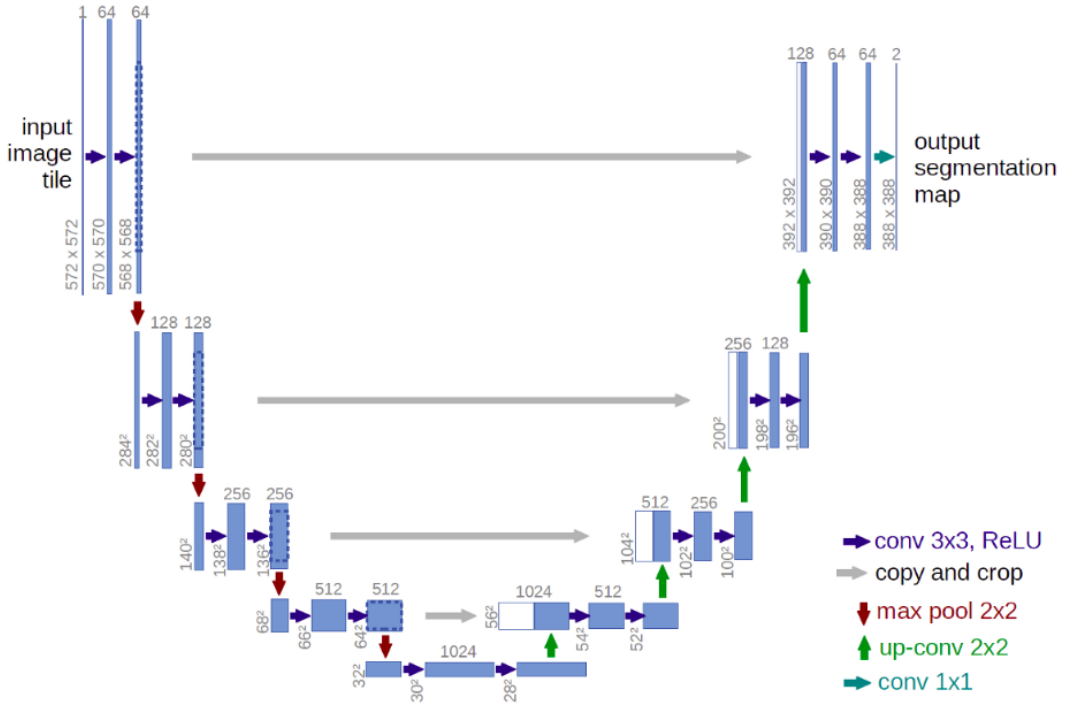

FIGURE 4.20: U-Net Architecture as proposed by Olaf Ronneberger, Philipp Fischer, and Thomas Brox at the University of Freiburg, Germany in 2015 [23].

- **3 x 3 convolutions**

  In this operation, 3 x 3 convolution filters are applied on the input image (as shown previously in Figure 4.6). On a grayscale image, a 3 x 3 x 1 convolution filter is applied . On a colored image, a 3 x 3 x 3 convolution filter is applied [63].

- **Upsampling**

  Upsampling is the opposite of a pooling operation, which downsamples the resolution by choosing a single value (using average-pooling or max-pooling) from each feature map. It instead performs as an unpooling operation, wherein a single value is distributed into a higher resolution. Different approaches may be used to implement upsampling and increase the resolution of a feature map [24] (as shown in Figure 4.21)).

- **Skip connection**

  Skip connections enable the flow of data to skip some layers in the neural network. The output of one layer is fed into another non-adjacent layer by skipping some layers in between. By using skip connections, features and other information captured in the initial layers can be utilized during upsampling operations when otherwise they would have been too heavily abstracted to use. In Deep Neural

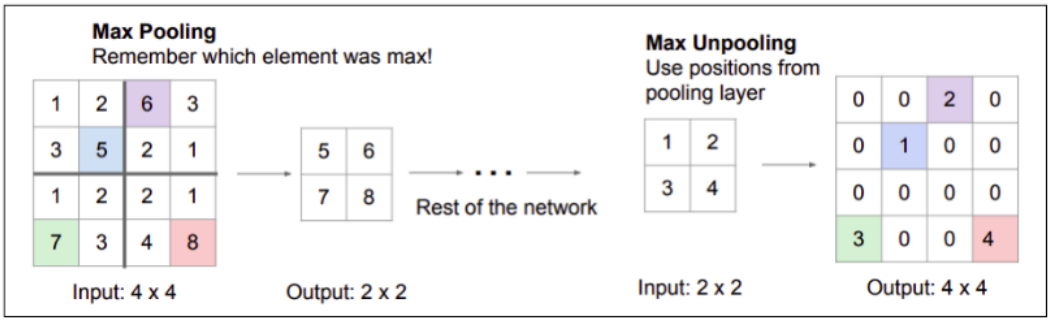

FIGURE 4.21: Max Pooling vs Max Unpooling [24].

Networks (DNNs), skip connections also help to traverse information more quickly [64].

- **Concatenation**

  During concatenation, feature maps are learned from the downsampling layer and concatenated with the feature maps from the corresponding upsampling layer [26]. As shown in Figure 4.20, the first convolutional layer has 64 feature maps. These are concatenated with the 64 feature maps in the upsampling layer, resulting in a total of 128 feature maps [65].

4.5.1.1 The encoder or downsampling path

The basic idea of the encoder path is to gradually decrease the input image size, which increases the depth and essentially helps the neural network to learn "WHAT" information is present in the image [13]. This is achieved by following two consecutive "3 x 3 convolutions + activation function", followed by a "2 x 2 max-pooling" function [62], [66]. As shown in Figure 4.20, the input image is sized at 572 x 572 x 1 and has 64 initial feature maps. Two consecutive 3 x 3 convolutions are applied, which reduces the image size gradually. This is followed by a max pooling layer, which doubles the size of the feature map. This helps the network to learn the context of the complex structures in the input image effectively. The values on top of the box in the architecture diagram (as shown in Figure 4.20) represent the feature maps. The values on the left side of the architecture diagram represent the current image size. In the contracting path, the feature maps are doubled, starting from 64 to 128, 256, 512, and so forth. Meanwhile, the image size gradually reduces from 572 x 572 to 32 x 32 [62], [66].

4.5.1.2 The bottleneck layer

The bottleneck layer is the intermediary layer between the contracting and expansive paths. This layer is composed of two 3 x 3 convolutions with dropout, accompanied by a 2 x 2 up-convolution [13].

4.5.1.3 The decoder or upsampling path

The decoder path gradually increases the image size and reduces the depth, which helps the network model to recover the [26] "WHERE" information. This provides precise localization information in the image. This "WHERE" information is learned by the model using two consecutive "3 x 3 convolutions + activation function", "concatenating" the feature maps from the downsampling path [26], followed by "2 x 2 up convolution" (also referred to as deconvolution). The deconvolution layer, or the transposed convolution, does the exact opposite of a normal convolution layer, where the input volume size is low-resolution and the output volume is high-resolution [62], [66].

As shown in Figure 4.20, the image size after completion of the contracting path is 32 x 32 x 1024, where 32 x 32 is the low-resolution image and 1024 is the dimension of the feature map [26]. Two consecutive 3 x 3 convolutions are applied with the concatenation of feature maps from the downsampling path, accompanied by a 2 x 2 deconvolution, which reduces the dimension of the feature map by half from 1024 to 512, 256, and so forth [26]. Concurrently, the image size gradually increases from 32 x 32 to 388 x 388 [62], [66]. To get the precise location information in the image, the decoder uses skip connections by concatenating the output of the deconvolution layer with the feature maps from the encoder in the same layer. The white boxes on the decoder side indicate concatenation [62], [66].

As mentioned previously, the primary advantage of using the U-Net-based approach for complex medical image segmentation is that it performs pixel-based image segmentation. This model has achieved good performance in the field of medical imaging and much research has been performed to improve its efficiency. The U-Net model uses skip connections between the downsampling and upsampling path by "concatenating" the feature maps [66].

4.6 Residual Networks (ResNet)

ResNet, on the other hand, uses skip connections by "adding" the input directly in every block, which resolves the vanishing gradient problem [66]. In this section, the workings of a residual block (ResBlock) are explained by comparing it with a regular block (as shown in Figure 4.22). The Residual Network (ResNet) architecture is made up of a series of ResBlocks with skip connections, which makes it different from other CNN architectures. The input is denoted as $\mathbf{x}$, from which $\mathbf{f(x)}$ is obtained after learning the input. This happens in a regular block. Unlike a regular block, however, there are two connections from the input end $\mathbf{x}$ in a residual block. One connection goes through a series of convolutions, batch normalization, and activation functions, while the other connection skips the series of convolutions and functions. The latter is known as a cross connection or skip connection. Finally, the outputs of both connections $\mathbf{f(x)}$ and $\mathbf{x}$ are added together: $\mathbf{f(x)} + \mathbf{x}$. The ResBlock's inputs and outputs will correspondingly each have the same number of channels [25].

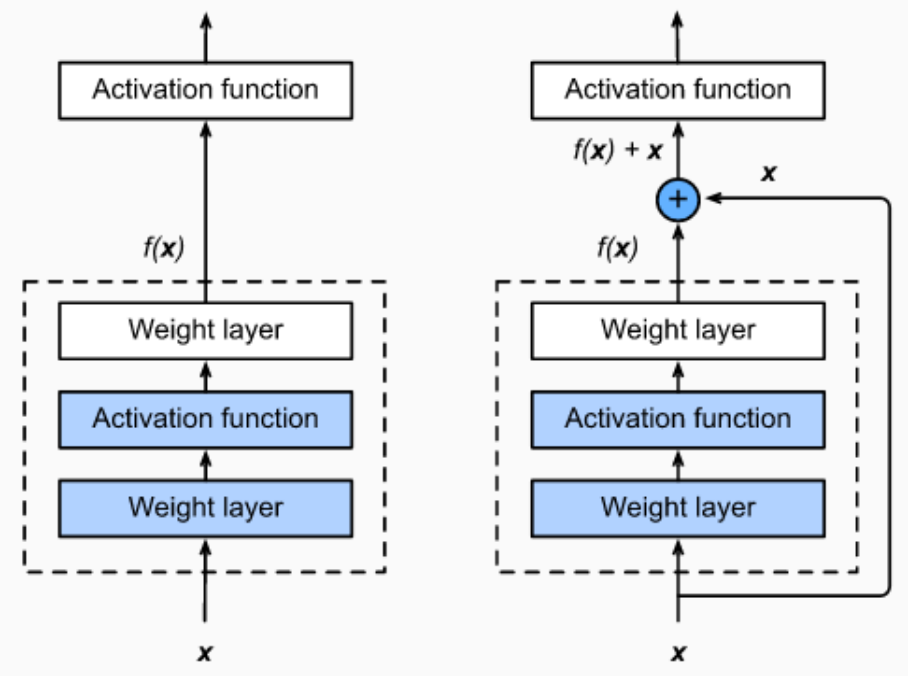

FIGURE 4.22: Left: Regular block, Right: Residual block [25].

If the number of channels needs to be changed , additional 1 x 1 convolution layers can be used to adjust the number of channels and resolution by 1 x 1 convolutions each (example shown in Figure 4.23). This transforms the input into the desired shape [25].

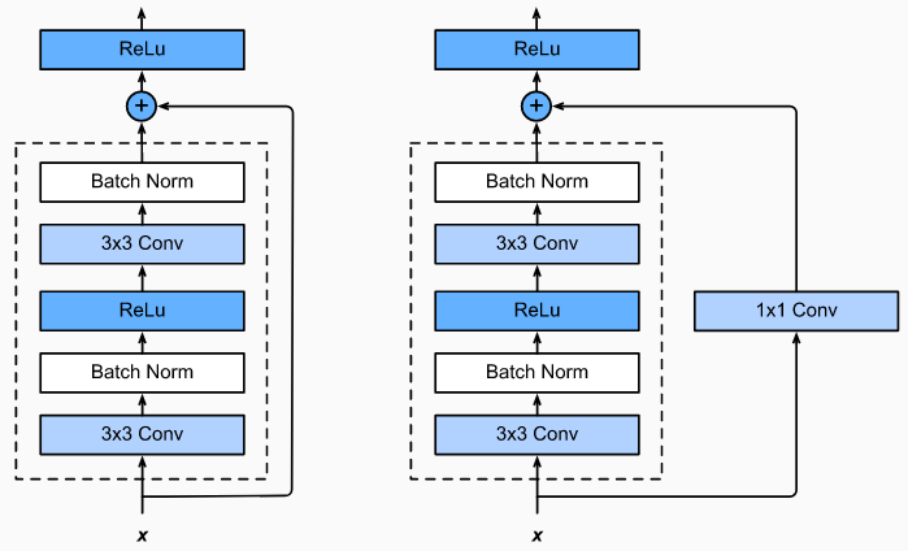

FIGURE 4.23: Regular ResBlock (Left), ResBlock with 1 x 1 convolution (Right) [25].

In a ResBlock without a 1 x 1 convolution, the input goes through all the regular convolutions and functions accordingly. In the other connection, the input goes through a 1 x 1 convolution before the addition operation. This helps with changing the resolution of the image. For example, if the input image size is 6 x 6 x 3 (H x W x D), the output can be transformed using a 1 x 1 convolution to a different size 3 x 3 x 6. This results in a height and width reduction and an increase in the number of channels [25].

ResNet decomposes the function into a simple linear term and a complex nonlinear term [25], as shown below.

$$f(x) = x + g(x) \tag{4.5}$$

However, what if the model wants to go beyond the two terms x and g(x)? To solve this issue, the DenseNet model was introduced.

4.7 DenseNet

DenseNet was introduced by Huang, Liu, Van Der Maaten, and Weinberger in 2017 [67]. This model provides a consistent improvement in accuracy without degrading performance. DenseNet has achieved state-of-the-art performance results with extremely complex datasets. The primary difference between DenseNet and ResNet is that DenseNet

concatenates the output, whereas ResNet adds the output [25] (as shown in Figure 4.24).

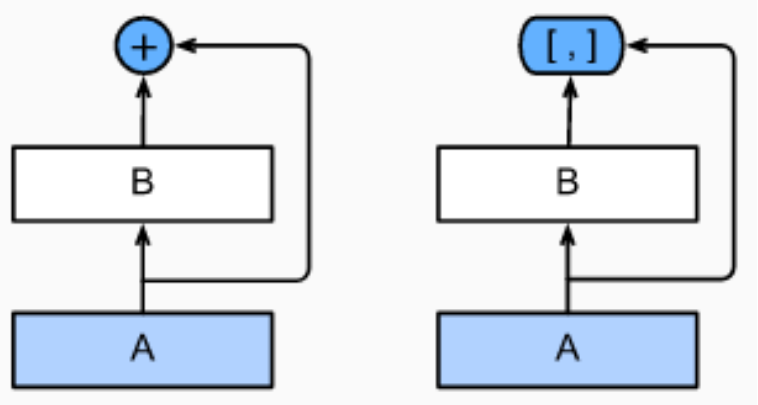

FIGURE 4.24: The Primary difference between ResNet and DenseNet: ResNet (left) uses addition and DenseNet (right) uses concatenation with skip connection [25].

Each skip connection in the DenseBlock makes the network denser. As a result, a ResBlock becomes a DenseBlock and the whole network becomes a DenseNet. A DenseNet contains a series of DenseBlocks. A single DenseBlock is shown in Figure 4.25 [25].

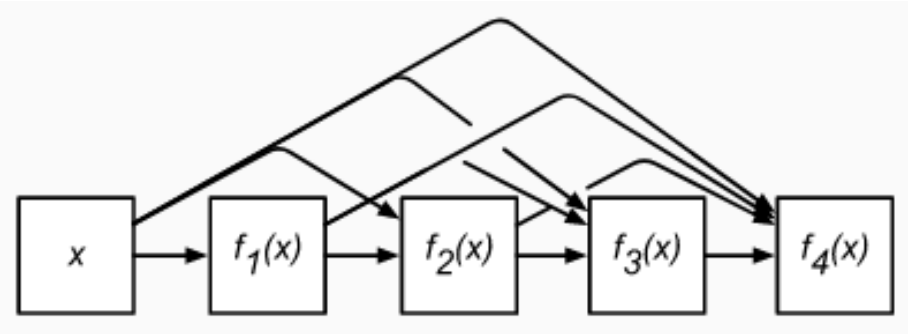

FIGURE 4.25: A DenseBlock [25].

The DenseNet concatenation (as shown in Figure 4.25) is represented as:

$$X \longrightarrow [X, f_1(X), f_2(X, f_1(X)), f_3(X, f_1(X), f_2(X, f_1(X)), ...] \tag{4.6}$$

The network becomes quite dense from the use of a concatenation operation in this model. This is especially true for the last layer in the DenseBlock chain, which is densely connected to all of the previous layers.

DenseNet is composed of two major components: DenseBlocks and transition layers. DenseBlocks use batch normalization, activation functions, and convolution layers with concatenation, thereby increasing the number of channels [32]. As such, using too many

DenseBlocks will lead to an extremely complicated model. Thus, a transition layer is used after a DenseBlock to avoid this scenario. A 1 x 1 convolution is used by the transition layer to decrease the number of channels. Additionally, pooling is utilized to downsize the volume of the image (H x W) [26]. As a result, the complexity of the network is controlled [25].

4.8 Conclusion

Deep Learning (DL) utilizes numerous algorithms to help analyze multiple representation and abstraction layers simultaneously. These algorithms make use of mathematical equations, establishing critical parameters to better analyze the input data. Based on predetermined classes, Deep Learning (DL) models predict "1" (true) or "0" (false), where "1" indicates that sufficient features have been detected to attribute a particular class to the data consumed by the model. The greater the amount of input data provided, the better Deep Learning (DL) models become at predicting the presence of key features and correctly classifying that data.

In the context of brain tumor detection, the data provided as input to these models consists of images produced by brain scanning methods, such as Magnetic Resonance Imaging (MRI). As more images of brain tumors are fed as input and processed through each layer in the model, the easier it becomes for the model to correctly predict the tumor region in the image. In this chapter, several Deep Learning (DL) architectures, having both differing and shared components were discussed, and each having their own advantages and disadvantages.

Two Deep Learning (DL) architectures that show promise in the area of brain tumor detection are U-Net and DenseNet. U-Net was designed for medical image processing and has already performed well in that field due to its pixel-based segmentation method and advancements in efficiency. DenseNet is another promising architecture. Its use of DenseBlocks has proven advantageous at handling complex data sets with high levels of accuracy while mitigating performance degradation. One model that has been proposed for 3D brain tumor image segmentation combines the U-Net architecture with Dense-Blocks, incorporating the advantages and state-of-the-art results produced by each of

these respective architectures. The effectiveness of this combination with regards to image segmentation and brain tumor detection will be further discussed in this document.

Chapter 5

Image Segmentation: Related Works

5.1 Introduction

Pixelation of image data into segments and partitions is a process referred to as image segmentation. It is a major component of certain practical applications of machine learning, in which computational tasks are applied to a range of data to build a model. There are two types of image segmentation that will be covered in this chapter; namely salient object segmentation and medical image segmentation. Both can be used to detect and analyze the anatomical structures of tumors in order to diagnose and better treat patients.

In recent years, computational power has become cheaper and has allowed researchers to take advantage of the vast amount of medical images that have been collected via different scanning methods. Graphics Processing Units (GPUs) are leveraged to process these images, training machine learning models for more accurate and efficient detection. One of the best-known machine learning architectures for medical image segmentation is U-Net. It breaks down the image data into multiple segments, allowing medical professionals to pinpoint the presence and exact locations of tumors.

5.2 Image Segmentation

Image segmentation is a popular machine learning concept. This is the mechanism by which the image pixels are partitioned into "segments", which are then related with various classes. The primary purpose of segmentation is to simplify how an image is depicted so that it can be more easily interpreted, evaluated, and understood. Segmentation of images is utilized in a wide range of machine learning functions; namely locating an object, detecting boundaries, medical image segmentation, and face recognition. Essentially, segmentation is carried out by labeling each and every pixel in an image on the basis of similar features, such as color, intensity, keypoints, or patterns [26]. The outcome of the segmentation of an image is indeed a set of segments [68].

For the purposes of this thesis, two types of segmentation will be discussed; namely salient object segmentation and medical image segmentation. Salient object segmentation involves detecting the most obvious and noticeable important object within an image. In understanding the image, the pixel is classified by a saliency map as being associated with or constituting either a salient object or the background. Figure 5.1 provides a good example of an image and the saliency map associated with it. This type of segmentation is particularly important, as it concentrates entirely on the image's most significant aspects. This can be applied in smart photo cropping and downsizing, in which the image is cropped automatically without any loss in the most salient image artifacts [69].

Another application of salient object detection is in User Interface (UI) development, where this process is employed to understand which UI components are particularly useful and/or convenient. This aids in the improvement of better user interfaces, which provides a smoother environment for the customers. By automating the method of identifying the essential parts in an image, a variety of different product pathways are assisted through salient object detection. [69].

Next, medical image segmentation will be discussed. Accurate medical image segmentation is an essential and mandatory step in a patient's diagnostic analysis and a crucial step in the planning of treatment [26]. The automatic detection of anatomical structures, tumors, etc. in medical images, whether 2D or 3D, is a key issue in medical image segmentation [26]. In this work, MRI medical images are used to detect a tumor in 3D

MRI scans. The goal is to propose a model in order to accurately detect and segment a tumor in an MRI scan, whether the images are 2D or 3D. Illustrations of 2D and 3D brain tumor image segmentation are shown in Figure 5.1. Deep learning approaches to the segmentation of medical images face many challenges that are not encountered in the detection of salient objects. Several examples of these challenges include the shortage of extensive training data sets, advanced preprocessing methods on different modalities of medical imaging, and storage limitations of medical images (3D). Additionally, as the tumor shapes are specific to every patient, it is impractical to use priors on tumor shapes [26] [70].

In the past few years, state-of-the-art performance on standard datasets with different image recognition tasks have been achieved by using deep convolutional neural networks [26]. These accomplishments are largely owed to the ever increasing computational capability of GPUs and the greater abundance of available training datasets [26] [71]. Various fully convolutional architectures have also benefited from image segmentation; particularly the U-Net architecture [23].

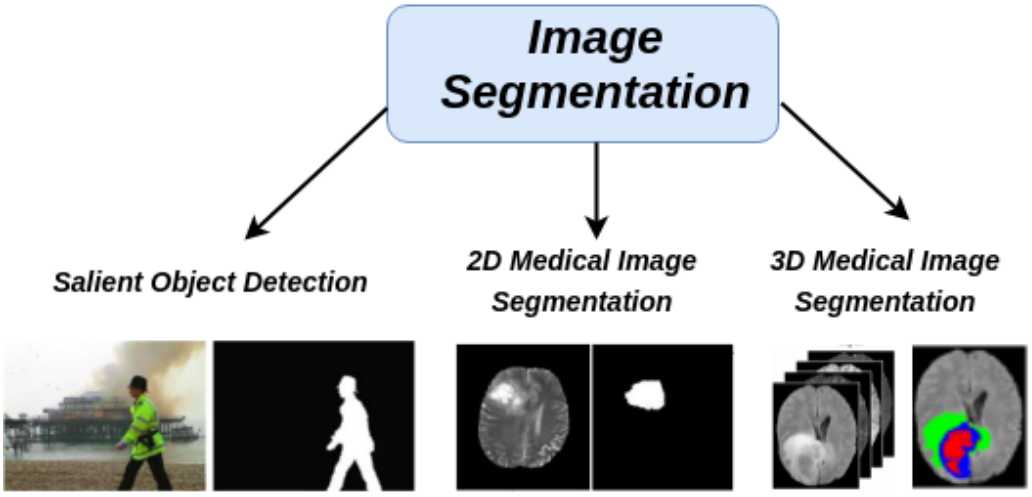

FIGURE 5.1: Image Segmentation - Salient Object Detection, 2D Medical Image Segmentation, 3D Medical Image Segmentation [26].

5.3 2D Medical Image Segmentation and 3D Medical Image Segmentation

Enhanced modern technologies, such as MRI, CT, etc. have improved the generation of high-quality medical images, which helps doctors to diagnose diseases accurately.

Generally, the first step in the diagnosis procedure is medical image segmentation. The referred paper [18] explains that an important step in treatment planning is accurate segmentation. Typically, images are segmented to target organs (liver, kidneys, etc.) or tumors within organs. Prior to the arrival of computer vision in the areas of medical image analysis, segmentation methods for detecting the boundaries of distinct organs and tumors were carried out manually by expert technicians. These methods are not only tedious for professionals, but are also susceptible to mistakes, as precise boundaries can sometimes be very difficult to trace manually [70].

Over the last few years, numerous efficient algorithms have been introduced to support the process of segmentation [26] [72]. However, finding extensive amounts of labeled training data in the medical field is rare. Thus, clustering methods (e.g. k-means) that were first applied to medical images in 1996 usually failed to be robust. Compounding this lack of abundant tumor segmentation data, each tumor is unique to a specific patient. For these reasons, it is hard to achieve a beforehand model for tumor segmentation [26] [73].

During the last few years, "the growth of deep learning techniques in computer vision has been remarkable" [26]. In 2015, the use of "fully convolutional networks for semantic segmentation" [74] was proposed by Long, Shelhamer, and Darrell. Deep learning (DL) is extensively used in the areas of medical image segmentation because of the learning and feature extraction capabilities it provides without requiring any manual work by experts [26]. In order to learn meaningful appearance details and generate pixel-wise prediction charts, convolutional and pooling layers are involved and utilized [26]. Such prediction tasks are done on arbitrary sized input images. Nonetheless, the predicted object boundaries regularly appear blurred because of resolution loss resulting from pooling.

To minimize the issue of blurred resolution, Badrinarayanan, Kendall, and Cipolla (2015) proposed Segnet, "A deep convolutional encoder-decoder architecture for image segmentation" [75]. The proposed architecture, called Encoder-Decoder, maps low-resolution encoder features to the initial input resolution. In 2015, Ronneberger, Fischer, and Brox (2015) presented "U-net: Convolutional networks for biomedical image segmentation" [23], in which the addition of skip connections at various resolutions during image processing was proposed [26]. Since then, promising results have been demonstrated with

several Deep Learning (DL) image segmentation architectures for 2D and 3D images [26] [23].

5.4 Related works in 2D U-Net and 3D U-Net

Among the best-known architectures for medical image segmentation is U-Net. The most significant characteristic of U-Net is its implementation of communicating directly between the layers in the encoder path and the decoder path, which provides the required high-resolution features to the decoder layers [76]. This unique architecture has attracted much attention in the areas of medical image processing. Several variations of the U-Net model have been developed [76], such as a network that was introduced for volumetric xenopus kidney segmentation that learns from sparsely annotated images. This 3D U-Net model provides dense 3D image segmentation from the learned sparse annotations [77], thus providing accurate segmentation results. It achieves an average Intersection over Union (IoU) of 0.863 (IoU is the overlap ratio between the ground truth and prediction and is a type of loss function.) for highly complicated xenopus kidney images [77].

Further illustrating the capabilities of U-Net, researchers used chest X-ray images for analysis of lung cancer. In this paper [78], a deep learning model for automatic detection of disease was demonstrated. This deep learning U-Net method efficiently segments lung images by excluding bone shadow in 2D chest X-ray images. To demonstrate the capability to effectively segment X-ray images by eliminating bone shadow, short running time and simplified configuration were prioritized over achieving the highest accuracy and lowest loss. The results obtained by the model show that U-net is capable of eliminating bone shadow to achieve fast and accurate image segmentation. This model was attempted on single CPU, multi-CPU, and GPU configurations to measure the model's performance running on different hardware. The results show a maximum speed of 0.3 times in multi-CPU mode and up to 9.5 times in GPU mode for the largest image size of 1024 x 1024 at 8 images per batch [78].

The 2D sematic segmentation problem for brain tumor detection is addressed using a deep hourglass U-Net model [79]. The adapted hourglass network (refer to [79] for the architecture) improves the U-Net model by using denser residual bottleneck blocks and

adding convolutions to the skip connections. The hourglass architecture is capable of classifying the Enhancing Tumor (ET), Tumor Core (TC), and Whole Tumor (WT) in a single pass [80]). This model achieves an overall dice coefficient (a loss function which calculates the similarities between the ground truth and the prediction [80]) of 92% on the training dataset. On the validation dataset, it achieves scores of 0.59, 0.82, and 0.63 for Enhancing Tumor (ET), Whole Tumor (WT), and Tumor Core (TC) respectively. Adding residual blocks depth-wise will improve the overall accuracy of the model [79]. However, in this mode (U-Net with residual blocks) the achieved scores with respect to identifying the tumor regions (0.59, 0.82, 0.63 [79]) is low when compared to other models using the validation dataset.

A two-step segmentation (as shown in Figure 5.2) model consisting of coarse segmentation and fine segmentation has been proposed based on a 3D U-Net model with dense blocks [27]. This model uses the advantages of both U-Net and dense blocks to efficiently segment the tumor region. Coarse segmentation uses an input size of 112 x 128 x 64 and fine segmentation uses an input size of 144 x 144 x 48 [27]. This model was trained on the keras framework with TensorFlow backend and four NVIDIA TitanX GPUs, taking four days to complete. This model has shown better results, producing dice coefficient scores of 0.75, 0.89, and 0.74 for Enhancing Tumor (ET), Whole Tumor (WT), and Tumor Core (TC) respectively [27]. U-Net with dense blocks has shown efficient results and improves segmentation tasks with feature reuse and vanishing gradients.

The 3D patch-based U-Net model [34] has shown improved results with the validation dataset. Patch-extracted 3D images are used as input in this model. 64 x 64 x 64 patches have been extracted to avoid the non-tumor regions. This model was trained with 50 epochs using an NVIDIA P100 GPU with 128 GB RAM. Training of the model took 48 hours to complete. This model achieved superior results with an average dice coefficient of 93% on the training dataset and an 87% dice score on the validation dataset. The dice scores are 0.75, 0.88, and 0.83 for Enhancing Tumor (ET), Whole Tumor (WT), and Tumor Core (TC) respectively [34]. As you can see, the accuracy of segmentation has improved significantly.

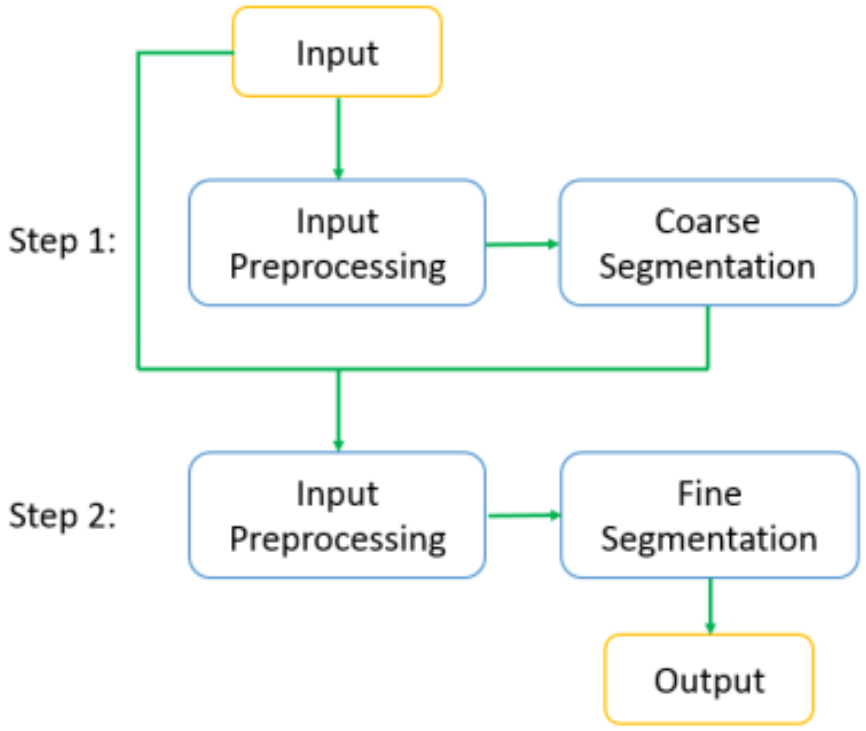

FIGURE 5.2: Two-Step Segmentation - Block Diagram [27].

5.5 Conclusion

Image segmentation have long been recognized as playing a major role in the analysis of medical images [26]. While originally performed manually by expert technicians, modern image segmentation methods take advantage of machine learning algorithms and architectures. Advancements in machine learning have allowed medical professionals to pinpoint tumors using image segmentation executed on a pixel by pixel basis. Machine learning-based image segmentation has further progressed in recent years, with newer models taking advantage of and further contributing to machine learning advancements. Among these are U-Net and DenseNet, the former of which is still widely considered to be the most effective medical image segmentation model.

Adding Dense blocks to the U-Net model yields measurable improvements over each model alone in the analysis of validation datasets. In other words, this enhanced model is effective not only at analyzing the data with which it was trained, but also at detecting tumors from new images yet "unseen" by the model. It does so both efficiently and accurately, making a compelling case for the use of a Dense U-Net in the detection of tumors from medical imaging. This will be discussed in the next chapter.

Chapter 6

Proposed Architecture and Pseudocode

6.1 Introduction

In this chapter, topics covered include why the Dense U-Net model has been selected for proposal and how such a model will be implemented as a means to segment and detect tumors from scanned images of the brain. Background and details on about the dataset used for the training and testing purpose will be provided, followed by an explanation of the overarching model being proposed. Finally, the case for proposing a Dense U-Net model will be discussed and each of its components will be covered in detail, complete with pseudocode to illustrate the method of implementation.

6.2 Input Dataset

This model is based on the BraTS 2018 (Multimodal Brain Tumor Segmentation Challenge) datasets. The main aim of BraTS is to evaluate state-of-the-art methods for brain tumor image segmentation using multimodal magnetic resonance imaging (MRI) scans. BraTS 2018 focuses on segmenting brain tumors, which are heterogeneous in shape and appearance [81], [82], [83], [84], [85].

6.2.1 Dataset Request

A BraTS 2018 dataset request is submitted by following the steps listed below [81], [82], [83], [84], [85]:

- Create an account in CBICA's Image Processing Portal (ipp.cbica.upenn.edu). Once the request has been approved, an Email will be sent to the registered mail-id.
- Then, log into the portal at ipp.cbica.upenn.edu
- Click on "BraTS'18 Data Request".
- Fill in the application with the required details and submit the job (click "Submit Job"). You will receive another Email as soon as the job is submitted.
- Now login to the IPP portal and access the zip file ("Results.zip"), in which you will find a text file ("REGISTRATION_STATUS.txt") that will provide the links to download the BraTS 2018 data.

6.2.2 BraTS 2018 Data

The BraTS 2018 dataset consists of a training dataset (MICCAI_BraTS_2018_Data_Training), a validation dataset (MICCAI_BraTS_2018_Data_Validation), and a testing dataset [81], [82], [83], [84], [85]. The training dataset is comprised of tumor cases of **HGG (High-Grade Glioma)** and **LGG (Low-Grade Glioma)**. There are 210 cases of HGG and 75 cases of LGG. The validation dataset is comprised of 66 cases covering a mix of HGG and LGG tumors. Each case consists of four MRI sequences; namely native (T1), gadolinium contrast-enhanced T1-weighted (T1Gd), T2-weighted (T2), and Fluid Attenuated Inversion Recovery (FLAIR).

All included datasets have been manually segmented by four raters. Annotations were marked following an annotation protocol that has been approved by experienced neuro-radiologists. Annotation labels include **Enhancing Tumor (ET – Label 4)**, **Edema (ED – Label 2)**, and **Necrotic and Non-Enhancing Tumor (NCR/NET — Label 1)**, as detailed in the BraTS reference paper [28]. The tumor region is predicted by merging all three regions: **Whole Tumor (WT-all three labels)**, **Tumor Core**

(TC- ET and NCR/NET), and **Enhancing Tumor (ET)**, as shown in Figure 6.1. The datasets provided by BraTS 2018 are preprocessed. Each MRI sequence is skull stripped, registered to a shared space and an isotropic resolution of $1mm^3$ [85] (transaxial plane X, Y and in longitudinal direction Z). The dimensions of the MRI sequences are 240 x 240 x 155 [85].

The labels in the provided data are as follows [85]:

- Label 1 – NCR/NET
- Label 2 - ED
- Label 4 – ET
- Label 0 – Everything else (Background)

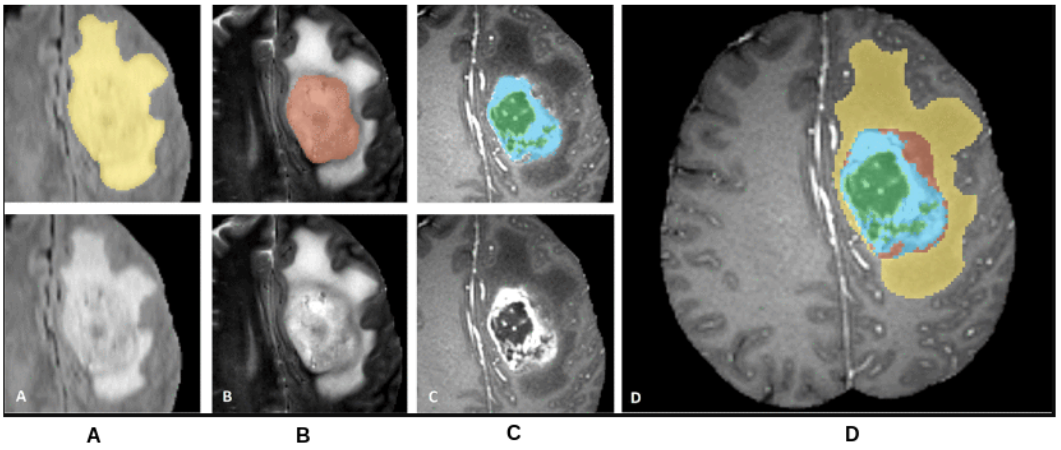

FIGURE 6.1: Tumor Sub-regions: From left to right, the image patches show A) The Whole Tumor (Yellow) visible in a FLAIR MRI sequence, B) The Tumor Core (Red) visible in T2, C) The Enhancing Tumor structure (Blue) visible in T1GD surrounding the necrotic components of the core (Green), and D) All three segmentations merged to generate the final layers of the tumor sub-regions (yellow, red, blue and green) [28].

6.3 System Requirements for the proposed Dense U-Net model

- NVIDIA Tensor core GPU, 32 to 128 GB RAM (Min)
- Between 32 – 48 cores Processor
- NVidia T4 GPU focused towards Tensor cores

- Linux OS
- 128 GB RAM
- 1TB SSD

6.3.1 Softwares Required for the proposed Dense U-Net model

- Anaconda (version from 4.3.0)
- Python (version 2.7 - 3.7)
- Keras framework (version from 2.0.5) with Tensorflow (version 2.0) backend

6.4 Proposed Model

The proposed model consists of two phases: the data preprocessing phase and the network architecture phase. The base model was inspired by work from three different papers [67], [34], [27]. The segmentation of brain tumors from MRI images has remained a challenging task over past decades. The heterogeneous appearance of brain tumors in MRI images greatly increases the difficulty of diagnosing them. This paper proposes a model to segment the brain tumors by merging two deep learning models; namely U-Net and DenseNet.

The advantages of using Dense U-Net include the use of dense skip connections that raise the number of convolutional layers to prevent overfitting, as well as improved performance. Using the BraTS 2018 dataset as the input parameter (original image size 240 x 240 x 155) in the block diagram (as shown in Fig:2), the tumor region will be segmented from the MRI images [85], [67], [34], [27].

6.4.1 Block Diagram

The block diagram (as shown in Figure 6.2) illustrates the process of 3D brain tumor image segmentation using Dense U-Net architecture. As discussed earlier, each MRI sequence in the BraTS 2018 dataset is 240 x 240 x 155 in dimension and is skull stripped. Old MRI machines produce a bias field signal, which is a smooth low-frequency signal

that corrupts the MRI images. Hence, bias field correction is performed on the MR data [10]. While acquiring the MRI images, different scanners could be used that may result in intensity variations. Therefore, normalization of intensity is a vital preprocessing step for brain MRI images [86]. Patch extraction is then performed, wherein 64 x 64 x 64 patches are extracted from the training data in a way that includes significant tumor areas and avoids non-tumor pixels. This must be done for all four MRI sequences. As a result, the 64 x 64 x 64 patches are given as input to the first layer of the Dense U-Net model, which is further discussed in this chapter [34].

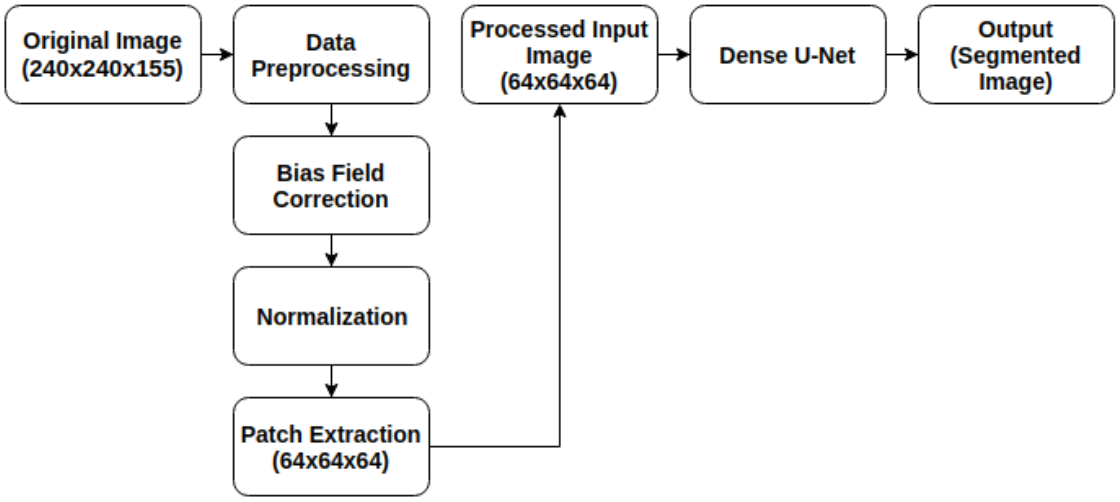

FIGURE 6.2: Process model - 3D brain tumor image segmentation using Dense U-Net.

6.4.2 Phase 1 - Data preprocessing

Phase 1 is the data preprocessing stage, in which the 3D MRI images are cleansed and made fit for the second phase. Bias field correction, normalization, and patch extraction (64 x 64 x 64) are carried out in the cleansing process. The remainder of this section will explain the preprocessing steps in detail [85].

6.4.2.1 Bias Field correction

A bias field signal is a smooth low-frequency signal that corrupts MR images due to the unevenness in the magnetic fields produced by MRI machines. This signal blurs the images and reduces the intensity values of the image pixels where the same tissue has different grey-level values across the same image. This variation is low-level and

will not have a major impact when diagnosed manually by radiologists. However, image processing algorithms performing segmentation and classification will show degraded performance and will not produce satisfactory results. Hence, this preprocessing step is required to rectify the effects of bias field signals and prevent corrupted MR images from being presented to segmentation or classification algorithms [87]. N4ITK is the most common bias field correction algorithm [88].

```
InputImage: MR images with the uneven magnetic field.
sitk.shrink: SimpleITK, reduces the size of the image
sys.argv: command line argument (in terms of list)
sitk.N4BiasFieldCorrectionImageFilter: implements SimpleITK N4 bias field correction
sitkFloat32: Pixel will be represented as 32 bit float value
    #inputImage = sitk.Shrink(inputImage,[int(sys.argv[3])]*inputImage.GetDimension())
MaskImage: defines the structure, binary mask.
    #maskImage = sitk.Shrink(maskImage,[int(sys.argv[3])]*inputImage.GetDimension())
    #inputImage = sitk.Cast(inputImage,sitk.sitkFloat32)
    #corrector = sitk.N4BiasFieldCorrectionImageFilter();
    #numberFilltingLevels = 4
Output: Processing result.
    #output = corrector.Execute(inputImage,maskImage)
```

LISTING 6.1: Pseudocode - N4ITK: Bias Field correction

6.4.2.2 Normalization

As the MRI images are acquired from different scanners with varying protocols, there may be intensity variations as a result. This is particularly true when using a single algorithm to segment brain tumors acquired from different MRI scanners. Therefore, normalization of modalities is required. Normalization ensures that the value ranges match between different patients. The modalities of each patient are normalized by independently subtracting the mean and dividing by the standard deviation (6.1) of the brain region [89]. By normalizing the data, the accuracy of the network will be increased and the required training time will be decreased [65]. A positive Z-score value indicates that the x value is greater than μ. A negative Z-score value indicates that the x value is less than μ. The normalization formula is as follows:

$$Z = \frac{(x - \mu)}{\sigma} \tag{6.1}$$

Where [65]:

- Z is Z-score normalization (Z-test)
- x is the current intensity,
- μ is the mean of the modality, and
- σ is the standard deviation of the modality.

6.4.2.3 Patch extraction

In comparison to the tumor area, the brain region will take up much more of an MRI image. As such, the training time may be significantly prolonged when the whole MR image is given as input. This can also lead to an increased number of steps and prolonged calculation of gradients at each step. Such class imbalance might also cause the model to focus mainly on the background. Therefore, to efficiently utilize the training data, patch extraction is required. Smaller patches of size 64 x 64 x 64 are extracted from all four MRI sequences. Patches are extracted in such a way that the significant tumor area is included and the non-tumor area is avoided. This patch extraction training takes approximately 48 hours to complete for the BraTS 2018 training dataset with 50 epochs using the TensorFlow framework on a NVIDIA P100 GPU with 128 GB RAM (as mentioned in [34]). The output will give four probability maps generated for Edema, ET, Necrosis, and the non-tumor region (background). The labels are assigned to the image with the highest probability amongst them all [34]. A sliding window operation is used to extract the patches. The unfold function is used to simplify the sliding window operation with patch stride 64. The sample code for Patch Extraction (64 x 64 x 64) using "Tensor.unfold" is given below [90].

```
S: channel dimension
    #S = 155
W: width of the image
    #W = 240
H: height of the image
    #H = 240
Batch Size: number of patches per image
    #batch\_size = 10
    #x = torch.randn(batch\_size, S, W, H)
size: patch size
    #size = 64
stride:  patch stride
    #stride = 64
```

```
Patch Formula
unfold: simplifies the "sliding window" operations
    #patches = x.unfold(1,size,stride).unfold(2,size,stride).unfold(3,size,stride)
    #print(patches.shape)
    #torch.Size([10, 2, 4, 4, 64, 64, 64])
patches now contains [2, 4, 4] patches of size [64, 64, 64].
```

LISTING 6.2: Pseudocode - Patch extraction (64 x 64 x 64)

6.4.3 Phase 2 - Network Architecture

In the field of medical image processing, brain tumor segmentation is a challenging task. The shapes of tumors are complex and irregular, differing from one patient to another [27]. The enhancements of deep learning algorithms, network structure, and computer hardware have rapidly made them become the preferred method for image segmentation and processing as a result. Convolutional neural networks in particular help to solve medical image processing problems, such as accurate detection, segmentation, and classification of tumors [27].

Convolutional Neural Networks (CNNs) have been precisely designed using a powerful family of neural networks. The field of computer vision is dominated by CNN-based network architectures, performing tasks that include recognizing an image, detecting an object, and semantic segmentation [91]. The combination of two CNN-based network approaches, namely U-Net and DenseNet, are presented in this paper as means to efficiently segment brain tumors [27].

6.4.3.1 Why Dense U-Net?

The combination of two CNN-based network approaches is inspired by the recently introduced DenseNet architecture for image classification. The core unit of a DenseNet architecture is the densely connected blocks, also referred to as DenseBlocks, where pooling layers are substituted by dilated convolutions [31]. In using these DenseBlocks, the output result from every layer is concatenated with the input of the forthcoming layer [32]. The DenseNet network model consists of a number of dense blocks separated by transition layers. Each transition layer contains a pooling operation and downsampling feature maps when combined with the U-Net architecture [31].

U-Net is a popular CNN model for object segmentation that learns segmentation in an end-to-end setting. The U-Net architecture is comprised of a downsampling path, which increases the "What" and reduces the "Where", followed by an upsampling path, which creates a high-resolution segmentation map [23] (as shown in Figure 6.3). Thus, merging the DenseNet and U-Net approaches will lead to an effective image segmentation algorithm.

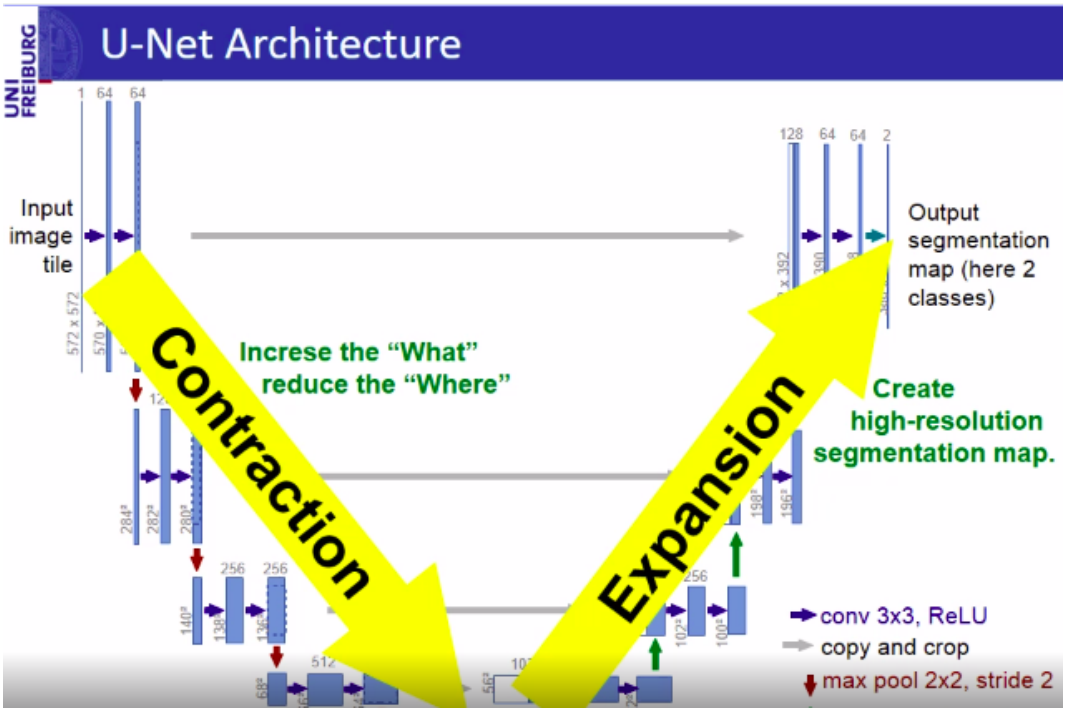

FIGURE 6.3: U-Net Architecture – Contraction path (increases the "What" and reduces the "Where") and Expansion path (creates high resolution segmentation map) [29].

6.4.3.2 Structure of a Dense Block

As mentioned previously, the DenseNet architecture consists of DenseBlocks. Each layer is densely connected to all the other layers in a feed-forward fashion [30] (as shown in Figure 6.4). The Feature-maps of all preceding layers are used as inputs to the upcoming layer. The output from the l^{th} layer connects to the $(l+1)^{th}$ layer in a traditional feed-forward neural network after the application of composite operations $H_l(.)$. This composite function $H_l(.)$ is comprised of a convolution operation, batch normalization, and an activation function ReLu (Rectified Linear Units). The equations for traditional feed-forward networks (6.2) and DenseNets (6.3) are shown below. DenseNet concatenates the output feature maps with the incoming feature maps [30].

Where $(x_0, x_1, ..., x_{l-1})$ refers to the concatenation of the feature maps generated from the layers $(0, 1, ...l-1)$ [67].

$$x_l = H_l(x_{l-1}) \tag{6.2}$$

$$x_l = H_l([x_0, x_1, ..., x_{l-1}]) \tag{6.3}$$

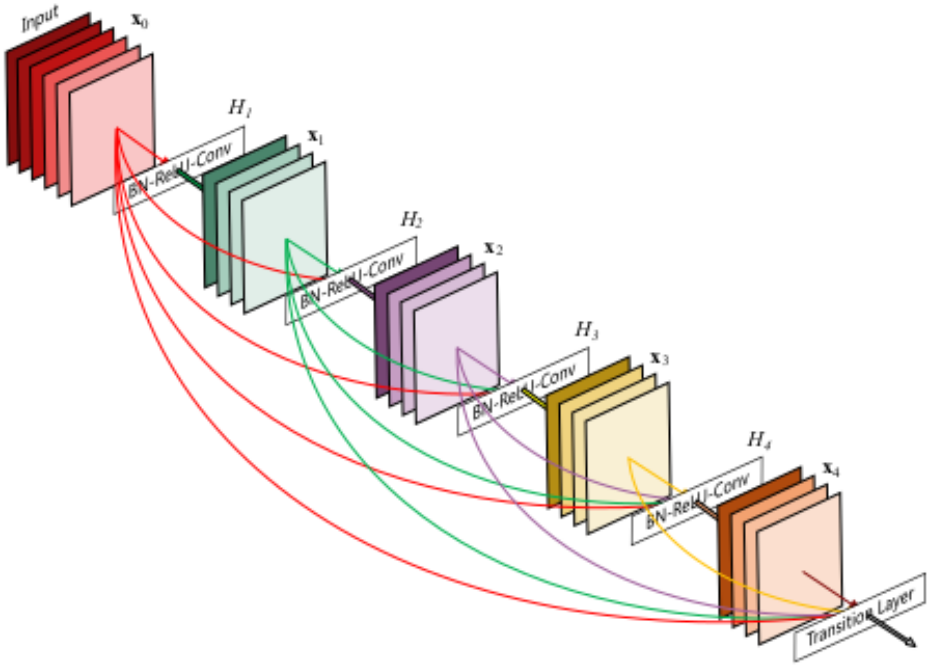

FIGURE 6.4: Structure of a Dense Block [30].

A DenseBlock is a bunch of layers connected to all its previous layers. A single DenseBlock layer consists of the following [92] (as shown in Figure 6.5):

- Batch Normalization
- Convolution
- ReLU activation

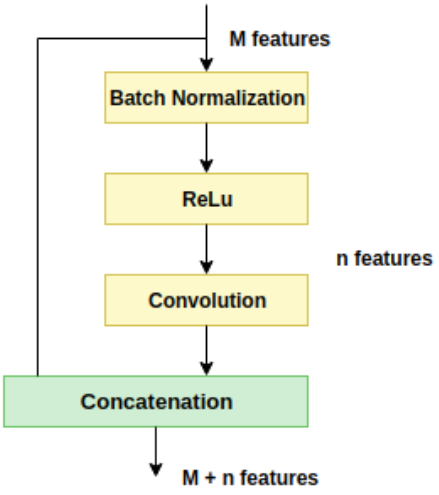

FIGURE 6.5: A Dense Block Layer [31].

6.5 Proposed Dense U-Net Architecture

Recent research has shown that a convolution network can be trained more deeply, effectively, and accurately if the network contains a shorter connection between the input layer and the output layer. By using a different connectivity pattern, DenseNet takes advantage of this observation. Other advantages of DenseNet include a reduction of the vanishing-gradient problem, an increase in feature propagation, support for feature reuse, and a significantly reduced number of parameters [27]. DenseNet has proven to be effective in the task of image segmentation. U-Net is widely used in medical image segmentation because of its encoder-decoder model, wherein the encoder progressively expands the field of view and the decoder retrieves the object details. In this architecture, DenseBlocks are inserted into the U-Net model, which allows the network to be much deeper and more efficient [27].

The Dense U-Net architecture also has two paths: the downsampling path on the left and the upsampling path on the right, similar to the traditional U-Net architecture (as shown in Figure 6.6). This network architecture consists of 4 encoder levels, followed by a base level and 4 decoder levels. Each encoder level in the encoder path is comprised of a DenseBlock. Every DenseBlock layer uses the feature-maps from all the preceding layers as input (as shown in Figure 6.4). Its own feature-maps are used as inputs for all the other subsequent layers. Each dense block is composed of repeated layers of Batch Normalization (BN), an Activation Function (AF) ReLu (Rectified Linear Unit), convolution, and concatenation operations (as shown in Figure 6.4, Figure 6.5). Zero

padding equalizes the input and output dimensions when used in every layer. Each decoder level in the decoder path consists of an upsampling layer, followed by two convolutional layers. Upsampling is also known as a transition block that uses simple resizing. As with U-Net, depth-wise concatenation of the upsampled layer with a feature map from the last layer of every pooling step is performed [27].

The major advantage of using a DenseBlock is that it has direct access to all of the layers, which helps in absolute deep supervision and potentially reuses the feature maps. Concatenating the feature maps from all the layers will improve the efficiency of the model [67]. Training a DenseNet model may lead to memory inefficiency and requires powerful tensor Graphical Processing Units (GPUs). Significant reduction in memory consumption using a shared memory buffer may increase the computation time, as stated in "Memory-Efficient Implementation of DenseNets" [93] by Geoff Pleiss, Danlue Chen, Gao Huang, and Tongcheng Li in 2017 [93].

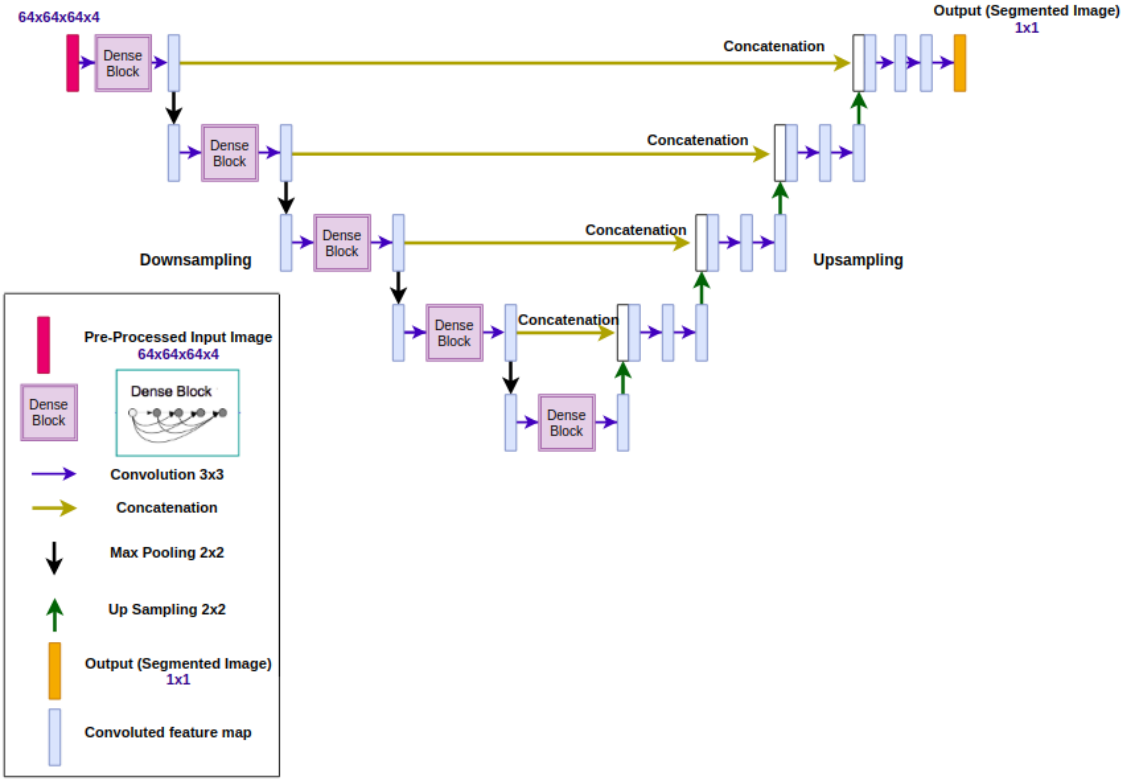

FIGURE 6.6: Dense U-Net Architecture – As this is a proposed architecture, the output may differ with respect to the given input [27].

- **Input**

  The input to the Dense U-Net model is all four 3D MRI sequences of patch size **64 x 64 x 64 x 4**. The reason behind using all four modalities is that different tumor

regions might be visible in different MRI sequences. Use of all four modalities allows the model to more accurately identify the tumor region [34].

- **Convolution Layer**

  The convolutional layers are the building blocks of this architecture. The input data is transformed by this layer using a local patch connecting the neurons of the prior layers (as shown in Figure 6.7). The dot product is calculated using the input layers between the neurons region and the output layer with weights, which are locally connected [32]. The output generated usually has the same or smaller spatial dimensions. The fundamental concept of these layers is called a convolution [32].

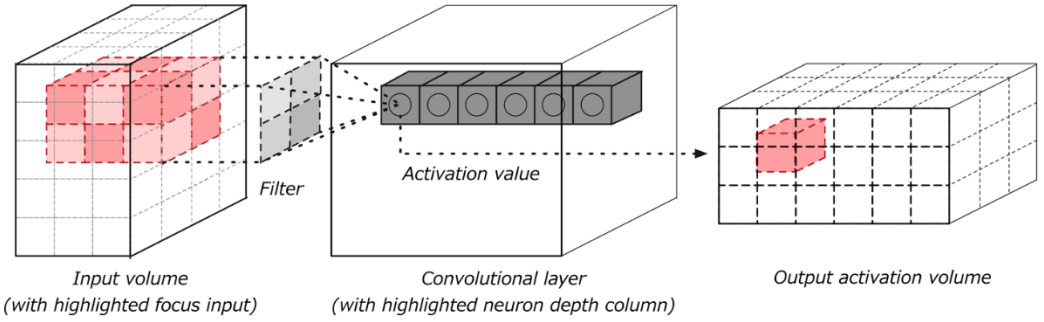

FIGURE 6.7: Convolutional layer with input volume and output activation volume [32].

- **Convolution Operation**

  The convolutional operation is a mathematical process generally defined as how two types of data can be combined [32]. The convolution takes an input, to which a convolutional kernel is applied, thus providing an output feature map (as shown in Figure 6.8). The convolution operation is also called a feature detector. The input provided to a convolutional layer could be a feature map outputted by the prior layer or raw data. It is always seen as a filter performed on the input data for some kind of information [32]. Figure 6.8 demonstrates the way the kernel moves throughout the input data to generate the convoluted output feature. In each and every move, the input data is multiplied with the kernel values, thus generating a single value within the convoluted feature map (output). In reality, the output value will be huge in the event that the feature that you are looking for within the input is detected [32].

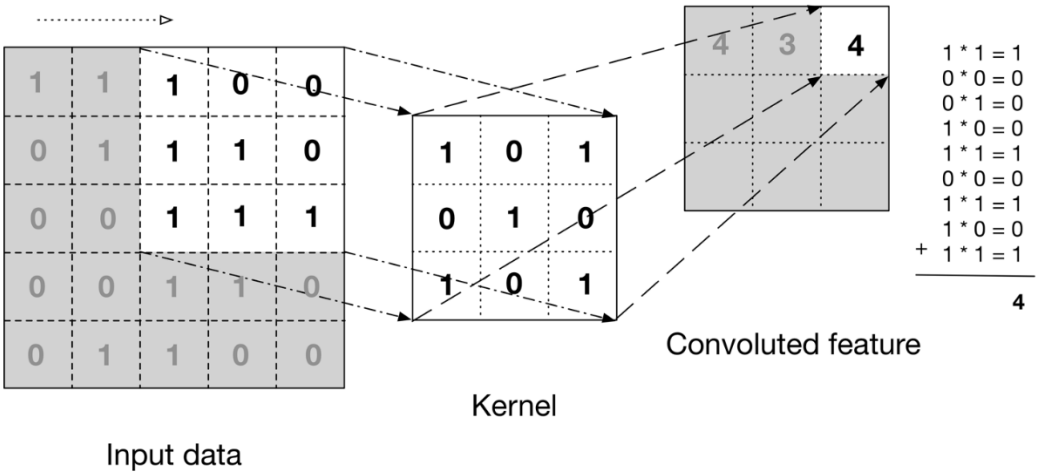

FIGURE 6.8: The convolution operation [32].

- **Kernel**

  The kernel, also called a filter, is used to generate a feature map (also known as activation map) by performing the convolutional operation on the kernel and the input data [32].

- **Stride**

  Stride determines how far the sliding window wants to move in the filter function. Depth columns are created in the output data by the stride functionality. If the stride value is lower, then there will be higher number of depth columns within the output data [32]. This leads to overlapping fields between the columns, which yields a larger output volume. If the stride value is higher, then overlap will be lessened and the output volumes will be smaller [32].

- **ReLu Functions**

  ReLu activation functions are often used with CNNs. The element-wise activation function is applied on the input data by the ReLu layer, which gives the same dimension of output data as the input layer [32].

- **Filter Size**

  Each and every filter is spatially small in regards to the filter size height and width. For instance, if the filter size is 3 x 3 x 3, then the filter is three pixels tall by three pixels wide. The final 3 represents the three RGB (Red, Green, Blue) color channels [32].

- **Zero-Padding**

  To manage the output volume's spatial dimension, Zero-padding is used [32].

- **Batch Normalization (BN)**

  Batch normalization speeds up the learning process (training the model) by including normalization within the network architecture. This can achieve significantly greater learning speeds by enforcing normalization for every set of training input data [32].

- **Pooling**

  To reduce the variance and computational complexity and extract the required features, pooling layers are used. There are two types of pooling operations; namely average-pooling and max-pooling. In the beginning, it was a common practice to use average-pooling, where the average of all the input values was considered in a feature map. However, in recent models, max-pooling layers are used to select the maximum value in the receptive field. Consider an example with input image size of 4 x 4, filter of 2 x 2, and a stride of two. Max-pooling outputs the maximum value from each 2 x 2 region. Average-pooling instead outputs the average rounded integer value from each 2 x 2 region [32].

  In this model, max-pooling operations are used. This form of pooling extracts essential features, such as edges, and is the better pooling operation for pulling out the features that are required (as shown in Figure 6.9). Average-pooling, on the other hand, will extract features based on the average. This will smooth the image, which in some cases cannot extract the required features [94].

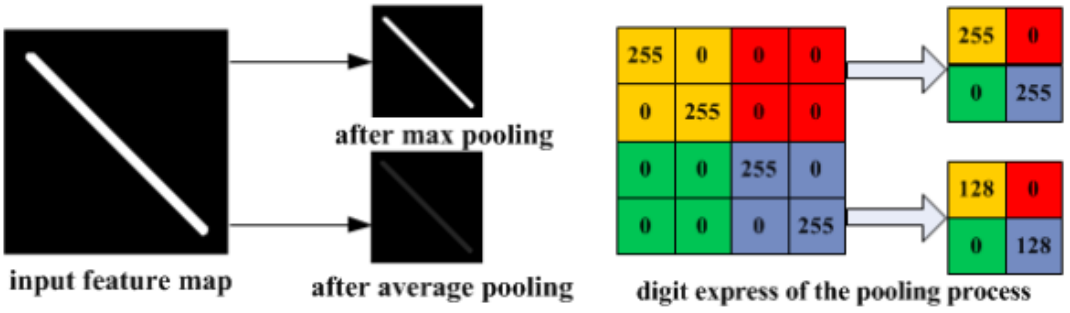

FIGURE 6.9: Max pooling vs Average pooling - with respect to the input feature map [33].

- **Dropout Rate**

  Dropout brings improved performance to the network model; especially in the

case of supervised learning [95]. As this model is a supervised learning model, dropout is used to address the overfitting problem. The fundamental concept is to drop units and their links randomly out from the network model during the training process. To expand on this, all incoming and outgoing connections for a selected unit are temporarily removed from the neural network during dropout. In this model, the dropout rate is set to 0.5 (probabilistic), which is optimal for a wide range of networks. If the dropout rate is set to 1, there will be no dropout. Lowering the dropout rate value results in increased dropout. A larger dropout rate will decrease dropout to the point that it will not be effective at preventing overfitting [95].

The different layers described above are important in this network architecture. The pseudo code will be explained further in the sections below.

6.6 Pseudocode for the proposed Dense U-Net model

6.6.1 Required Modules

- from keras.models import Model
- from keras.layers import Input, merge, ZeroPadding2D
- from keras.layers.core import Dense, Dropout, Activation
- from keras.layers.convolutional import Convolution3D
- from keras.layers.pooling import MaxPooling2D
- from keras.layers.normalization
- import keras.backend
- import numpy
- import tensorflow [96]

There are many other modules that can be imported according to the architecture requirements.

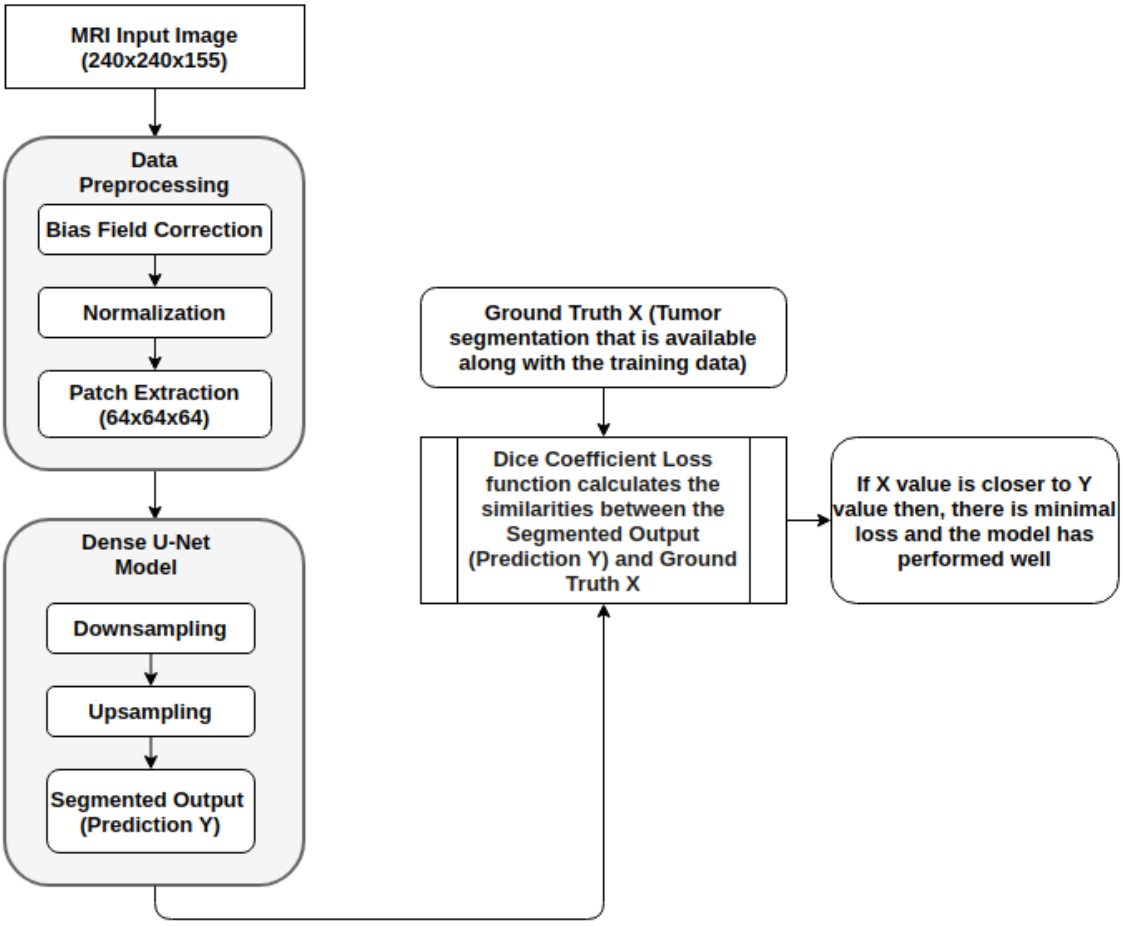

FIGURE 6.10: Flowchart for the Proposed Dense U-Net Model

6.6.2 To install the modules

- pip install "modulename" [96]

6.6.3 To define the DenseNet

Arguments

- dense_block: number of dense blocks
- growth_rate: to add the number of filters per dense block
- filter: number of the initial filter
- reduction: transition block reduction factor
- dropout_rate: dropout rate

- weight_decay: weight decay factor
- weights_path: pre-trained weight path [96]

This is a function in the Dense U-Net model.

```
#def DenseNet(dense_block=5, growth_rate=32, filter=64, reduction=0.0,
dropout_rate=0.5, weight_decay, weights_path=None)
```

LISTING 6.3: Defining DenseNet

6.6.4 To define one layer in the Dense U-Net model (Downsampling Layer)

Figure 6.11 is a single Dense U-Net layer [96]:

FIGURE 6.11: Single Dense U-Net layer.

```
#inputs = Input ((depth of the image, rows of the image, columns of the image,
no of input images))
#convolution_layer1 = Conv3D (32(number of filters or Kernel), (3, 3, 3)
(size of the filter), activation='relu', padding='same')(inputs)
#Dense_Block1 = concatenate ([inputs, convolution_layer1], axis=4)
#convolution_layer2 = Conv3D (32(number of filters or Kernel), (3, 3, 3)
(size of the filter), activation='relu', padding='same') (Dense_Block1)
#pooling = MaxPooling3D(pool_size=(2, 2, 2))(convolution_layer2 )
```

LISTING 6.4: Defining one layer in the Dense U-Net model

6.6.5 To define the Upsampling layer

```
#upsampling6 = concatenate ([Conv3DTranspose (256 (number of filters or Kernel),
(2, 2, 2) (size of the filter), strides= (2, 2, 2), padding='same')
(Dense_Block5), Dense_Block4], axis=4)
#convolution_layer11 = Conv3D (256 (number of filters or Kernel), (3, 3, 3)
(size of the filter), activation='relu', padding='same') (upsampling6)
#convolution_layer12 = Conv3D (256 (number of filters or Kernel), (3, 3, 3)
(size of the filter), activation='relu', padding='same') (convolution_layer11)
```

LISTING 6.5: Defining the Upsampling layer

6.6.6 To define the output layer

```
#convolution_layer19 = Conv3D (1, (1, 1, 1) (size of the output/segmentation))
(convolution_layer18)
#model = Model(inputs=[inputs], outputs=[convolution_layer19])
```

LISTING 6.6: Defining the output layer

Once the model definition is completed, the model must be trained and used to predict the segmentation output [96].

6.6.7 General Loss function formula

A major obstacle in brain tumor image segmentation is that the tumor region is extremely small when compared to the full MRI image. This leads to class imbalance problems. Thus, the loss function must be calculated to compare the predicted segment with the ground truth (6.4). The generalized dice coefficient formula is as follows [80]:

$$DiceCoefficient = 2 * |X \cap Y|/(|X| + |Y|) \tag{6.4}$$

Where:

- X is the ground truth (set of elements; ground truth images are available with the training data)
- Y is the prediction (the segmented output from the Dense U-Net mode)
- $\cap$ is the intersection (To find the similarities between the ground truth X and the predicted output Y)
- $|X|$ is the cardinality of X (number of elements), and
- $|Y|$ is the cardinality of Y (number of elements)

The coefficient calculates the similarities between the ground truth and the prediction to see if they contain the same elements. The coefficient value will be 1.0 when there are no common elements between X and Y, between 1.0 and 0.0 when X and Y have some elements in common, and 0.0 when all the elements are common between X and Y [80]. Pseudocode for dice coefficient [96]:

```
def dice_coef(X, Y):
    # X_f = K.flatten(X)
    # Y_f = K.flatten(Y)
    # intersection = K.sum(X_f * Y_f)
    # return (2. * intersection + smooth) / (K.sum(X_f) + K.sum(Y_f) + smooth)
def dice_coef_loss(X, Y):
    # return -dice_coef(X, Y)
In order to maximize the dice, it is required to minimize the negative dice loss.
```

Listing 6.7: Defining dice coefficient

6.6.8 Sample Output

A sample output from the segmentation algorithm is shown in Figure 6.12.

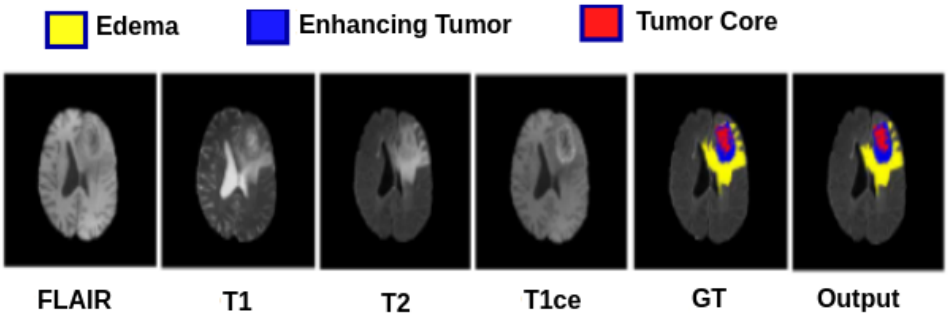

Figure 6.12: Sample output of the segmentation algorithm [34].

6.7 Conclusion

In this chapter, a detailed explanation of the proposed Dense U-Net model, pseudocode, and implementation steps have been provided. The described Dense U-Net model is a novel deep learning approach for brain tumor segmentation. Dense U-Net models have multiple advantages with respect to the image segmentation process, such as access of feature maps from the first layers through the last layer and high efficiency resulting from

feature reuse. As this is a pseudo model, all of the layers and functionalities of each block have been explained and illustrated using pseudocode. The segmented output shown in the previous section (Figure 6.12) is the expected output for the proposed model. Segmentation results may differ according to the number of input channels and the number of dense blocks used. Future improvements will be discussed in the next chapter.

Chapter 7

Conclusion

The brain is a vital organ in the human body. Any injury can be dangerous, if not fatal. Brain tumors are caused by abnormal cell growth in the brain and can exhibit cancerous behaviors. These tumors can be identified using imaging techniques, such as MRI and CT scanning. However, medical image processing and diagnosis are tedious and time-consuming tasks. This research was carried out to identify an effective algorithm that will accurately classify and segment a brain tumor from 3D MRI sequences using Dense U-Net, a deep convolutional neural network architecture. Developments in the field of Artificial Intelligence (AI), Machine Learning (ML), and Deep Learning (DL) can be combined to become an effective and efficient tool in medical image processing [13]. DL allows the combination of AI and ML to aggregate data, which helps to determine the precise location of the tumor.

There are a number of kinds of MRI sequences that utilize differing contrast enhancements, such that different tumor regions might be visible in different sequences. The four main imaging modalities are T1, T1CE, T2-weighted, and FLAIR. These contrast enhancements help radiologists in the process of diagnosing a tumor region. Diagnosis and classification of brain tumors to determine whether they are cancerous, and therefore dangerous to the patient requires the knowledge and experience of expert radiologists and physicians.

Producing accurate data on the presence and locations of tumors is critical to beginning patient treatment. Hyperthermia (HT) is a method of treatment that increases

the temperature of the tumor tissue to 39–44 C, significantly enhancing the effectiveness of radiotherapy and chemotherapy. Personalised Hyperthermia treatment planning is becoming a powerful tool to improve treatment quality, aided by enhanced growth in computational techniques and computing power. This treatment planning requires patient-specific data, in particular accurate data detailing the tumor region's location. Utilizing this data, tumor cells can be destroyed by increasing their temperature using thermal redistribution of energy, essentially killing them.

Accurate detection of the tumor region is a challenging, but necessary task that can be achieved through image segmentation. Without data detailing the tumor region, Hyperthermia (HT) cannot be performed. 3D MRI brain tumor image segmentation is the process of dividing a digital 3D image into multiple smaller portions, or segments. The key idea is to find out the tumor region in the MRI brain image scan and where exactly it is located. Digital images of abnormal tissue regions are segmented so that they can be accurately diagnosed.

Many DL algorithms and techniques, such as Convolutional Neural Networks (CNNs), U-Net, ResNet and DenseNet, are used for computer vision-related tasks [13]. Examples of such tasks include object detection, image segmentation, and classification. As more and more techniques are developed, processing and prediction becomes easier. However, research on other models, image processing techniques, and deep learning methodologies has shown that there is still room for improvement. A new model has been proposed that uses deep convolutional neural networks based on the U-Net approach for 3D image segmentation, merging DenseNet and U-Net into a single architecture: Dense U-Net.

Each of these techniques on their own are widely used in medical image processing. U-Net was specifically designed for this purpose, demonstrating high performance and efficiency through its pixel-based segmentation methods. DenseNet has also proven to perform well despite the demands of handling complex data sets with high levels of accuracy, thanks to its use of DenseBlocks. The combined Dense U-Net takes advantage of the benefits of both these architectures, allowing for feature reuse, concatenation of input and output layers, as well as downsampling and upsampling paths in order to segment medical images and accurately detect the locations of brain tumors [66] [24].

In addition to exploring how deep learning, and particularly DenseNet and U-Net, can advance our abilities to accurately detect tumors via image segmentation, we have laid

out our plan to propose such a model. Each component of the proposed Dense U-Net model has been described in detail, down to the structure of the DenseBlocks that make up its architecture. Furthermore, the pseudocode detailing how to build the Dense U-Net model has been provided, along with an ideal dataset for training and validation.

In this thesis, a new Dense U-Net model has been proposed which will further improve on the efficiency and accuracy of those currently in use. The expectation is that this model will increase the effectiveness of image segmentation at detecting the locations of brain tumors. Thus, these improvements will drive further research and enable medical professionals to provide better diagnosis and care to their patients.

Bibliography

[1] Humanbrainfacts.org. *Human Brain Anatomy - Components of Human Brain with Images*. (Date accessed: 02.14.2020), 2019. URL https://www.humanbrainfacts.org/human-brain-anatomy.php.

[2] "Mayfield certified health info materials are written and developed by the Mayfield clinic". *Mayfield Brain Spine - Anatomy of Brain*. Mayfield Clinc, (Date accessed: 02.14.2020), 1998-2018. URL https://d3djccaurgtij4.cloudfront.net/pe-anatomybrain.pdf.

[3] Making Headway Center The Human Brain. *The Amazing Human Brain*. 2016. URL https://mhwcenter.org/visualize-the-human-brain/. Date accessed: 02.14.2020.

[4] "American Brain Tumor Association Resources" (ABTA Resources). *American Brain Tumor Association - Brain Tumors - A Handbook for the newly Diagnosed (Pages: 14-19)*. American Brain Tumor Association Resources (ABTA Resources www.abta.org, (Date accessed: 02.14.2020). URL https://www.abta.org/wp-content/uploads/2018/03/newly-diagnosed-1.pdf.

[5] Cancer Support Community's. Frankly speaking about cancer series - brain tumors (pages: iii, iv, 5-12), 2013. URL http://blog.braintumor.org/files/public-docs/frankly-speaking-about-cancer-brain-tumors.pdf. 2013 Cancer Support Community and the National Brain Tumor Society www.cancersupportcommunity.org.

[6] Kamer T. Moroni M. Maric N. Tepest R. Dani I. Honer W.G. Scherk H. Rietschel M. Schulze T.G. Schneider-Axmann, T. and D.J. Müller. Relation between cerebrospinal fluid, gray matter and white matter changes in families with schizophrenia. *Journal of psychiatric research, 40(7), pp.646-655.*, 2006.

[7] MD David C Preston. *Magnetic Resonance Imaging (MRI) of the Brain and Spine: Basics.* (Date accessed: 02.04.2020), 2006. URL https://casemed.case.edu/clerkships/neurology/Web%20Neurorad/MRI%20Basics.htm.

[8] Matthew Mayo. *The Essence of Machine Learning.* KDnuggets, (Date accessed: 02.04.2020), Dec 2018. URL https://www.kdnuggets.com/2018/12/essence-machine-learning.html.

[9] Mohammed Terry-Jack. *Deep Learning: Background Research.* Medium, (Date accessed: 02.05.2020), May 2019. URL https://medium.com/@b.terryjack/deep-learning-background-research-64578f0d551d.

[10] Hunter Heidenreich. What are the types of machine learning?, Dec 2018. https://towardsdatascience.com/what-are-the-types-of-machine-learning-e2b9e5d1756f, (Date accessed: 02.05.2020), Medium, Towards Data Science.

[11] Divyansh Dwivedi. Machine learning for beginners, May 2018. https://towardsdatascience.com/machine-learning-for-beginners-d247a9420dab, (Date accessed: 02.05.2020), Medium.

[12] Gowthamy Vaseekaran. Machine learning: Supervised learning vs unsupervised learning, Sep 2018, Medium, (Date accessed: 02.05.2020). https://medium.com/@gowthamy/machine-learning-supervised-learning-vs-unsupervised-learning-f1658e12a780.

[13] Taha T.M. Yakopcic C. Westberg S. Sidike P. Nasrin M.S. Hasan M. Van Essen B.C. Awwal A.A. Alom, M.Z. and V.K. Asari. "a state-of-the-art survey on deep learning theory and architectures". *Electronics, 8(3), p.292*, 2019.

[14] Arunava. *The Perceptron.* (Date accessed: 02.14.2020), July 2018. https://towardsdatascience.com/the-perceptron-3af34c84838c.

[15] Michael Nielsen. *Neural Networks and Deep Learning.* (Date accessed: 02.14.2020), Dec 2019. URL http://neuralnetworksanddeeplearning.com/.

[16] SuperDataScience Team. *The Ultimate Guide to Convolutional Neural Networks (CNN).* SuperDataScience Team, (Date accessed: 02.14.2020), Aug 2018.

https://www.superdatascience.com/blogs/the-ultimate-guide-to-convolutional-neural-networks-cnn.

[17] Amir Sadeghian (Head CA) Silvio Savarese (Associate Professor). *Introduction to Convolutional Neural Networks.* Computer Science Department, Stanford University, (Date accessed: 02.14.2020), Feb 2018. URL https://web.stanford.edu/class/cs231a/lectures/intro_cnn.pdf.

[18] Soham Chatterjee. Different kinds of convolutional filters., Dec 2017. https://www.saama.com/blog/different-kinds-convolutional-filters/, (Date accessed: 02.13.2020).

[19] Gasper Vozel karantan. Activation functions., Sep 2017. https://github.com/karantan/dl-notes/blob/master/activation_functions.md, (Date accessed: 02.14.2020).

[20] Chris Rowen and Rishi Kumar. Neural networks for image recognition., Sep 2016. https://www.elektroniknet.de/elektronik-automotive/assistenzsysteme/lernen-statt-programmieren-130142-Seite-3.html,(Date_accessed: 02.14.2020).

[21] Harsha Bommana. Loss functions explained., Sep 2019. https://medium.com/deep-learning-demystified/loss-functions-explained-3098e8ff2b27, (Date accessed: 02.14.2020).

[22] Amar Budhiraja. Dropout in (deep) machine learning., Dec 2016. https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5, (Date accessed: 02.14.2020).

[23] Brox T. Ronneberger O., Fischer P. U-net: Convolutional networks for biomedical image segmentation. *In: Navab N., Hornegger J., Wells W., Frangi A. (eds) "Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015". Lecture Notes in Computer Science, vol 9351.*, November 2015.

[24] Machine learning engineer. Jeremy Jordan. *An overview of semantic image segmentation.* (Date accessed: 02.14.2020), May 2018. URL https://www.jeremyjordan.me/semantic-segmentation/.

[25] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning*. 2020. URL https://d2l.ai. chapter 6.

[26] Debleena Sengupta. Deep learning architectures for automated image segmentation, 2019.
http://web.cs.ucla.edu/~ahatamiz/thesis_arch.pdf.

[27] Weijian Jian 2 Xiaoyue Zhang 1 and Kun Cheng 3. 3d dense u-nets for brain tumor segmentation. *MICCAI_BraTS_2018_proceedings_shortPapers, Pre-Conference Proceedings of the 7th MICCAI BraTS Challenge (2018)*, pages 562–568. QED Technologies. Co., Ltd, Beijing, China.

[28] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, L. Lanczi, E. Gerstner, M. Weber, T. Arbel, B. B. Avants, N. Ayache, P. Buendia, D. L. Collins, N. Cordier, J. J. Corso, A. Criminisi, T. Das, H. Delingette, Ç. Demiralp, C. R. Durst, M. Dojat, S. Doyle, J. Festa, F. Forbes, E. Geremia, B. Glocker, P. Golland, X. Guo, A. Hamamci, K. M. Iftekharuddin, R. Jena, N. M. John, E. Konukoglu, D. Lashkari, J. A. Mariz, R. Meier, S. Pereira, D. Precup, S. J. Price, T. R. Raviv, S. M. S. Reza, M. Ryan, D. Sarikaya, L. Schwartz, H. Shin, J. Shotton, C. A. Silva, N. Sousa, N. K. Subbanna, G. Szekely, T. J. Taylor, O. M. Thomas, N. J. Tustison, G. Unal, F. Vasseur, M. Wintermark, D. H. Ye, L. Zhao, B. Zhao, D. Zikic, M. Prastawa, M. Reyes, and K. Van Leemput. The multimodal brain tumor image segmentation benchmark (brats). *IEEE Transactions on Medical Imaging*, 34(10):1993–2024, Oct 2015. ISSN 1558-254X. doi: 10.1109/TMI.2014.2377694.

[29] Thomas Brox Olaf Ronneberger, Philipp Fischer. U-net: Convolutional networks for biomedical image segmentation. 5 minute teaser presentation of the u-net. 2015. URL https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/. University of Freiburg, Online; (Date accessed: 02.12.2020).

[30] DASLab. Harvard University. Pablo Ruiz, Deep Learning Researcher. Densenets, August 2018, Date accessed: 02.12.2020. URL http://www.pabloruizruiz10.com/resources/CNNs/DenseNets.pdf.

[31] Richard McKinley, Raphael Meier, and Roland Wiest. Ensembles of densely-connected cnns with label-uncertainty for brain tumor segmentation. *In Pre-conference proceedings of the 7th medical image computing and computer-assisted interventions (MICCAI) BraTS Challenge (pp. 322-328)*, 2018.

[32] Josh Patterson Adam Gibson. Chapter 4. major architectures of deep networks, 2020, O'Reilly Media, Inc, Date accessed: 02.12.2020. URL https://www.oreilly.com/library/view/deep-learning/9781491924570/ch04.html.

[33] Peiqiu Chen Dingjun Yu, Hanli Wang and Zhihua Wei. Mixed pooling for convolutional neural networks. page pp. 364–375. Springer, Cham., 2014. doi: 10.1007/978-3-319-11740-9_34.

[34] Swapnil Rane & Siddhesh Thakur & Aliasgar Moiyadi & Meenakshi Thakur & Ujjwal Baid & Abhishek Mahajan, Sanjay Talbar and Sudeep Gupta. Gbm segmentation with 3d u-net and survival prediction with radiomics. *MICCAI_BraTS_2018_proceedings_shortPapers, Pre-Conference Proceedings of the 7th MICCAI BraTS Challenge (2018)*, pages 28–32, Shri Guru Gobind Singhji Institute of Engineering and Technology, Nanded, India, Tata Memorial Centre, Mumbai, India.

[35] National Brain Tumor Society. *Brain Tumors By The Numbers.* Jan 2019, (Date accessed: 02.21.2020). URL https://events.braintumor.org/wp-content/uploads/2019/02/BrainTumorsBytheNumbers_Jan_2019.pdf.

[36] MD Matthew Hoffman. *Picture of the Brain - Human Anatomy.* WebMD, LLC., 2014, Date accessed: 02.14.2020. URL https://www.webmd.com/brain/picture-of-the-brain#1.

[37] InformedHealth.org (Internet). *How does the brain work?* InformedHealth.org [Internet] - Cologne, Germany: IQWiG (Institute for Quality and Efficiency in Health Care) Last Update: October 31, 2018, Date accessed: 02.14.2020. URL https://www.ncbi.nlm.nih.gov/books/NBK279302/.

[38] Abdelouahab Moussaoui Messaoud Hameurlaine. *Survey of Brain Tumor Segmentation Techniques on Magnetic Resonance Imaging.* Nano Biomed. Eng., 2019, 11(2): 178-191., 2019. doi: 10.5101/nbe.v11i2.p178-191.

[39] RadiologyInfo Web site (http://www.radiologyinfo.org). *RadiologyInfo.org for Patients - Brain Tumors*. 2019 Radiological Society of North America, Inc., 2019. URL https://www.radiologyinfo.org/en/pdf/braintumor.pdf.

[40] By Mayo Clinic Staff. *Brain Tumor - Symptoms and Causes*. 1998-2019 Mayo Foundation for Medical Education and Research (MFMER)., 1998-2019, (Date accessed: 02.14.2020). URL https://www.mayoclinic.org/diseases-conditions/brain-tumor/symptoms-causes/syc-20350084.

[41] Amit Mehndiratta and Frederik L Giesel. Brain tumour imaging. In Ana Lucia Abujamra, editor, *Diagnostic Techniques and Surgical Management of Brain Tumors*, chapter 2. IntechOpen, Rijeka, 2011. doi: 10.5772/23507. URL https://doi.org/10.5772/23507.

[42] FACP FACR William C. Shiel Jr., MD. *Magnetic Resonance Imaging (MRI Scan)*. 1996-2020 MedicineNet, Inc, Nov 2019, (Date accessed: 02.14.2020). URL https://www.medicinenet.com/mri_scan/article.htm.

[43] Prof. Stacy Goergen Dr. Nick Ferris. *Gadolinium Contrast Medium (MRI Contrast agents)*. The Royal Australian and New Zealand College of Radiologists, 2017, (Date accessed: 02.14.2020). URL https://www.insideradiology.com.au/gadolinium-contrast-medium/.

[44] Judith Hurwitz and Daniel Kirsch. *Machine Learning For Dummies, IBM Limited edition*. John Wiley Sons, Inc, 2018. URL https://www.ibm.com/downloads/cas/GB8ZMQZ3.

[45] Jason Brownlee. *What is Machine Learning?* Machine Learning Mastery, (Date accessed: 02.04.2020), April 2018. URL https://machinelearningmastery.com/what-is-machine-learning/.

[46] Farhan Saeed. *9 Powerful examples of Artificial Intelligence in use today*. IQVIS, (Date accessed: 02.05.2020), Jan 2017. URL https://www.iqvis.com/blog/9-powerful-examples-of-artificial-intelligence-in-use-today/.

[47] Prince Yadav. *Machine Learning vs. Deep Learning*. Medium, (Date accessed: 02.05.2020), June 2018. URL https://medium.com/@mail2princeyadav/machine-learning-vs-deep-learning-b5c5a4fc5c.

[48] Udacity India Rachit Kumar Agrawal. Difference between machine learning, deep learning and artificial intelligence, March 2018, Date accessed: 02.05.2020. https://medium.com/@UdacityINDIA/difference-between-machine-learning-deep-learning-and-artificial-intelligence-e9073d43a4c3, Medium.

[49] Dr. Michael J. Garbade. *Regression Versus Classification Machine Learning: What's the Difference?* Medium, (Date accessed: 02.05.2020), Aug 2018. https://medium.com/quick-code/regression-versus-classification-machine-learning-whats-the-difference-345c56dd15f7.

[50] Thomas Simonini. An introduction to reinforcement learning, March 2018. https://medium.com/free-code-camp/an-introduction-to-reinforcement-learning-4339519de419, Medium,freecodecamp.org (Date accessed: 02.05.2020).

[51] Isha Salian. *SuperVize Me: What's the Difference Between Supervised, Unsupervised, Semi-Supervised and Reinforcement Learning?* NVIDIA, (Date accessed: 02.05.2020), Aug 2018. URL https://blogs.nvidia.com/blog/2018/08/02/supervised-unsupervised-learning/.

[52] L. Deng and D. Yu. *Book: Deep learning: methods and applications.* Foundations and Trends in Signal Processing, 7(3–4), pages.24-40, 2014.

[53] Niessen W.J. Klein S. de Groot M. Ikram M.A. Vernooij M.W. Li, B. and E.E. Bron. A hybrid deep learning framework for integrated segmentation and registration: evaluation on longitudinal white matter tract changes. *In International Conference on Medical Image Computing and Computer-Assisted Intervention (pp. 645-653). Springer, Cham.*, October 2019, arXiv:1908.10221.

[54] Fang S. Zhao Y. Wang P. Liu, T. and J. Zhang. Implementation of training convolutional neural networks. *arXiv preprint arXiv:1506.01195.*, 2015.

[55] SP.Chokkalingam N. Deepa. Deep convolutional neural networks (cnn) for medical image analysis. *International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249 – 8958, Volume-8, Issue-3S.*, February 2019.

[56] Wiesel T. N. Hubel, D. H. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology, 160(1), 106–154. doi:10.1113/jphysiol.1962.sp006837*, 1962.

[57] Jianxin Wu. Convolutional neural networks. pages 5–6, February, 2020. URL https://cs.nju.edu.cn/wujx/teaching/15_CNN.pdf. National Key Lab for Novel Software Technology Nanjing University, China.

[58] Y.H. Liu. Feature extraction and image recognition with convolutional neural networks. *In Journal of Physics: Conference Series (Vol. 1087, No. 6, p. 062032). IOP Publishing.*, 2018. URL https://iopscience.iop.org/article/10.1088/1742-6596/1087/6/062032/pdf.

[59] Creative Commons Attribution 4.0 International License. *Zero padding.* (Date accessed: 02.08.2020), Dec 2017. URL https://machinelearning.wtf/terms/zero-padding/.

[60] Ijomah W. Gachagan A. Nwankpa, C. and S. Marshall. Activation functions: Comparison of trends in practice and research for deep learning. *arXiv preprint arXiv:1811.03378.*, 2018.

[61] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[62] Harshall Lamba. Understanding semantic segmentation with unet., Feb 2019. https://towardsdatascience.com/understanding-semantic-segmentation-with-unet-6be4f42d4b47, Online; (Date accessed: 02.13.2020).

[63] IceCream Labs. *3x3 convolution filters — A popular choice.* (Date accessed: 02.14.2020), Aug 2018. URL https://medium.com/@icecreamlabs/3x3-convolution-filters-a-popular-choice-75ab1c8b4da8.

[64] MBA from Indian Institute of Management Bangalore. Arjun Balasubramanian. *How do skip connections work in a fully convolutional neural network?* (Date accessed: 02.14.2020), Nov 2018. URL https://www.quora.com/How-do-skip-connections-work-in-a-fully-convolutional-neural-network.

[65] Michael P.; French Andrew P.; Jackson Aaron S.; Pridmore Tony P. Benson, Eze; Pound. Deep hourglass for brain tumor segmentation. *In Pre-conference proceedings of the 7th medical image computing and computer-assisted interventions (MICCAI) BraTS Challenge (pp. 46-51). In BrainLes 2018: Brainlesion: Glioma,*

Multiple Sclerosis, Stroke and Traumatic Brain Injuries, 419-428. Springer. doi:10.1007/978-3-030-11726-9$_3$7, *pages*419 − −428, 26*January*2019. *ISSN*0302 − 9743.

[66] LISA lab. Deeplearning 0.1 documentation - unet., Last updated on Jun 15, 2018. URL http://deeplearning.net/tutorial/unet.html. Online; (Date accessed: 02.13.2020).

[67] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. volume abs/1608.06993. CoRR, arXiv, 1608.06993, 2016. URL http://arxiv.org/abs/1608.06993. dblp computer science bibliography, https://dblp.org, Mon, 10 Sep 2018 15:49:32 +0200.

[68] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[69] Besir Kurtulmus. Introduction to image saliency detection., Jan 2017. https://algorithmia.com/blog/introduction-image-saliency-detection, (Date accessed: 02.09.2020).

[70] Preeti Aggarwal, Vig Renu, Sonali Bhadoria, and C. Dethe. Role of segmentation in medical imaging: A comparative study. *International Journal of Computer Applications*, 29:54–61, 09 2011. doi: 10.5120/3525-4803.

[71] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[72] Zhen Ma, João Manuel R.S. Tavares, Renato Natal Jorge, and T. Mascarenhas. A review of algorithms for medical image segmentation and their applications to the female pelvic cavity. *Computer Methods in Biomechanics and Biomedical Engineering*, 13(2):235–246, 2010. doi: 10.1080/10255840903131878. PMID: 19657801.

[73] Flip Korn, Nikolaos Sidiropoulos, Christos Faloutsos, Eliot Siegel, and Zenon Protopapas. Fast nearest neighbor search in medical image databases. Technical report, USA, 1996.

[74] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, April 2017. ISSN 1939-3539. doi: 10.1109/TPAMI.2016.2572683.

[75] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. 2015. 1511.00561, arXiv, cs.CV.

[76] Jia W. He X. et al. Hesamian, M.H. Deep learning techniques for medical image segmentation: Achievements and challenges. *J Digit Imaging 32, 582–596*, May 2019. URL https://doi.org/10.1007/s10278-019-00227-x.

[77] Özgün Çiçek, Ahmed Abdulkadir, Soeren S. Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation, 2016. 1606.06650, arXiv, cs.CV.

[78] Yu. Gordienko, Peng Gang, Jiang Hui, Wei Zeng, Yu. Kochura, O. Alienin, O. Rokovyi, and S. Stirenko. Deep learning with lung segmentation and bone shadow exclusion techniques for chest x-ray analysis of lung cancer. *Advances in Computer Science for Engineering and Education*, page 638–647, May 2018. ISSN 2194-5365. doi: 10.1007/978-3-319-91008-6_63. URL http://dx.doi.org/10.1007/978-3-319-91008-6_63. Springer International Publishing, ISBN: 9783319910086.

[79] Michael P.; French Andrew P.; Jackson Aaron S.; Pridmore Tony P. Benson, Eze; Pound. Deep hourglass for brain tumor segmentation. *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries; Lecture Notes in Computer Science*, pages 419–428, 2019. ISSN 0302-9743. URL https://link.springer.com/chapter/10.1007/978-3-030-11726-9_37. Springer, ISBN: 9783030117252.

[80] Ekami Tuatini Godard. Understanding the dice coefficient., September 2017. URL https://forums.fast.ai/t/understanding-the-dice-coefficient/5838. Date accessed: 02.12.2020.

[81] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, L. Lanczi, E. Gerstner, M. Weber, T. Arbel, B. B. Avants, N. Ayache, P. Buendia, D. L. Collins, N. Cordier, J. J. Corso, A. Criminisi, T. Das, H. Delingette, Ç. Demiralp, C. R. Durst, M. Dojat, S. Doyle,

J. Festa, F. Forbes, E. Geremia, B. Glocker, P. Golland, X. Guo, A. Hamamci, K. M. Iftekharuddin, R. Jena, N. M. John, E. Konukoglu, D. Lashkari, J. A. Mariz, R. Meier, S. Pereira, D. Precup, S. J. Price, T. R. Raviv, S. M. S. Reza, M. Ryan, D. Sarikaya, L. Schwartz, H. Shin, J. Shotton, C. A. Silva, N. Sousa, N. K. Subbanna, G. Szekely, T. J. Taylor, O. M. Thomas, N. J. Tustison, G. Unal, F. Vasseur, M. Wintermark, D. H. Ye, L. Zhao, B. Zhao, D. Zikic, M. Prastawa, M. Reyes, and K. Van Leemput. The multimodal brain tumor image segmentation benchmark (brats). *IEEE Transactions on Medical Imaging*, 34(10):1993–2024, Oct 2015. ISSN 1558-254X. doi: 10.1109/TMI.2014.2377694.

[82] Akbari H. Sotiras A. Bilello M. Rozycki M. Kirby J.S. Freymann J.B. Farahani K. Bakas, S. and C. Davatzikos. Advancing the cancer genome atlas glioma mri collections with expert segmentation labels and radiomic features. *Scientific Data 4, 170117 EP*, September 2017. doi: DOI:10.1038/sdata.2017.117.

[83] Akbari H. Sotiras A. Bilello M. Rozycki M. Kirby J. Freymann J. Farahani K. Bakas, S. and C. Davatzikos. Segmentation labels and radiomic features for the pre-operative scans of the tcga-gbm collection. *The Cancer Imaging Archive, 286.*, 2017. doi: DOI:10.7937/K9/TCIA.2017.KLXWJJ1Q.

[84] Akbari H. Sotiras A. Bilello M. Rozycki M. Kirby J. Freymann J. Farahani K. Bakas, S. and C. Davatzikos. Segmentation labels and radiomic features for the pre-operative scans of the tcga-lgg collection. *The Cancer Imaging Archive, 286.*, 2017. doi: DOI:10.7937/K9/TCIA.2017.GJQ7R0EF.

[85] Miccai_brats_2018_proceedings_shortpapers. *Pre-Conference Proceedings of the 7th MICCAI BraTS Challenge (2018)*, pages 2,3,28–32, 46–51, 562–568, September 16, 2018. URL https://www.cbica.upenn.edu/sbia/Spyridon.Bakas/MICCAI_BraTS/MICCAI_BraTS_2018_proceedings_shortPapers.pdf.

[86] Shi L. Luo Y. Yang W. Li H. Liang P. Li K. Mok V.C. Chu W.C. Sun, X. and D. Wang. Histogram-based normalization technique on human brain magnetic resonance images from different acquisitions. *Biomedical engineering online, 14(1), p.73.*, 2015.

[87] Jaber Juntu, Jan Sijbers, Dirk Van Dyck, and Jan Gielen. Bias field correction for mri images. In Marek Kurzyński, Edward Puchała, Michał Woźniak, and Andrzej

żołnierek, editors, *Computer Recognition Systems*, pages 543–551, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[88] Andrey Fedorov (SPL BWH) Ron Kikinis (SPL BWH) Slicer Wiki, Nick Tustison (UPenn). Documentation/4.3/modules/n4itkbiasfieldcorrection., 2013. URL https://www.slicer.org/w/index.php. (Online; Date accessed: 02.11.2020).

[89] Fabian Isensee, Philipp Kickingereder, Wolfgang Wick, Martin Bendszus, and Klaus H. Maier-Hein. No new-net. In Alessandro Crimi, Theo van Walsum, Spyridon Bakas, Farahani Keyvan, Mauricio Reyes, and Hugo Kuijf, editors, *Brainlesion*, pages 234–244. In Pre-conference proceedings of the 7th medical image computing and computer-assisted interventions (MICCAI) BraTS Challenge (pp. 222-229), 1 2019. ISBN 9783030117252. doi: 10.1007/978-3-030-11726-9_21. Springer Verlag.

[90] NVIDIA @ ptrblck.de ptrblck, DL Software Engineer. How to extract smaller image patches (3d)?, April 2018. URL https://discuss.pytorch.org/t/how-to-extract-smaller-image-patches-3d/16837/3. Online; Date accessed: 02.12.2020.

[91] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning*. 2020. URL https://d2l.ai. chapter 6 (Convolutional Neural Networks), page: 231.

[92] PhD student Research Scientist Arthur Douillard. Densely connected convolutional networks, 7 May 2018. URL https://arthurdouillard.com/post/densenet/. Online; Date accessed: 02.12.2020.

[93] Geoff Pleiss, Danlu Chen, Gao Huang, Tongcheng Li, Laurens van der Maaten, and Kilian Q. Weinberger. Memory-efficient implementation of densenets, 2017.

[94] Analytics Nouroz Rahman and Data Science. What is the benefit of using average pooling rather than max pooling?, May 2017. URL https://www.quora.com/What-is-the-benefit-of-using-average-pooling-rather-than-max-pooling. Date accessed: 02.12.2020.

[95] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res., JMLR.org*, 15(1):1929–1958, January 2014. ISSN 1532-4435.

[96] Martin Kolařík, Radim Burget, Vaclav Uher, Kamil Riha, and Malay Dutta. Optimized high resolution 3d dense-u-net network for brain and spine segmentation. *Applied Sciences*, 9:404, 01 2019. doi: 10.3390/app9030404.

Appendix: A
Experimental Testing

Background: To test with a set of images via U-Net before moving them to DenseNet.

Testing Grounds: Use original images with default sizes. If that set fails, shrink the size to (64, 64, 64)

System Requirements:

- **System 1:** Ubuntu OS, 8 GB Ram, 1 TB HDD, i5 Core Processor, NVidia GeForce 840MX
- **System 2:** Windows 10, 32 GB Ram, 500 GB SSD, i7 Quad Core Processor, NVidia GTX980 (4 GB GPU RAM)
- **System 3:** Vmware 14 with Ubuntu OS, 8 GB Ram, 50 GB HDD, 2 Core (8 Processors total), Virtualization ON

System 1 - Ubuntu OS

- Code was loaded perfectly, pre-processing completed. (Success).
- Numerous missing libraries were installed via python script. (Success)
- There were two hardware options for training the model:
    1. **Nccl (GPU-based):** System 1's GPU is not very powerful and crashed during execution.

2. **CPU (internal CPU-based):** System 1's CPU memory was insufficient to handle the demands of TensorFlow, which exhausted the available memory on this system. The training process was not completed.

- System 1 lacks multi-threading capabilities, which is required to execute model training.

System 2 - Windows OS

- Numerous missing libraries were installed, in addition to anaconda. Being a Windows system, the different coding structure presented additional challenges in the form of code modifications needed in order to run. T This system ended up being taxed more than System 1 despite having greater processing power.
- Code was loaded perfectly, pre-processing completed. (Success)
- When running the training model using 114 variables and 8269676 parameters with an input size of 31.55 MB, the script crashed and produced the error below.

1. **ValueError:** Op type not registered 'MaxBytesInUse' in binary running on HOME-PC. Make sure the Op and Kernel are registered in the binary running in this process. Note that if you are loading a saved graph which used ops from tf.contrib, accessing (e.g.) 'tf.contrib.resampler' should be done before importing the graph, as contrib ops are lazily registered when the module is first accessed while building NodeDef 'PeakMemoryTracker/MaxBytesInUse'
2. **Follow-up research** on this error indicates that, because TensorFlow uses a significant amount of memory and does not provide a means to track memory usage, use of a GPU with at least 32 GB of memory is recommended. Other users facing the same issue have posted to the following Github thread: https://github.com/tensorflow/tensorflow/issues/492

System 3 - Ubuntu OS VMWare

- Numerous missing libraries were installed via python script. (Success)
- Insufficient CPU power (2 cores) resulted in crashes each time the script was run.

- As the GPU was virtualized, the script did not recognize it and threw an error stating that NVidia is not configured to run.
- On a positive note, the idea to shrink the images from original to (64, 64, 64) sprung from this round of testing.

Ideal Dense-Net Environment

- **System Requirements:**
  1. Between 32 and 48 CPU cores
  2. NVidia T4 GPU focused towards Tensor cores (https://www.nvidia.com/en-us/data-center/tesla-t4/)
  3. Linux OS
  4. 128 GB RAM
  5. 1 TB SSD

Utilizing the currently available system architecture (detailed above), processing of each image would take months to test. This is in addition to the time that it would take to complete programming and testing of code. Given the issues encountered in attempting the above U-Net implementation, pseudo code is being provided in lieu of a working Dense U-Net model. This pseudo code details how to implement the transition of images from U-Net to DenseNet and can be utilized by other students performing similar research.