Menoufia University
Faculty of Science
Department of Mathematics and Computer Science

A Survey on Brain Tumor Classification

A Graduation Project
Submitted in Partial Fulfillment of B.Sc. Degree

Prepared By:

1- Nada Abdal Rasoual Safan

2- Sana Mohamed Safwat Elshamy

3- Mariam khalid El-gebali

4- Dai Hamdy Ibrahem

Supervised By

Dr. Ibrahim Omara

2020/2021

Table of Contents

1 Chapter 1 . Introduction

1.1 Overview

A **brain tumor** is an abnormal growth of tissue in the brain or central spine that can disrupt proper brain function Doctors refer to a tumor based on where the tumor cells began, and whether they are cancerous (malignant) or not (benign) All brain tumors can grow to damage areas of normal brain tissue if left untreated, which could be disabling and possibly fatal Brain and spinal cord tumors are different for everyone. They form in different areas develop from different cell types, and may have different treatment options. There are two basic kinds of brain tumors

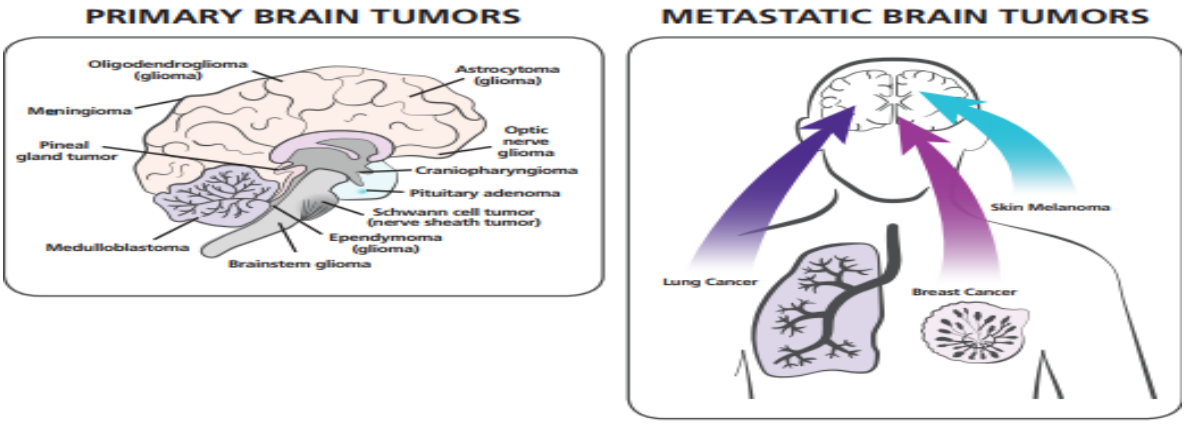
1.2 Metastatic brain tumors (secondary cancer)

Sometimes cancer starts in another part of the body and then travels through the bloodstream to the brain. This is known as a secondary cancer or metastasis. The cancers most likely to spread to the brain are melanoma, lung, breast, kidney and bowel. A metastasis keeps the name of the original cancer. For example, bowel cancer that has spread to the brain is still called metastatic bowel cancer; even though the person may be having symptoms because cancer is in the brain (begin as cancer elsewhere in the body and spread to the brain).

1.3 Primary brain tumors

A tumor that first develops in the brain is called primary brain cancer.It may spread to other parts of the nervous system, but rarely spreads to other parts of the body (start and tend to stay).

When doctors describe brain tumors, they often use the words “benign” or “malignant.” Those descriptions refer to the degree of malignancy or aggressiveness of a brain tumor. It is not always easy to classify a brain tumor as “benign” or “malignant” as many factors other than pathological features contribute to the outcome.



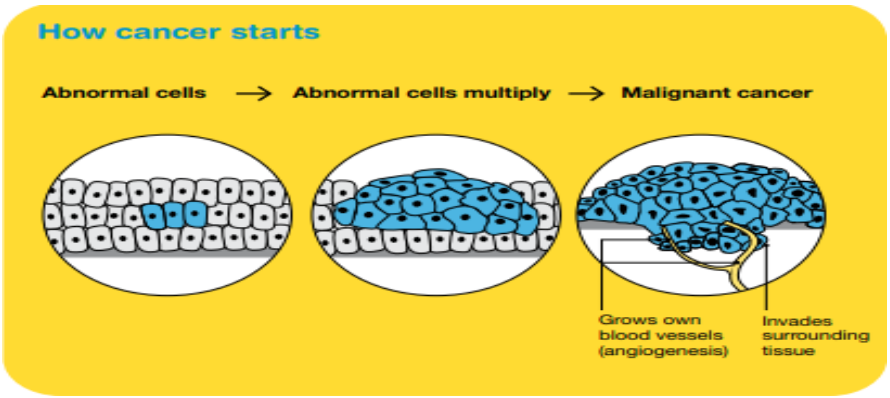
A tumor that starts in the brain is a primary brain tumor. Glioblastoma multiforme, astrocytoma, medulloblastoma and ependymoma are examples of primary brain tumors. Primary brain tumors are grouped into benign tumors and malignant tumors.

1.3.1 Benign brain tumors

Usually grow slowly and are unlikely to spread. They may also be called low-grade or non-malignant tumors. A benign tumor may grow and affect how the brain works. This can be life-threatening and may require urgent treatment. Sometimes a benign tumor can change over time and become high grade.

1.3.2 Malignant brain tumors

Malignant brain tumors can grow rapidly. They are considered life-threatening because they may spread within the brain and spinal cord, or come back after treatment. A malignant brain tumor may be called brain cancer. Unlike malignant tumors in other parts of the body, malignant brain tumors usually do not spread outside the brain and spinal cord.



1.4 Brain Tumor Causes

Brain tumors are thought to arise when certain genes on the chromosomes of a cell are damaged and no longer function properly. These genes normally regulate the rate at which the cell divides (if it divides at all) and repair genes that fix defects of other genes, as well as genes

that should cause the cell to self-destruct if the damage is beyond repair. In some cases, an individual may be born with partial defects in one or more of these genes. Environmental factors may then lead to further damage. In other cases, the environmental injury to the genes may be the only cause. It is not known why some people in an "environment" develop brain tumors, while others do not.

Once a cell is dividing rapidly and internal mechanisms to check its growth are damaged, the cell can eventually grow into a tumor. Another line of defense may be the body's immune system, which optimally would detect the abnormal cell and kill it. Tumors may produce substances that block the immune system from recognizing the abnormal tumor cells and eventually overpower all internal and external deterrents to its growth.

A rapidly growing tumor may need more oxygen and nutrients than can be provided by the local blood supply intended for normal tissue. Tumors can produce substances called angiogenesis factors that promote the growth of blood vessels. The new vessels that grow increase the supply of nutrients to the tumor, and, eventually, the tumor becomes dependent on these new vessels. Research is being done in this area, but more extensive research is necessary to translate this knowledge into potential therapies.

1.5 Symptoms:

Symptoms vary depending on the location of the brain tumor, but the following may accompany different types of brain tumors:

- Headaches that may be more severe in the morning or awaken the patient at night
- Seizures or convulsions
- Difficulty thinking, speaking or articulating
- Personality changes
- Weakness or paralysis in one part or one side of the body
- Loss of balance or dizziness
- Vision changes
- Hearing changes
- Facial numbness or tingling
- Nausea or vomiting, swallowing difficulties
- Confusion and disorientation

1.6 Side effects of a brain tumor: emotional, cognitive, physical

1.6.1 Emotional side effects

Emotional side effects are natural for any major disease, brain tumor or not. When a brain tumor is diagnosed, it can take away your sense of security and control. Uncertainty is among the most challenging things that you may have to grapple with on a day-to-day basis.

Depression is also very common. In addition to the emotional side effects related to receiving the diagnosis, the type, size and location may also affect your emotions. Some people with brain tumors experience intense emotions or personality changes because the tumor is located in an area that controls emotional functioning. You do not have to feel guilty about emotional challenges. They are very common. A member of your healthcare team can refer you to a professional like a clinical social worker, clinical psychologist, or neuropsychologist. Support groups, which can be found on the ABTA website, may also help.

1.6.2 Cognitive side effects

Cognitive side effects are those that affect your ability to process information and communicate. You may find it harder to find the words you need or calculate the tip at a restaurant. It may be difficult to concentrate or remember things. Your abilities may be better on some days and worse on others. Again, medical professionals and special types of therapy can help strengthen these abilities during and after treatment.

1.6.3 Physical effects

Physical effects are common as treating a brain tumor can take a great toll on your body. While the effects are different for every person, a brain tumor and subsequent treatments may change your appearance, strength and ability, as well as your ability to carry on a full, active day. Additional common side effects include seizures, pain, fatigue, weakness, nausea, headaches and hair loss. Many people with brain tumors are able to handle these changes by being realistic. They set priorities and do only what needs to be done. They plan frequent rest and ask for help. In addition, medical services, such as physical and occupational therapy, can help improve body function. Make sure to speak with your doctor about any symptoms you make have, so that they can be medically treated as optimally as possible.

1.7 Types of brain tumors

1.7.1 Benign tumor

This kind of tumor is not cancer. It tends to grow slowly. Most benign brain tumors don’t grow into nearby tissue. Once removed, they usually don’t grow back. A benign tumor can cause symptoms like a malignant tumor depending on its size and location in the brain.

1.7.2 Malignant tumor

This kind of tumor is cancer. It usually grows fast, and grows into nearby tissue. This can make it hard to remove fully. A malignant brain tumor may grow back after treatment.

The most common type of primary brain tumor is a glioma. This type begins in the supportive (glial) tissue of the brain. Some gliomas tend to grow slowly. Others grow and spread quickly. Some types of glioma include:

1.7.3 Astrocytoma:

This kind of tumor comes from small star-shaped cells called astrocytes. In adults, an astrocytoma usually grows in the cerebrum. In children, they can grow in the cerebellum, cerebrum, and brain stem. Most astrocytoma's spread into nearby normal brain tissue and are hard to cure with surgery. Glioblastoma is a type of astrocytoma that tends to grow very quickly.

1.7.4 Brain stem glioma:

This kind of tumor of the brain stem is more common in children than in adults. Because the brain stem controls many important functions, such as breathing and heart rate, this kind of tumor usually can’t be removed by surgery.

1.7.5 Ependymoma:

This kind of tumor starts in cells that line the fluid-filled spaces within the brain (ventricles). It doesn’t often grow into nearby brain tissue. This means in some cases it can be cured with surgery.

1.7.6 Oligodendroglioma:

This kind of tumor starts in cells that make myelin, the fatty substance that surrounds nerve cells. Like an astrocytoma, this tumor tends to spread into nearby brain tissue and is often hard to cure with surgery.

1.7.7 Optic nerve glioma:

This kind of tumor grows in or around the nerve that sends messages from the eyes to the brain. This can cause vision changes. It can also cause hormone changes, due to its location near the pituitary gland.

1.7.8 Primitive Neuroectodermal Tumor (PNET) :

This kind of tumor grows more often in children. It can grow anywhere in the brain in the primitive form of nerve cells. One type is the medulloblastoma. This kind of tumor is found in the cerebellum. They are more common in children than in adults. They tend to grow and spread quickly, but they can often be treated effectively.

1.7.9 Tumor of the pineal gland:

This kind of tumor grows in and around the pineal gland. This is a tiny organ near the center of the brain. The tumor can be slow-growing, called pineocytoma. Or it can be fast-growing, called pineoblastoma.

1.7.10 Pituitary tumor:

This kind of tumor starts in the pituitary gland at the base of the brain. It is almost always benign. But it can cause serious symptoms because of its location, and because it may secrete excess hormones.

1.7.11 Craniopharyngioma:

This kind of tumor starts near the pituitary gland. It is usually slow growing. But it can cause symptoms if it presses on the pituitary gland or on nearby nerves.

1.7.12 Schwannoma:

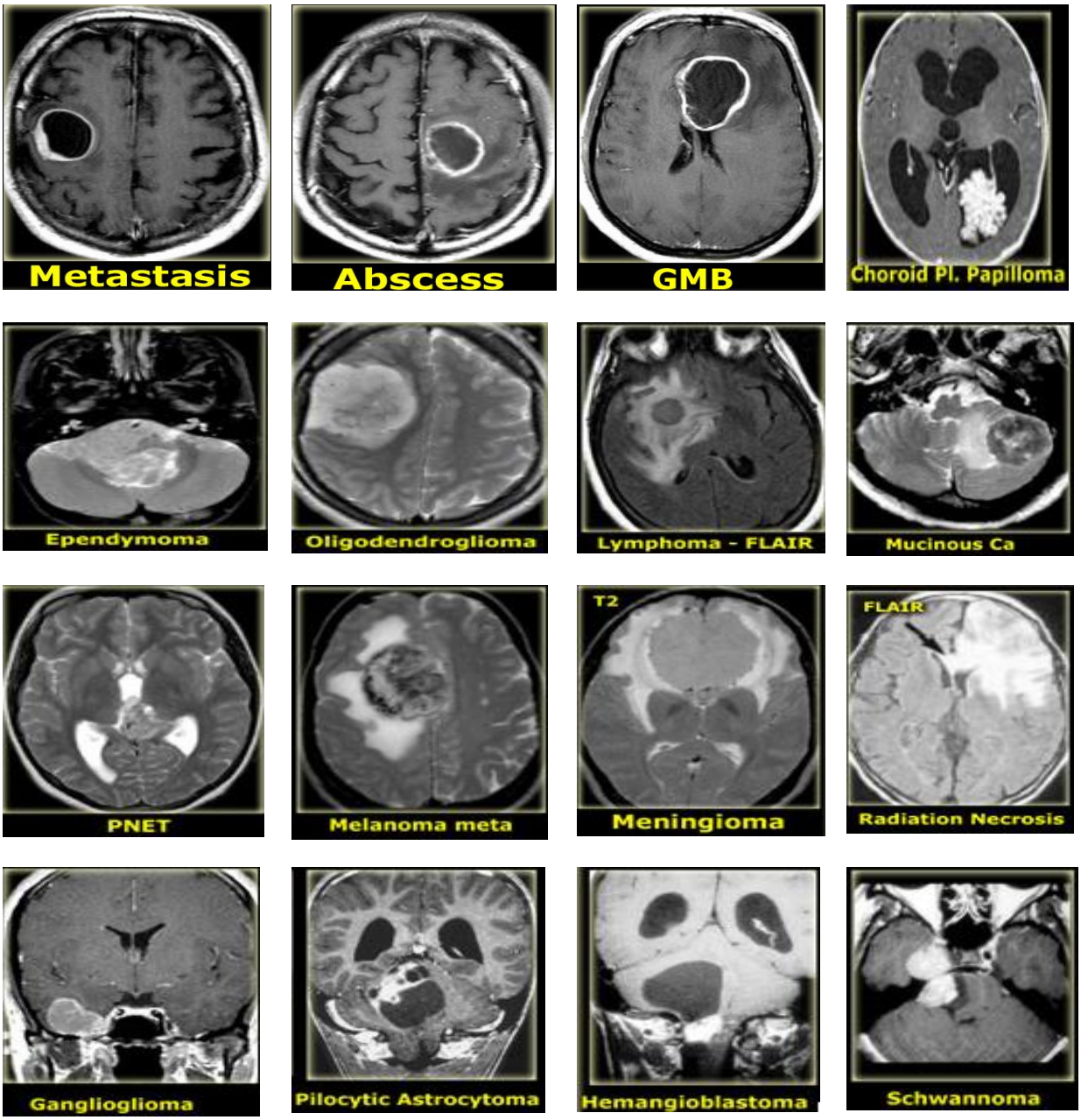
This kind of tumor starts in myelin-making cells that surround certain nerves. It’s most common in the vestibular nerve in the inner ear that helps with balance. If it grows there, the tumor is called a vestibular schwannoma or an acoustic neuroma. This type of tumor is usually benign.

1.7.13 Meningioma:

This kind of tumor starts in the outer linings of the brain (meninges). It is more common in adults. Many meningiomas can be removed with surgery, but some may grow back.

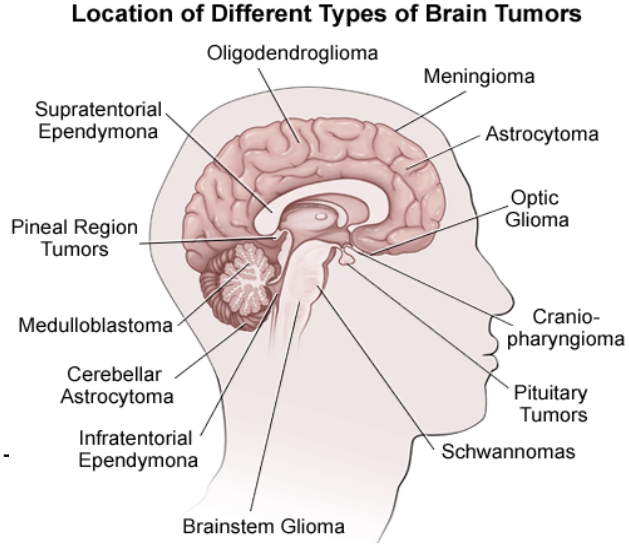
1.7.14 Primary central nervous system lymphoma:

This is an aggressive, rare type of tumor that starts in lymphocytes. This is a type of immune cell. The tumor is more common in people with a disease of the immune system, such as AIDS. But it can grow in healthy people.



1.7.15 **Types of** secondary **brain tumors**

A secondary brain tumor is also known as a metastatic brain tumor. This is cancer that starts in another organ and then travels to the brain. In adults, secondary brain tumors are more common than primary brain tumors. Cancer in the brain that has spread from another part of the body is not considered brain cancer. It is still the same type of cancer as where it started. For



-

example, lung cancer that has spread to the brain is called metastatic lung cancer. These are some of the most common types of cancer that spread to the brain:

- **Lung cancer**
- **Breast cancer**
- **Melanoma**
- **Colon cancer**
- **Kidney cancer**

1.8 Radiology Imaging Techniques of Brain Tumors

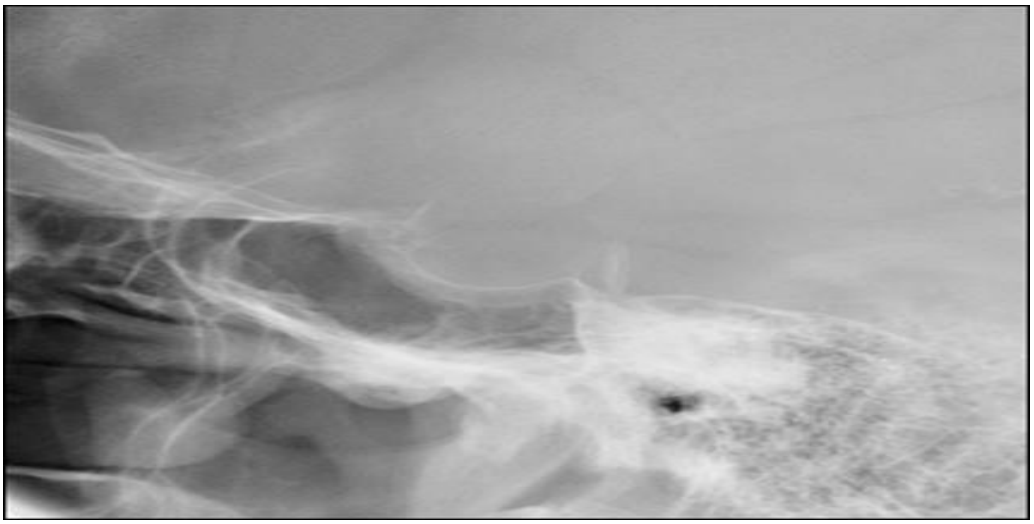
1.8.1 Conventional imaging methods

For decades, diagnostic imaging dominated conventional non-invasive and invasive methods, and later invasive contrast x-ray techniques. During the second half of the twentieth century, a number of different projection x-ray radiographs of the head and their modifications, as well as complex invasive contrast imaging techniques, such as pneumoencephalography, ventriculography, and myelography, were improved. Another imaging method is ultrasonography, which can be used for neuron avigation during operation of brain tumors.

1.8.2 Conventional non-invasive X-ray methods

In the past, conventional non-invasive x-ray examination (radiography of the head) was the basic diagnostic method in neuroradiology. The baseline projections are posteroanterior (PA) and lateral x-ray projections of the skull. A PA projection is centered by orbitomeatal lines and provides anatomical information about the skull and frontal structures. A lateral projection shows the configuration of the skull and the skull base.

Modification of a PA projection by Caldwell with an x-ray beam inclination of 15°–23°, caudal to the orbitomeatal line, provides a clearer view of the os petrosum. With an x-ray beam inclination of 37°, caudal to the orbitomeatal line, we obtain a semiaxial Water's projection, which shows paranasal cavities and structures of the zygomaticomaxillar complex. An x-ray beam inclination of 30°, caudal to the orbitomeatal line, in the anteroposterior (AP) direction provides the Towne's projection, which is appropriate for imaging the os spheoidale, foramen magnum, and pyramids, and their dorsal edges in particular.



A submentovertical projection is an axial projection of the skull with the x-ray beam passing approximately perpendicular to the orbitomeatal line, and is suitable for imaging the os sphenoidale and the base middle fossa foramina. The Stenvers projection with a 45° rotation of the head from the PA line, and with a caudal x-ray beam inclination of 10°–15°, is the most common projection for imaging the os petrosum, providing a good display of the tip of the pyramid, the structures of the inner ear, and the meatus acusticus internus. The Schüller projection is a lateral projection with a caudal x-ray beam inclination of 30° and is employed for enhanced imaging and evaluation of the processes mastoideus pneumatization. A modification of these projections is a lateral projection by Runström I, with an x-ray beam inclination of 15° and a projection by Runström II with a caudal x-ray beam inclination of 45° [1]. Other special projections focus on the sella turcica, canalis opticus.

1.8.3 Computed Tomography – CT

From its first test scan on a mouse, in 1967, to current medical practice, the CT scanner has become a core imaging tool. Initially financed by money from Beatles' record sales, the first patient scan was performed in 1971. Only 8 years later, a Nobel Prize in Physics and Medicine was awarded to Gofrey New bold Hounsfield and Allan McLeod Cormack for their discovery. The prototype (EMI Ltd.) was installed at Atkinson Morley’s Hospital in South London where the first patient, a middle-aged lady with a suspected frontal lobe tumor, was scanned on 1st October 1971.

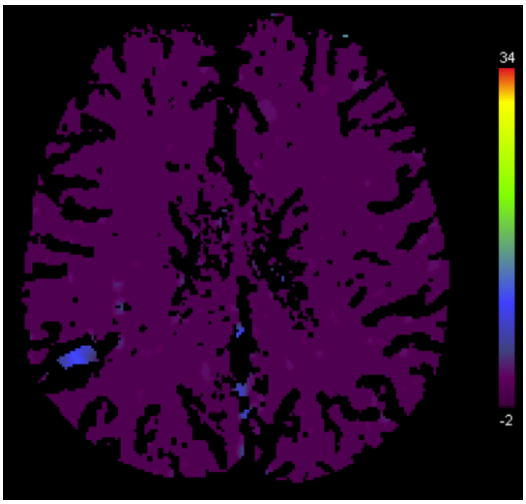
The rapid development of CT scanners, a new generation of CT devices, and advanced post processing technologies in recent years has enabled the creation of progressive, advanced CT protocols for the diagnosis of individual anatomical regions with respect to the pathological processes that can be diagnosed. Technological improvements and new CT applications in neuroradiology are mainly related to CT angiography and CT perfusion with a dynamic contrast agent bolus.

The basic CT examination of brain tumors involves standard non-contrast enhanced and contrast enhanced imaging. Compared to MR, CT is superior in the detection of calcification and bone abnormalities, and it is also less time consuming.

In CT diagnosis, depending on the type of examination, iodinated contrast agents are administered, in different quantities and by different modes. Iodinated contrast agents are divided into ionic, high-osmolar contrast agents and non-ionic, low-osmolar or iso-osmolar contrast agents. Intravenous administration of contrast agents may cause various negative allergic reactions, which are divided into early (within 20 min) and late effects. In practice, non-ionic contrast media are generally preferred as, due to their low osmolality, they result in significantly fewer negative effects.

1.8.4 Magnetic Resonance Imaging – MRI

Historically, many scientists have contributed to the study of NMR (MRI), which led to construction of reliable MR scanners for clinical practice. Isidor Isaac Rabi in 1930 began by studying the magnetic properties of atomic nuclei (Nobel Prize in Physics in 1944). The first successful nuclear magnetic resonance experiment with NMR precision measurements was made independently in 1946 by Felix Bloch and Edward Mills Purcell (they jointly received the Nobel Prize in Physics in 1952). In 1971, Raymond Vahan Damadian, measured T1 and T2 relaxation times of excised normal and cancerous rat tissue and stated that tumor tissue had longer relaxation times than normal tissue. He is the inventor of the first MR Scanning Machine (1977). In March 1973 Paul C. Lauterbur



published the first 2D NMR images of two 1 mm capillaries filled with water [10] and in 1974 the image of thoracic cavity of mouse. He called his imaging method zeugmatography. This term was later replaced by NMR imaging. Peter Mansfield with Grannel described the use of magnetic field gradients to acquire spatial information in NMR. P.C. Lauterbur and Sir Peter Mansfield received the Nobel Prize in 1952. The first commercial MR scanner (Picker Ltd.) in Europe was installed in 1983 in Manchester Medical School.

The main advantages of MRI are the possibilities of imaging individual anatomical regions in vivo with high tissue contrast, imaging in arbitrary planes, non-invasivity, and the absence of demonstrable detrimental effects on human health. Qualitative evaluation of tissues allows for four basic physical attributes: T1 and T2 relaxation, proton density, motion, and flow.

1.8.5 Digital subtraction angiography – DSA

Digital subtraction angiography (DSA) is a computer-assisted x-ray technique that subtracts images of bone and soft tissue to permit viewing of the cardiovascular system.

At the beginning of the process of subtraction, an image (the mask) is obtained before arrival of contrast material at the area of interest, and the mask image is placed into one of two digital memories. Then, one or more subsequent images are obtained after the arrival of a contrast bolus and placed into a second digital memory. The mask image is digitally subtracted from the succeeding contrast image, resulting in contrast-filled structures that are rendered visible free of background detail. Subtraction is performed in real time.

Iodine contrast media are used for the visualization of vessels, however cerebral angiography using gadolinium as an alternative contrast medium in a patient with severe allergy to iodinated contrast medium may be performed.

Radiation, today known as X-rays, was discovered by the German physicist Wilhelm Röntgen (March 27, 1845–February 10, 1923) on November 8, 1895 [52]. Discovery of X-rays is ranked as one of the best discoveries in medicine. X-rays are electromagnetic waves. The range of wavelengths corresponding to diagnostic imaging span from about 0.1 nm (at 12.4 keV) to 0.01 nm (at 124 keV) [53]. This type of radiation is ionizing.

In a vacuum X-ray tube, the electrons that make up the beam are emitted by a heated cathode filament. The electrons are then focused and accelerated towards the focal spot by a high voltage that is applied between the cathode filament and the anode. A generator is used to supply the X-ray tube with a controlled high voltage between the cathode and anode, and a controlled current to the cathode. The electron beam strikes the rotating anode “target” and part of its kinetic energy (less than 1%) is converted into X-ray photons, while the rest is converted

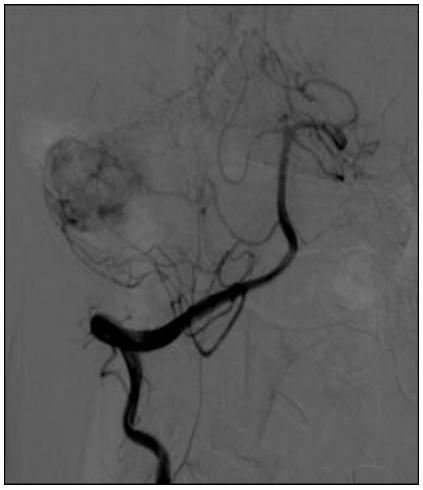
into heat, which heats up the anode. The X-ray beam leaves the tube through the tube window and passes onto the patient. Some of the X-rays pass through the patient, while some are absorbed. The resulting radiation pattern is detected by a flat panel digital X-ray detector (FPD).

FPD system is superior to the image intensifier as it visualizes small intracranial vessels combined with a significant reduction of radiation dose, and is able to create high-quality 3D DSA images on which high spatial resolution allows precise visualization of small vessels, such as perforating vessels. DSA images are then displayed on the LCD monitor with high resolution and different screen layouts, which can be connected to several image sources.

The first carotid angiography was performed by Portuguese Egas Moniz (1874-1955) in 1927; he is considered as a pioneer of cerebral angiography. He reported the first case of cerebral angiography at the Societe de Neurologie in Paris on July 7, 1927. Surprisingly, most angiograms were performed to visualize the intracranial portion of the carotids in cases of tumors, to look for abnormal displacement of arterial branches, with little interest in the vascular disease itself.

The technique, how to obtain safe access to blood vessels was published by Sven-Ivar Seldinger (1921-1998) in 1953. DSA is an invasive technique, performed using a catheter; the most commonly used approach is the transfemoral approach. At the end of angiography, the puncture site can be safely closed by a closure device.

DSA is used to detect the blood vessels supplying the brain tumors, and also to control the hyper vascular tumor embolization (meningiomas, paragangliomas, haemangiopericytomas, juvenile nasopharyngeal angiofibromas and intraaxially located tumors: haemangioblastomas, hyper vascularized metastases and ependymomas). Presurgical or palliative embolization of a tumor can be performed by either an intra-arterial catheterization approach or direct puncture of the tumor artery.



DSA may also be used for a balloon occlusion test. Although 4D-CE-MRA may be useful for evaluating tumor stain in hyper vascular brain, head and neck tumors, it is not able to replace DSA in planning interventional procedures.Modern biplane DSA devices are very useful for neurovascular interventions, which also allows: 2D and 3D navigation for advanced embolization guidance; overlay of a DSA reference image over the matching live fluoro for guidance with less contrast media and less dose; cross-

sectional imaging to view anatomical structures of tumors in combination with the feeding vessels of the tumor; single-colour vascular flow visualization from a 2D DSA image series to visualize tumor perfusion tumor vascularization, tumor blush and demonstrate post embolization result; to fuse the dataset with a preprocedural CT, MR or PET image to show tumor activity; synchronize the 3D image to the gantry position; PACS connectivity; the reporting of patient exposure following an intervention.

Modern systems update dynamically to movements of the C-arm, table, zoom and source-to-image distance to facilitate efficient workflow during interventional procedures. By providing more effective and faster guidance, this potentially reduces the use of contrast agents and radiation dose. Pulse frequencies can be adapted to clinical needs according to the ALARA principle (As Low as Reasonably Achievable).

1.9 Types of databases (Datasets)

1.9.1 Data set 1

The experiments were done on the images taken from Harvard repository, the clinical dataset from Fortis Memorial Research Institute, Gurgaon, India, and images from Figshare repository. Images from the Harvard repository were divided into three versions as exiting in the literature. Table 1 details the three versions of the Harvard dataset that has been employed in the present paper. The first encompasses 50 images. The second contains 74 images. The third variant consists of 160 images. For the clinical dataset, images from 200 patients were used (age range 18–65 years) from the Fortis Memorial Research Institute, Gurgaon, India. The MRI scan was performed on a 3T MRI (Philips Ingenia, Best Netherland) using a 15-channel head coil. 3D fluid attenuating inversion recovery (FLAIR) was done using the scan parameters as follows: TR (ms) 4500, TE (ms) 279, NEX 1, and slice thickness 10 mm. From the data of 200 patients, 500 images were used out of which 250 were normal and 250 were abnormal, i.e., those affected by glioma tumors. The opinion of the radiologist was considered as the gold standard. Moreover, the distinction between the normal and abnormal categories was made by the radiologist having more than 25 years of experience in radiology. All the decisions were made by the consensus. The patient data were normalized using the T2.nii template on the SPM toolbox in MATLAB. A gap of about 10 mm between the slices was selected, thereby rendering 16 slices per patient. Additionally, solely to contrast the performance of the deep transfer learned model on another challenging medical data, experimentations were also done on the dataset from Figshare. It consists of 3064 brain MR images of 233 patients having glioma (1426), meningioma (708), and pituitary tumors (930).

T1-CE images were available in axial, coronal, and sagittal views. The images were available in. mat format with size as 512×512. Images were resized to match the dimensions of the input layer of the pre-trained model. Since the images were grayscale, additional channels were created by replicating the pixel values three times. The sample MR images are shown in Fig.1.1.

Particulars of the Harvard database

Dataset	Details
Version 1	50 T2-w: 20 abnormal affected with glioma tumor and 30 normal
Version 2	74 T2-w: 52 abnormal comprising of Glioma, herpes encephalitis, metastatic bronchogenic carcinoma, multiple sclerosis, and Alzheimer's, and 22 normal
Version 3	160 T2-w: 140 abnormal and 20 normal. The MR images in the abnormal category have glioma, meningioma, Alzheimer's disease, Pick's disease, sarcoma, Alzheimer's disease plus visual agnosia, and Huntington's disease

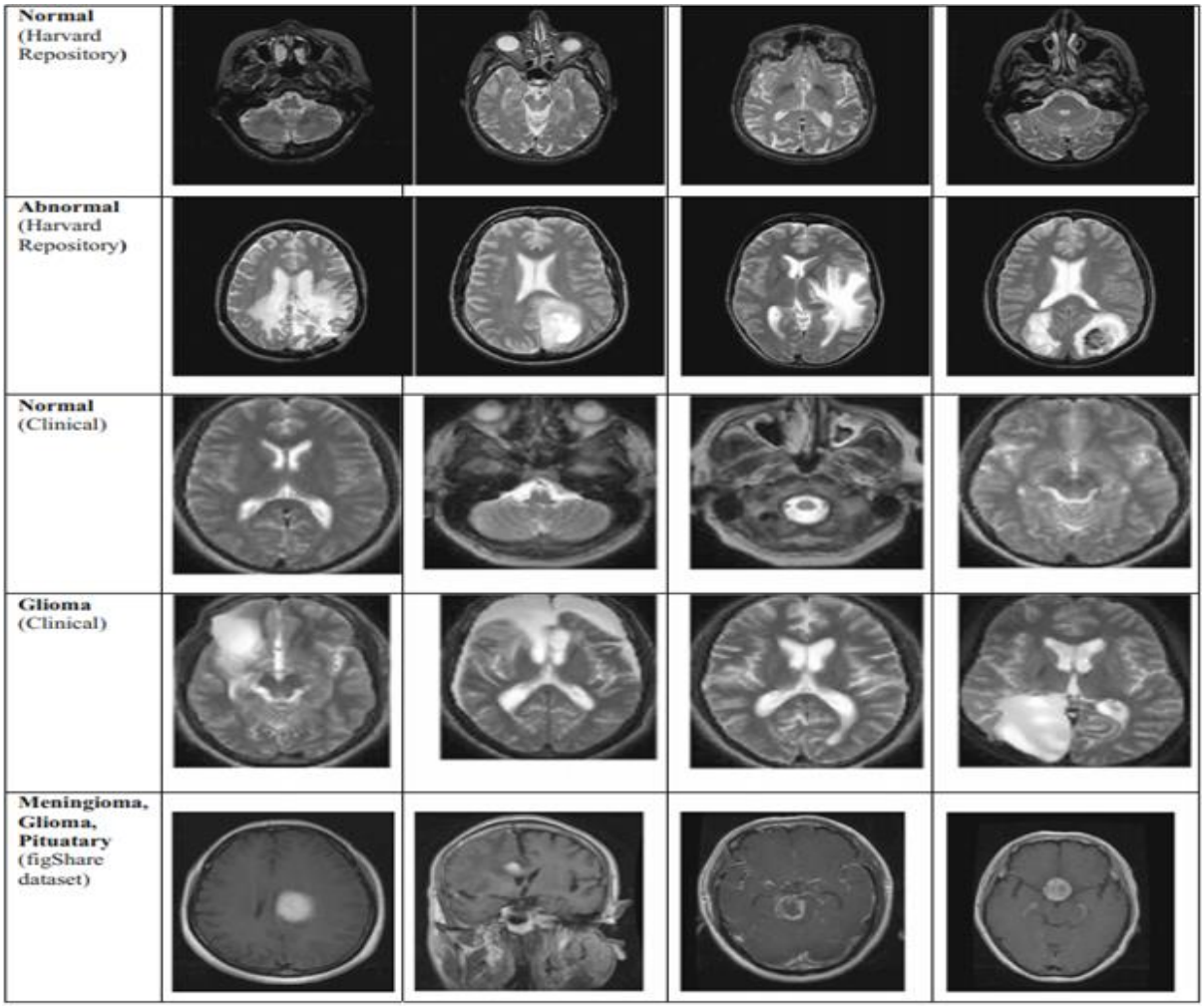


Fig.1.1. Sample normal and abnormal brains from the Harvard repository. Clinical dataset and Figshare dataset

Challenging medical data, experimentations were also done on the dataset from Figshare. It consists of 3064 brain MR images of 233 patients having glioma (1426), meningioma (708), and pituitary tumors (930). T1-CE images were available in axial, coronal, and sagittal views. The images were available in. mat format with size as 512×512. Images were resized to match the dimensions of the input layer of the pre-trained model. Since the images were grayscale, additional channels were created by replicating the pixel values three times. The sample MR images are shown in Fig.1.1.

1.9.2 Dataset 2

In this research, we used a publicly available CE-MRI dataset (Cheng, 2017) available at (https://figshare.com/articles/dataset/brain_tumor_dataset/1512427). The proposed brain tumor classification is based on two-dimensional images (2D slices), not 3-D volume, because in most clinical practice, the acquired and available CEMRI images are 2-D slices with a large slice gape. Therefore, our classification system based on 2-D MR images for clinical application is practical. The dataset was collected during 2005–2010 from Nanfang Hospital, Guangzhou, China, and General Hospital, Tianjin Medical University, China. The dataset contains three types of tumors (i.e., glioma, meningioma, and pituitary tumor as shown in 2Fig.) from 233 patients with total 3064 images across the axial, coronal, and sagittal views, as shown in Fig. Table 1 shows the details of the CE-MRI dataset. The images in the dataset are provided in matrix form. The size of each image is 512 × 512 pixels, and the pixel size is 49 mm × 49 mm.

Details of the CE-MRI dataset

Tumor type	Number of patients	Number of MR images	MRI View	Number of MR images
Meningioma	82	708	axial	209
			coronal	268
			sagittal	231
Glioma	89	1426	axial	494
			coronal	437
			sagittal	495
Pituitary	62	930	axial	291
			coronal	319
			sagittal	320
Total	233	3064		3064

Z.N.K. Swati et al. / Computerized Medical and Graphics 75 (2019) 34-46

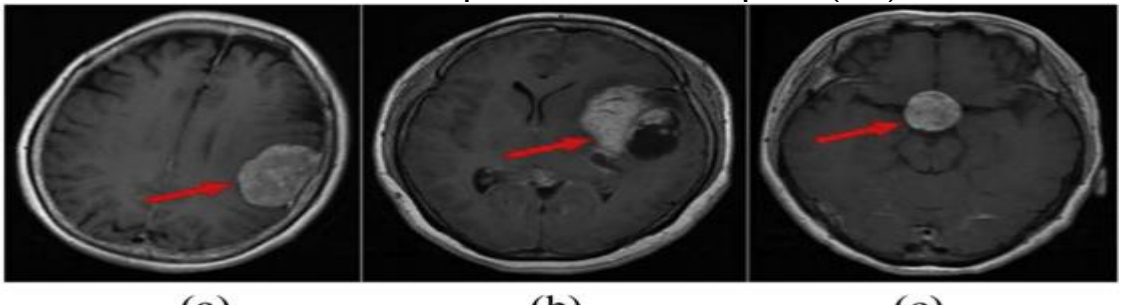


Fig.1.2. (a) and (b) are meningiomas with different appearance, and (c) is a pituitary tumor with an appearance

similar to that of (a).

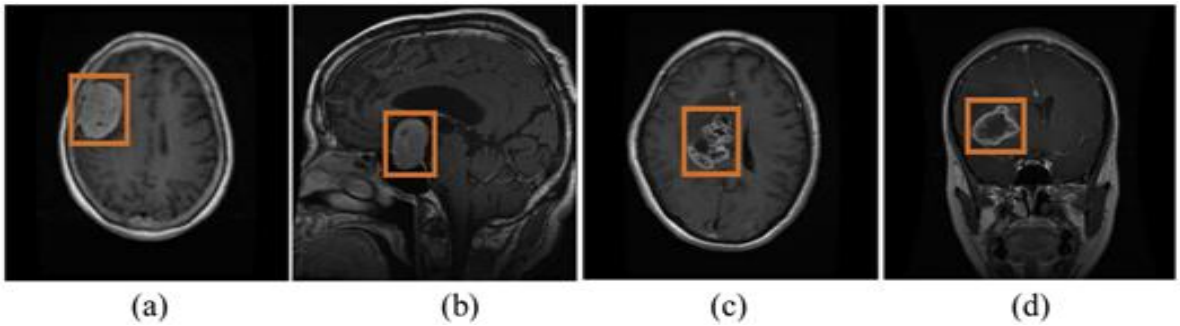


Fig.1.3. Four images of brain tumors in T1-weighted CE-MRI. The region inside the red rectangle contains a tumor. (a) Meningioma located near the skull, (b) pituitary tumor located near the sphenoidal sinus, (c) glioma containing edema and necrosis, and (c) glioma surrounded by edema.

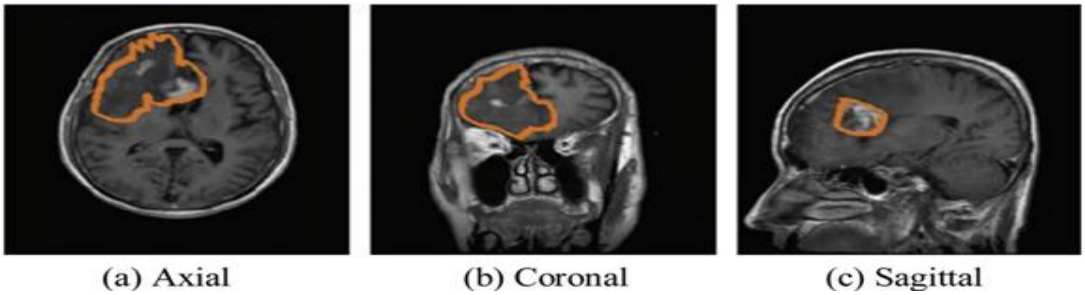
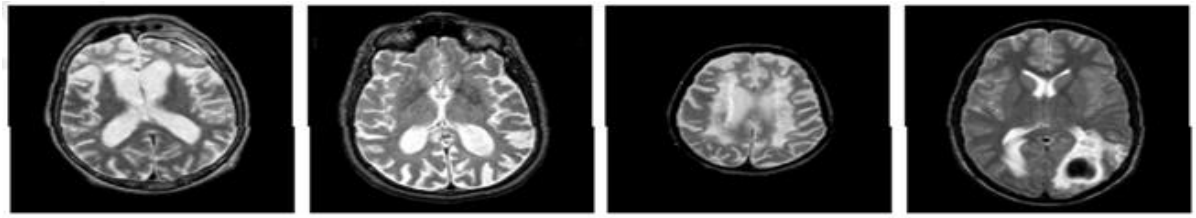


Fig.1.4. Three views of glioma in a patient. The area inside the yellow outline indicates a tumor.

1.9.3 Dataset 3

We obtained brain T2 weighted MRIs from the Whole Brain Atlas-Harvard Medical School (HMS) dataset, which is an open dataset. The slices are selected in axial orientation by experts of ten years' experience. The slices are resized in 256 × 256 pixels. The diseases of abnormal samples include AD, glioma, sarcoma, cerebral calcinosis, Huntington, etc. Originally, we obtained 177 pathological samples and only 28 normal samples, which is imbalanced. Firstly, 14 pathological and 14 normal samples were randomly selected to form the test set. Then, the rest 163 pathological and 14 normal samples from the training set. To balance the training set, resample was employed to generate 168 normal samples, i.e., the 14 normal samplings were copied for 11 times. So, we obtained a training set with 163 pathological and 168 normal samples at last. Figure presented some samples in our dataset.



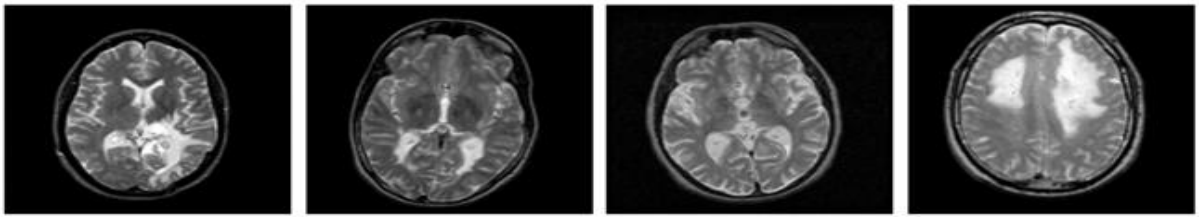


Fig.1.5. Pathological image samples

1.9.4 Dataset 4

Brain MR data set for evaluation of the projected work is acquired from The Cancer Imaging Archive (TCIA) Public Access repository (Clark et al., 2013). This assortment incorporates datasets from 20 patients with recently identified glioblastoma. We chose T1-weighted images as presented in Fig. It contains a sum of 696 MR images, of which 224 images are labeled as benign and the 472 are malignant images. The image size of every image is set to 225 × 225 in JPG/JPEG format.

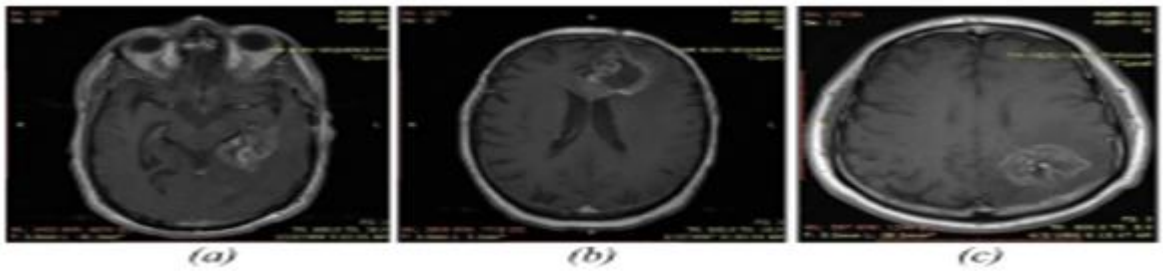


Fig.1.6. T1-weighted MR image

1.9.5 Dataset 5

The experiments were done on the publicly available AANLIB database of brain images from Harvard medical school .From this database total of 160 axial T2-weighted MR brain images with 140 abnormal, and 20 normal were taken. All the input images were having an in-plane resolution of 256 × 256. The MR images in the abnormal category have glioma, Alzheimer's disease, meningioma, Pick's disease, sarcoma, Metastatic bronchogenic carcinoma, Alzheimer's disease plus visual agnosia, Herpes encephalitis, and Huntington's disease. The sample normal and abnormal MR images are shown in Fig. The number of images corresponding to each disease category is given in Table 1. All the images in the disease category are treated as abnormal brains.

Abnormal MR images lying in each category

Sarcoma	12
Meningioma	9
Glioma	44
Picks	10
Huntington's	12
Alzheimer	7
Alzheimer's disease plus visual agnosia	20
Metastatic bronchogenic carcinoma	10
Herpes encephalitis	16

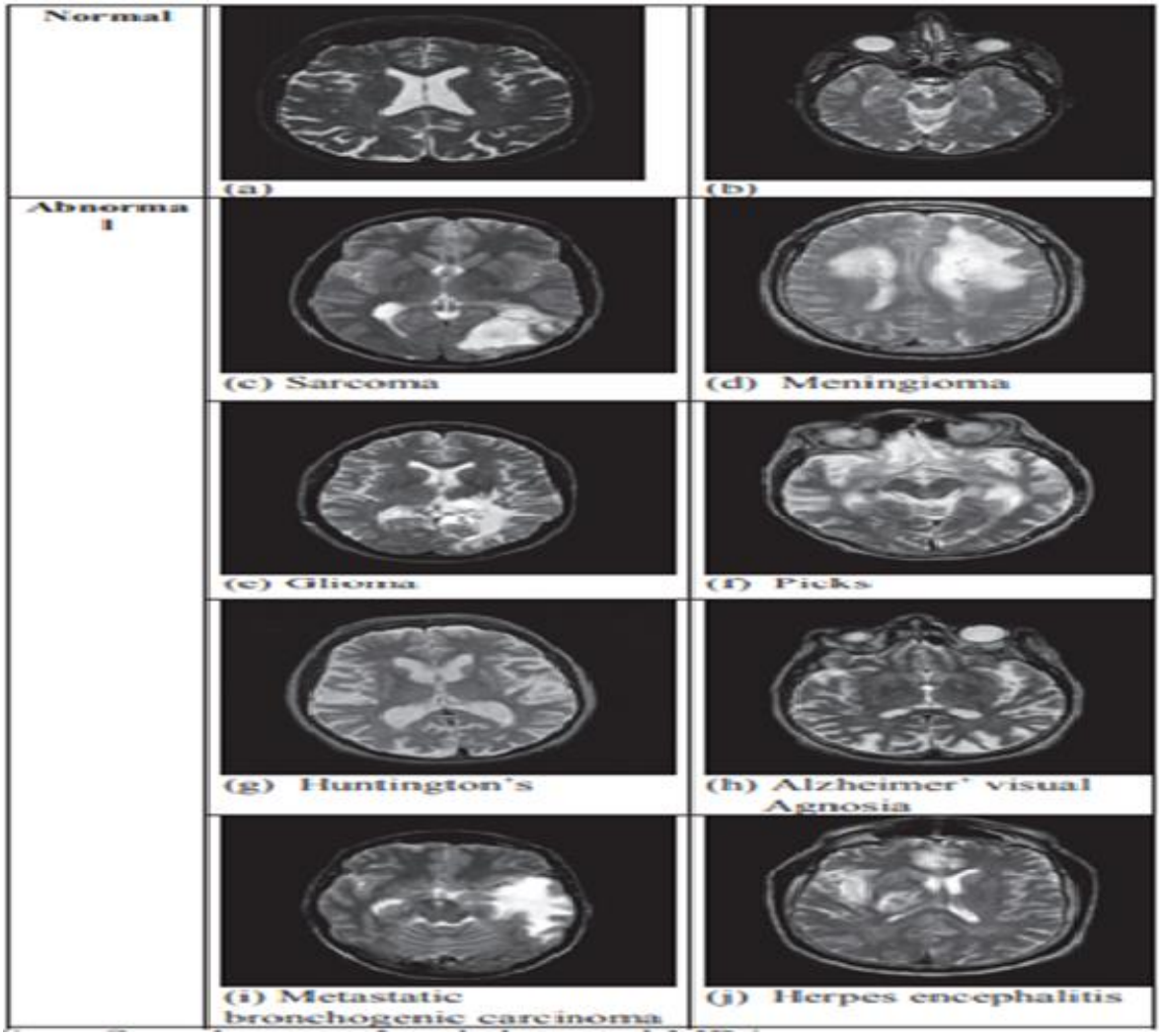


Fig.1.7. Sample normal and abnormal MR images

1.9.6 Dataset 6

Tumor database which has been collected in the year from 2005 to 2010, from Nanfang Hospital, Guangzhou, China, and General Hospital, Tianjin Medical University. The brain tumor dataset contained 3064 T1-weighted contrast-enhanced images from 233 patients with three kinds of brain tumor: meningioma (708 slices), glioma (1426 slices), and pituitary tumor (930 slices). The dimension of each MR slice was 512 × 512 pixels with a thickness and slice

gap of 6 mm and 1 mm respectively. Only 3049 out of 3064 images were used here because remaining 15 images were of lesser resolution of 256 × 256 pixels. The dataset also contained the tumor masks which was manually delineated and validated by three experienced radiologists from Medical College and Hospital Kolkata, India. One sample from each class of tumor is displayed in Fig.1.8 where the red line indicates tumor boundary. Fig.1.9. shows the Preprocessed images for the same.

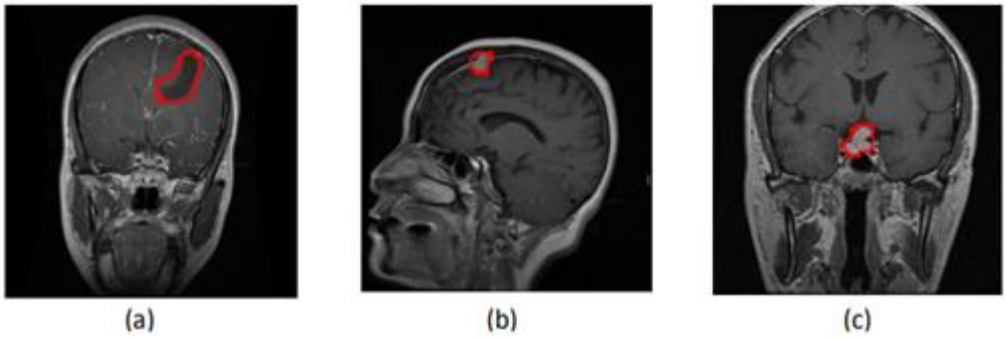


Fig.1.8. Brain MRI dataset sample of Glioma(a), Meningioma (b), pituitary tumor (c)

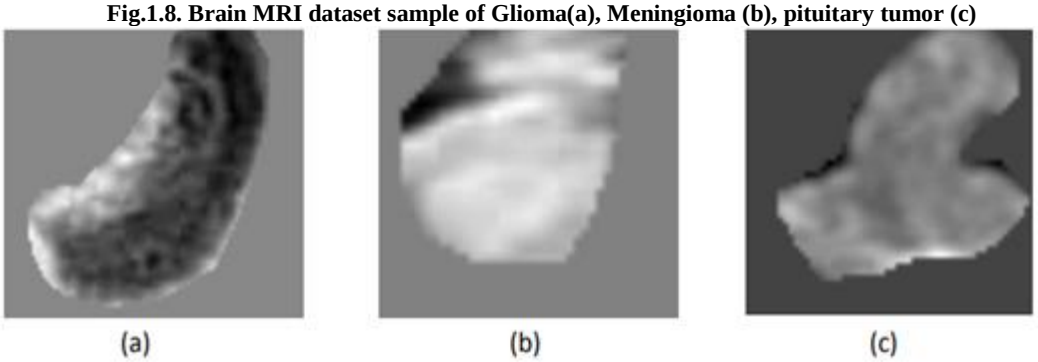


Fig.1.9. Preprocessed image of Glioma(a), Meningioma (b), Pituitary tumor(c)

1.9.7 Dataset 7

The performance of the brain tumor is implemented on the publicly available CE-MRI dataset. The dataset comprises of three different types of brain tumors, viz Meningioma, Glioma, Pituitary Tumor in three different views viz, Axial, Coronal and sagittal from 233 different patients. The images are T1 weighted contrast enhanced MRI used for analyzing the tumors. The detail of the dataset is given in Table

Table Details of CE-MRI Dataset

Tumor Type	No of Patients	Number of MRI	MRI View	Number of MRI
Meningioma	82	708	Axial	209
			Coronal	268
			sagittal	231
Glioma	89	1426	Axial	494
			Coronal	437
			sagittal	495
Pituitary Tumor	62	930	Axial	291
			Coronal	319
			sagittal	320a
Total	233	3064		3064

.

1.9.8 Dataset 8

Dataset used in this study consists of free accessible MR images categorized into two classes as normal and tumor . The images in the dataset were collected by field experts, such as doctors and radiologists and shared on the internet. The total number of images are 253 and each image was obtained from the volunteer patients. Therefore, the dataset has a heterogeneous structure. The number of tumor images is 155 while the number of normal samples is 98. The resolution of the images is not stable and the image quality is not high. The images have been converted to JPEG format. Sample images of the classes in the dataset are shown in Fig.1.10.

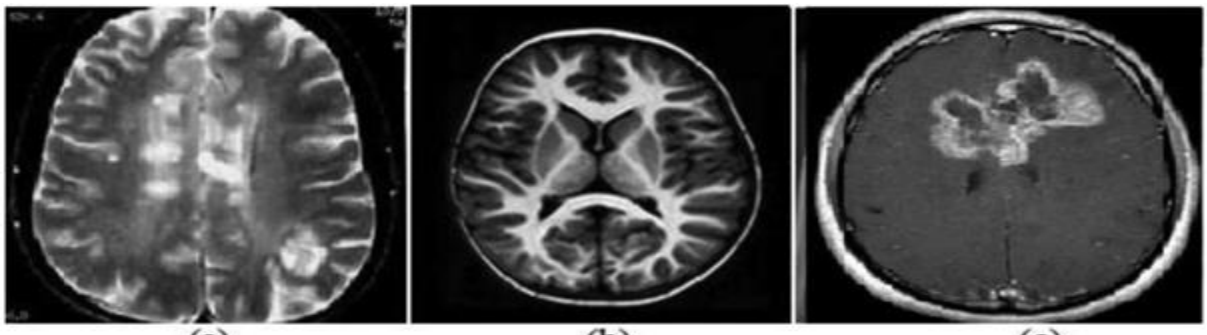


Fig. 1.10. (a) and (b) Normal class sample (c) tumor class sample.

1.9.9 Dataset 9

The data sets fed to training and testing procedure were collected from BRATS 2013 and BRATS 2015. It can be divided into two parts. High Grade Gliomas (HGG) data set includes 220 patients' MRI images and Low-Grade Gliomas data set includes 54 patients' MRI images.

Every patient has a group consisting 4 types of MRI brain images, T1, T2, T1c and FLAIR. Different from traditional RGB images

1.9.10 Dataset10

The dataset consists of open-access brain tumor MRI containing two classes of the tumor and normal (Chakrabarty, 2019). The samples belonging to the normal and tumor classes are illustrated in Fig. The dataset consists of 155 and 98 tumor and normal brain MRI, respectively. The dataset is heterogeneous MR images collected from 253 patients. The image in the dataset has not high resolution and each image has a different resolution. The image format is JPEG. In this study, the number of samples over the classes was balanced by increasing the number of normal brain tumor MRI so as to use the dataset more efficiently. In other words, the number of samples in the normal class was increased from 98 to 155 by using image augmentation methods. The augmented images were randomly selected from the original normal brain tumor MRI. In total, 310 (155 + 155) images were used for this study.

1.10 Project Organization

The project report is divided into four chapters. The organization of the project can be outlined as follows: Chapter 2 describes the different types of pre-processing and Feature Extraction techniques. Chapter 3 presents deep feature extraction and classification. Chapter 4 introduces the implementation and experimental results of the project.

2 Chapter 2. Pre-processing Techniques

2.1 A brief of Data preprocessing

Processing an image is very difficult task. Before any image can be processed, it is very significant to remove unnecessary items it may hold. After removing unnecessary artifacts, the image can be processed successfully. Image Pre-Processing is the first step of image processing. It is used to enhance the chances of identifying the relevant region Data preprocessing can be termed as a data mining technique which involves the transformation of raw data to a format which is more interpretable and makes the images more suitable for further processing. The following preprocessing steps The prime objective of the pre-processing is to improve the image data quality by suppressing undesired distortions (or) enhancing the required image features for further processing. The irrelevant data present in the image has been eliminated using the pre-processing technique. The pre-processing technique eliminates the incomplete, noisy and inconsistent data from the image in the training and test phase. In order to improve the quality of images taken from the CT-scan brain images and to make the feature extraction phase more reliable, pre-processing is necessary. CT-Scan brain images into normal, benign and abnormal.

Original image	Enhanced image

Fig1. Image Enhancement

2.2 Data Splitting

In this step, the complete dataset is divided into 3 segments, namely, Train, Test and Validation. Train data comprices the data sample that is used for fitting the model. Validation data is that sample of the data, which helps in providing an unbiased evaluation of the model which was trained on the training data along with tuning the hyper-parameters of the model. Test data includes those samples of data which provides an unbiased evaluation

2.3 Crop Normalization

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

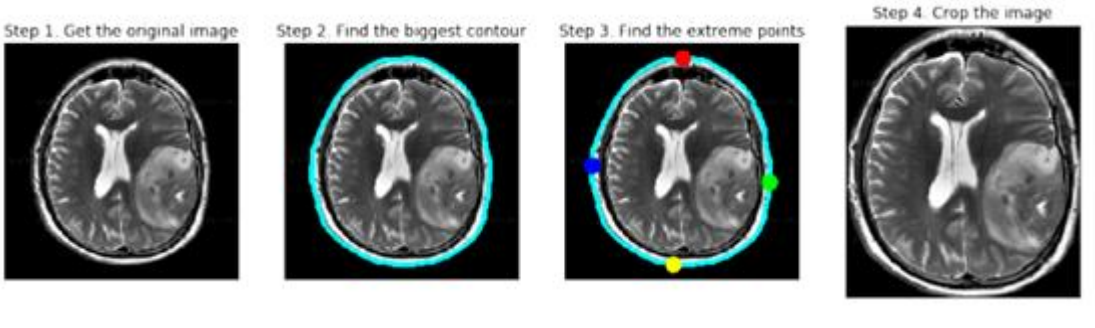


Fig2. finding extreme points in contours and clipping them

2.4 Resizing of images

The input dataset contains images with a different dimension and with different aspect ratio. Therefore, the images in the dataset are resized to a preset format, since the pretrained models used require the images to be 224×224×3 dimensions.

2.5 Median Filter

This is one of the most popular techniques used to remove noise. It is a non-linear filtration technology. This is used to remove salt and pepper noises from gray scale image. The average filter is based on the average pixel value. The advantages of the medium filter are that it is effective in reducing salt, pepper and splash noise. Also, the borders and edges are preserved. The main disadvantages are complexity and time consumption compared to the average filter. During the digitization process it is done by applying intermediate filtering techniques. Medium filtration is a non-linear process that is helpful in reducing Impulsive noise. Also useful for preserving edges in the image With random noise reduction. Impulsive noise can be caused by a Random bit error in communication channel. In a medium filter, A.The window slides along the image, the average value of the intensity The pixels inside the window become the intensity of the output file Pixel is being processed. Here a 3x3 medium filter is used.

2.6 Wiener Filter

Wiener filter is also a de-noising filter that is based on inverse filtering in frequency domain. Efficient for removing blurring effect form images is the main advantage of wiener filter. Due to working in frequency domain, it has low speed and it is not suitable for speckle noise.

2.7 Hybrid Filter

Hybrid filter is a combination of median and wiener filter. It can remove speckle noise, impulse noise and blurring effects from images. But the main disadvantage of hybrid filter is it is very complex and time consuming.

2.8 Modified Hybrid

Median Filter This filter is also a de-noising filter that is a combination of mean and median filter. It is very efficient for removing speckle noise, salt and pepper noise and Gaussian noise. But the main disadvantage of this filter is computation time is more as compared to simple median filter.

2.9 Morphology Based De-noising

This filter is based on Morphological opening and closing operations. It is very efficient and producing results better than other de-noising filters.

2.10 Morphological Opening

An opening is erosion followed by dilation with the same structuring element: $A \circ B = (A \ominus B) \oplus B$ Remember that erosion finds all the places where the structuring element fits inside the image, but it only marks these positions at the origin of the element. By following erosion by dilation, we “fill back in” the full structuring element at places where the element fits inside the object. So, an opening can be considered to be the union of all translated copies of the structuring element that can fit inside the object. Openings can be used to remove small objects, protrusions from objects, and connections between objects

2.11 Power law Transformation

Image enhancement is a very basic image processing task that defines us to have a better subjective judgment over the images. Image enhancement simply means, transforming an image f into image g using T. Where T is the transformation. The values of pixels in images f

and g are denoted by rand s, respectively. As said, the pixel values r and s are related by the expression, s = T(r) Where T is a transformation that maps a pixel value r into a pixel value s. The nth power and nth root curves shown in fig. A can be given by the expression, s = crγ This transformation function is also called as gamma correction. For various values of γ different levels of enhancements can be obtained. This technique is quite commonly called as Gamma Correction. Using the image negation formula given above, it is not necessary for the results to be mapped into the grey scale range [0, L-1]. Output of L-1-r automatically falls in the range of [0, L-1]. But for the Log and Power-Law transformations resulting values are often quite distinctive, depending upon control parameters like λ and logarithmic scales. So the results of these values should be mapped back to the grey scale range to get a meaningful outpu

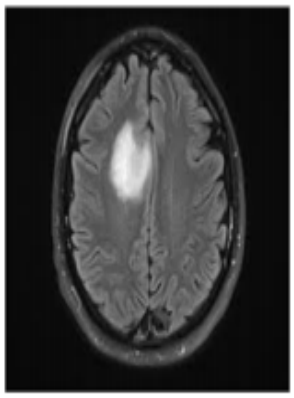


Fig1.MRI

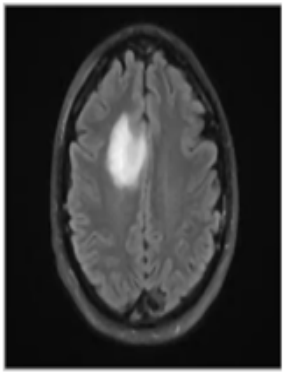


Fig2.Median Filtering

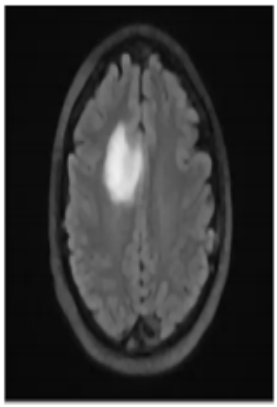


Fig3.Morphological Opening

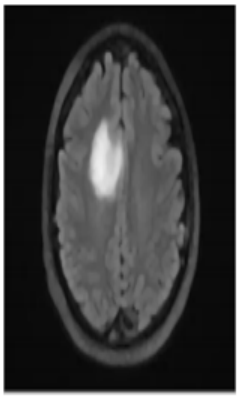


Fig4.power law Transformation

Fig3. We show in pictures how the images were processed in 1.5,1.9,1.10,1.11

2.12 ROI segmentation

A region of interest (ROI) is a subset of an image or a dataset identified from the original samples for a specific purpose. The boundary of a brain tumor in T1 weighted MRI slices is the ROI in this case. The annotated masks of the tumors, which contained labels '1' for tumor region and '0' for everything else, were provided in the brain tumor database. The exact tumors were extracted from the brain MRI samples by multiplying them pixel wise with the corresponding masks. Due to the variability of tumor sizes in the samples, tumor ROI images were resized and added zero padding to fit into the particular input shape for the proposed model. The dimension of each of the image after ROI segmentation was 256 × 256 pixels.

2.13 Intensity Zero-centering

Intensity zero-centering (or mean-centering) is a transformation of the data linearly which shifts the data so that it is centered at the origin. Generally, because of zero-centering, non-

linear operations and analysis perform well. Zero-centering to each of the ROI segmented dataset images was performed because it is much easier to understand variation when you are centered at the origin. This process was done by subtracting the mean of all the pixels of an image from all the pixels of that image.

2.14 Intensity Normalization

Generally, MRI intensities are acquired in arbitrary units; thus, it becomes difficult to compare images across scanners, subjects, and visits, even if the same procedure is used. Intensity normalization brings the intensities to a common scale across people. This affected the model's performance, prediction, and inference. It is an important step in any image analysis with more than one subject or time point to ensure comparability across images. Division of all pixel values of one image by their standard deviation was performed to get the normalized images.

2.15 Data Augmentation

Data augmentation is another way of Regularization which is used to reduce over fitting of models by an increase in the amount of training data using the present original information only. The idea of data augmentation is very old, and in fact, various data augmentation techniques have been applied to numerous problems. In this case, several combinations of transformations on the zero-centered, normalized, ROI segmented images of Glioma, Meningioma, and Pituitary tumor data were applied. The transformations were Flip, Rotate, Elastic transform and Shear with variable degrees of transformations, for example, 25 to 40 degrees of rotation, left-right, and top-bottom flip etc. Tensor Layer, a python library, is used as a framework for this purpose.

Data Augmentation is a strategy for artificially increasing the quantity and complexity of existing data. We know that training a deep neural network needs a large amount of data to fine-tune the parameters. But our dataset is very small, so we applied the technique of data augmentation on our training dataset by adding modifications to our images by making minor changes, such as flipping, rotation, and brightness. It will increase our training data size and our model will consider each of these small changes as a distinct image, and it will enable our model to learn better and perform well on unseen data. Figure displays the numerous augmented images from a single image.

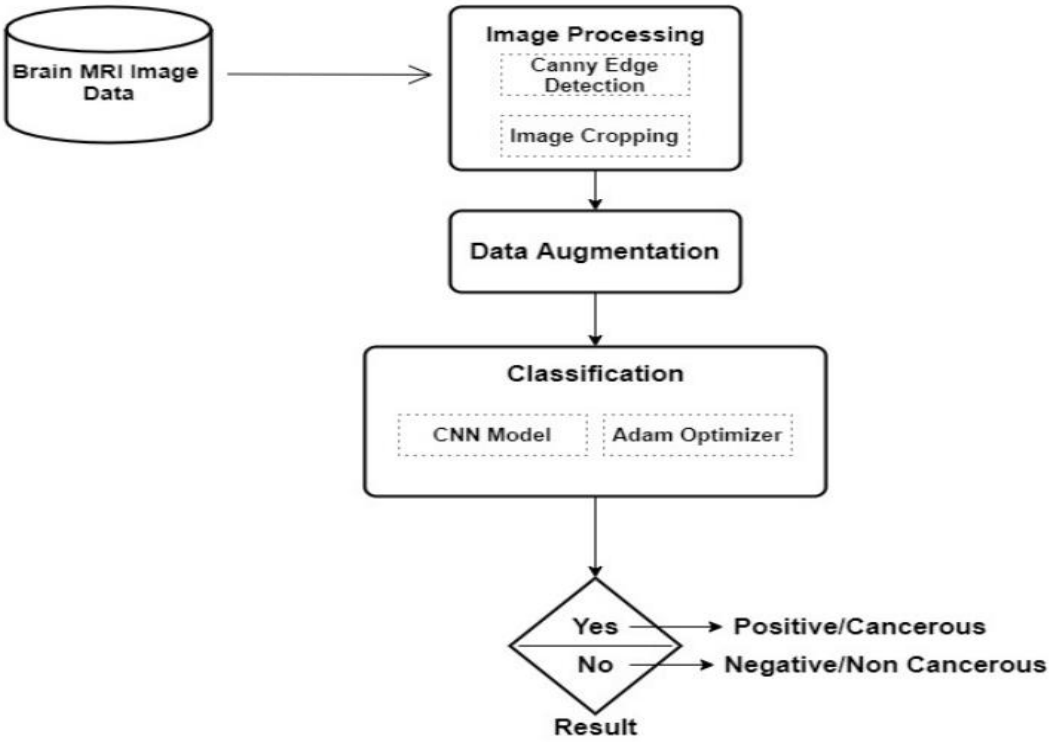


Fig4. Overview of proposed methodology.

Fig5. Brain MRI dataset sample of Glioma(a), Meningioma (b), Pituitary tumor (c)

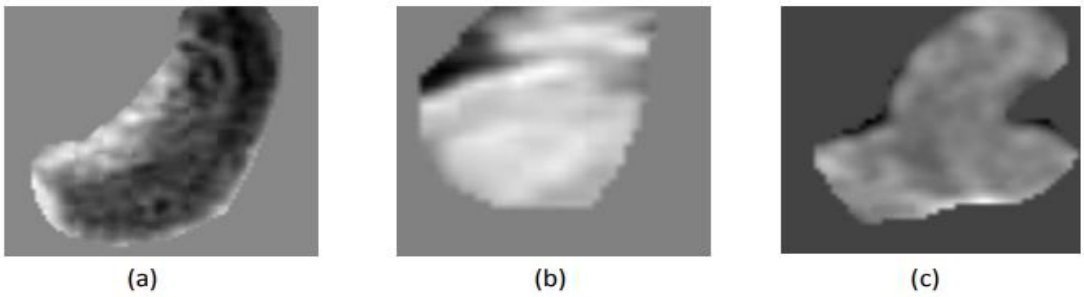


Fig6. Preprocessed image of Glioma(a), Meningioma (b), Pituitary tumor (c)

2.16 Image Re-sampling

Re-sampling is the process which converts the original image to a new image, by projecting, to a new coordinate system or altering the pixel dimensions. By applying geometric correction and translation, the net effect is that resulting redistribution of pixels involves their spatial displacements to new, more accurate relative positions. Re-sampling is commonly used to produce better estimates of the intensity values for individual pixels. An estimate of the new brightness value that is closer to the new location is made by some mathematical re-sampling technique. Three sampling algorithms are commonly used are, Nearest Neighbor technique, the transformed pixel takes the value of the closest pixel in the preshifted array. In the Bilinear Interpolation approach, the average of the intensity values for the 4 pixels surrounding the transformed output pixel is used. The Cubic Convolution technique averages the 16 closest input pixels; this usually leads to the sharpest image

2.17 Gray Scale Contrast Enhancement

The aim of contrast enhancement is to improve the interpretability or perception of information in images for preparing the image suitable for further processing like image understanding and interpretation. Contrast enhancement process is used to make the image brighter, to improve the visual details in the image. Contrast Enhancement is mainly categorized into two groups; they are direct methods and indirect methods. In the case of the direct method of contrast enhancement, a contrast measure is first defined, which is then modified by a mapping function to generate the pixel value of the enhanced image. On the other hand, indirect methods improve the contrast by exploiting the under-utilized regions of the dynamic range without defining the image contrast term. Indirect methods can further be divided into several subgroups

1. Decompose an image into high and lowfrequency signal e.g., homomorphism filtering,
2. Histogram modification techniques.
3. Transform-based techniques. Contrast stretching also known as normalization is a simple image enhancement technique that attempts to improve the contrast in an image by `stretching' the range of intensity values.

2.18 Noise Removal

Each imaging modality has many physical parameters that determine the visibility and sharpness of image. These are determined by spatial resolution and the clarity of boundaries. Both spatial resolution and contrast rendition are affected by noise. There are several de-

noising algorithms exists for noise removal each algorithm have its own advantage and disadvantage. Linear filters like Gaussian and wiener filters are conceptually simple, but I they degrade the details and the edges of the images. Therefore the denoised image would be blurred. Markov Random Field method is robust against noise and preserves the fine details in the image, but Markov random field algorithm implementation is complex and time consuming. In the case of high redundancy images, using non –local methods we can remove the noise but it eliminate nonrepeated details. Maximum likelihood estimation is another method of noise removal by adopting different hypothesis, but it does not retain the edge details.

2.19 Mathematical Operation

Mathematical Morphology based on set theory, which is applicable to binary images as well as grey scale images. Morphology is an image processing, technology that process images supported on shapes. There are four basic operations of mathematical morphology

1. Dilation
2. Erosion
3. Opening
4. Closing

Dilation is defined as the maximization value in the window. Hence, the image after dilation will be brighter or increased in intensity. It also expands the image and mainly used to fill the spaces. Erosion is just opposite to dilation. It is defined as the minimization value in the window. The image after dilation will be darker than the original image. It shrinks or thins the image. Opening and closing both parameters are formed by using dilation and erosion. In opening, the firstly image will be eroded and then it will be followed by dilation. In closing, the first step will be dilated and then the result of this is followed by erosion

2.19.1 Dilation Operation

Dilation operation is one of the bases of morphology processing. Dilation is the operation of “lengthening” or "thickening" in a binary image. This special way and the extent of thickening are controlled by structural elements. Mathematically, dilation is defined as set operation. A is dilated by B, written as A_B, is defined as $A \oplus B = \{z \mid (B^{\wedge}) \cap A \neq \varphi\}$ among them, φ is for the empty set, B is for the structure element, and $B^{\wedge}$ is for the reflection of collection B. In short, that A is dilated by B is the set composed of the origin positions of all structural elements. After mapping and translation, B at least has one overlap with A

2.19.2 Erosion operation

Erosion operation is also one of the bases of morphological processing. Erosion “shrinks” or “thins” the objects in the binary image. As in the dilation, the way to shrink and the extent is controlled by a structure element. The mathematical definition of erosion is similar to dilation A is eroded by B, recorded as AΘ B and defined as $A \ominus B = \{z \mid (B)z \subseteq A\}$ Among them, φ is for the empty set, B is for the structural element, and Ac is in the supplement of collection A. In another word, that A is eroded by B is the set composed of the origin positions of all structural elements, in which the background of translation B does not overlay on A’s

2.19.3 Operations of opening and closing

Opening operation generally makes the contour of objects smoother and disconnects narrow, discontinuous and removes thin protrusions. Similarly with opening operation, closing operation also makes an outline smooth, but the opposite is that it usually eliminates discontinuity and narrows long, thin gap, clears up small holes, and fill the ruptures of the contour line . $A \circ B = (A \ominus B) \oplus B$ As the same case with binary image, opening operation first using b to erode f plainly, and then using b to do dilate operation on the results obtained . Closing and opening operation Also, using B to do closing operation on A, expressed as A• B definite as $A \bullet B = (A \oplus B) \ominus B$ A• B is a complement of all translation union of B that do not overlap A. Pre-Processing techniques mainly aimed for the enhancement of the image without altering the information content in an image. In this paper, we implement a pre-processing method for the enhancement of brain tumor MRI image without altering image content and make suitable for further processing.

3 Chapter 3. FEATURE EXTRACTION

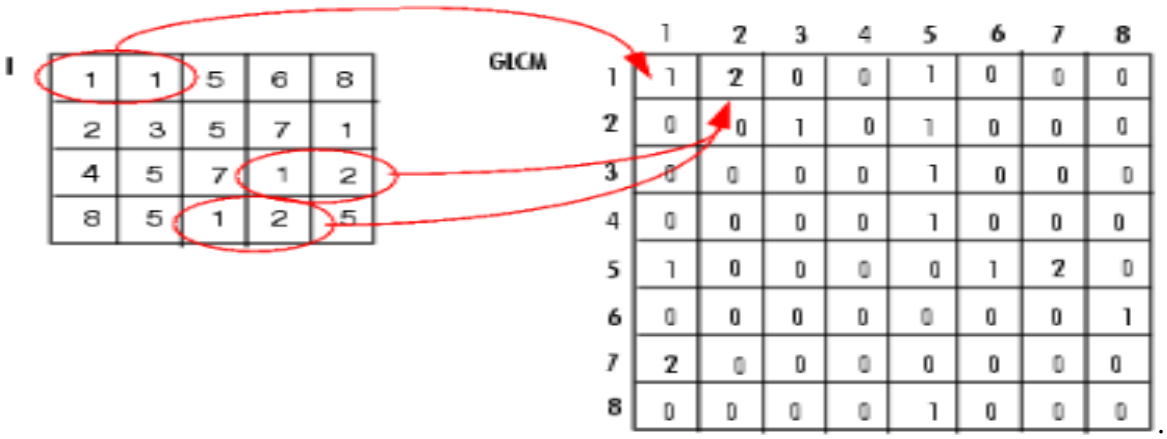
3.1 INTRODUCTION

In order to classify pictorial data, it is required to identify important features present in images that lead to categorization. Such features could be grouped into basic and complex features . Basic quantitative features may include average, variance, correlation, contrast, entropy, sum entropy, difference entropy, energy, homogeneity, kurtosis, inverse difference moment, kth moment and inertia. Using basic features, a variety of derived features could be extracted. Multiple algorithms have been developed to extract high-value complex features . In this study, we have grouped complex features into three major groups. There may be more candidate features for a particular domain but we have listed only those which are used in research under review.

Fig. 4.1. AI: Artificial Intelligence ML, NN, DL, and Spinking Neural Networks (SNN)

3.2 Gray Level Co-occurrence Matrix (GLCM)

A statistical method of examining texture that considers the spatial relationship of, also known as the gray-level spatial dependence matrix. The GLCM functions characterize the texture of an image by calculating how often pairs of pixel with specific values and in a specified spatial relationship occur in an image, creating a GLCM, and then extracting statistical measures from this matrix



From the GLCM , the texture features are extracted and described as follows:

3.2.1 Contrast (C)

Contrast of the image gives a measure of sudden change in intensity values in image,It is expected to be low if the intensity values of the pixel are similar

$$f_2 = \sum_{n=0}^{N_g-1} n^2 \left\{ \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} p(i,j) \right\} \qquad (1)$$

where Ng is the number of gray levels in the original image.

3.2.2 Autocorrelation

the autocorrelation feature of an image is used to estimate the fineness or roughness of the texture existing in the image. If the texture is coarse or unsmooth, then the autocorrelation function will bring down slowly, if not it will bring down very rapidly. For normal textures, the autocorrelation function will show peaks and valleys.

$$\text{Autocorrelation} = \sum_i \sum_j (p_x - \mu_x)(p_y - \mu_y)/\sigma_x \sigma_y \qquad (2)$$

3.2.3 Correlation (S)

It is a measure of correlation of a pixel to its neighbor within the selected region in the image. The statistical relationship between the two variables is denoted by this feature .

$$S = \sum_{ij} \frac{(i-\mu_i)\left(j-\mu_j\right) \cdot P_{\vec{o}}(i,j)}{\sigma_i \sigma_i} \qquad (3)$$

3.2.4 Cluster **prominence**

Cluster prominence is a measure of asymmetry. If the image is less symmetric, then the value of cluster prominence is high.

$$\text{Cluster prominence} = |\sum_i \sum_j (i + j - \mu_x - \mu_y)^4 p(i, j) \quad (4)$$

$$\text{Cluster shade} = \sum_i \sum_j (i + j - \mu_x - \mu_y)^3 p(i, j) \quad (5)$$

$$\text{Dissimilarity} = \sum_i \sum_j p(i, j) |i - j| \quad (6)$$

3.2.5 Energy (E)

Energy can be illustrated as the measure of the extent of pixel pair repetitions. It measures the homogeneity or uniformity of an image. When pixels are identical, the energy value will be huge.

$$\text{Energy} = \sum_i \sum_j p(i, j)^2 \quad (7)$$

3.2.6 Homogeneity (H)

Homogeneity is the measure which grows with lower contrast.

$$\text{Homogeneity} = -\sum_i \sum_j \frac{1}{1+i-j} p(i, j) \quad (8)$$

3.2.7 Angular Second Moment (ASM)

The summation of the squares of gray levels of the image is known as angular second moment. It is also named as energy. The energy is usually high when the intensity values are unequal.

ASM is a measure of homogeneity of an image. A homogeneous scene will contain only a few gray levels, giving a GLCM with only a few but relatively high values of P(i, j). Thus, the sum of squares will be high

$$f_1 = \sum_i \sum_j \{p(i, j)\}^2 \quad (9)$$

3.2.8 Coarseness (*Cness*):

Coarseness is a measure of roughness in the textural analysis of an image. For a fixed window size a texture with a smaller number of texture elements is said to be more coarse than the one with a larger number. The rougher texture means higher coarseness value. Fine textures have smaller values of coarseness. It is defined as:

$$C_{ness} = \frac{1}{2^{m+n}} \sum_{x=0}^{m-1} \sum_{y=0}^{n-1} f(x, y). \quad (10)$$

3.2.9 Structured Similarity Index (SSIM):

The Structural Similarity Index (SSIM) is a perceptual metric that signifies that the degradation in image quality may be caused by data compression or losses in data transmission or by any other means of the image processing. It is defined as:

$$\text{SSIM} = \left(\frac{\sigma_{xy}}{\sigma_x \sigma_y}\right)\left(\frac{2\overline{xy}}{(\overline{x}^2)+(\overline{y}^2)+C_1}\right) \cdot \left(\frac{2\sigma_x\sigma_y}{(\sigma_x)^2+(\sigma_y)^2+C_2}\right). \quad \textbf{(11)}$$

A Higher value of SSIM indicates better preservation of luminance, contrast, and structural content.

3.3 Mean Square Error (MSE)

Mean square error is a measure of signal fidelity or image fidelity. The purpose of signal or image fidelity measure is to find the similarity or fidelity between two images by providing the quantitative score. When MSE is calculated, then it is assumed that one of the images is pristine original, while the other is distorted or processed by some means and it is defined as:

$$\text{MSE} = \frac{1}{M \times N}\sum\sum\left(f(x,y) - f^R(x,y)\right)^2 \quad \textbf{(12)}$$

3.3.1 Peak Signal-to-Noise Ratio (PSNR)

Peak signal-to-noise ratio is a measure used to assess the quality of reconstruction of processed image and it is defined as:

$$\text{PSBR in dB} = 20\log_{10}\frac{(2^n - 1)}{\text{MSE}} \quad \textbf{(13)}$$

Lower value of MSE and higher value of PSNR indicate better signal-to-noise ratio.

3.3.2 Dice Coefficient

Dice coefficient or dice similarity index is a measure of overlap between the two images and it is defined as:

$$\text{Dice}(A,B) = 2 \times \frac{|A_1 \wedge B_1|}{(|A_1| + |B_1|)} \quad \textbf{(14)}$$

Where $A \in \{0, 1\}$ is tumor region extracted from algorithmic predictions and $B \in \{0, 1\}$ is the experts ground truth. The minimum value of dice coefficient is 0 and the maximum is 1; a higher value indicates better overlap between the two images.

3.3.3 Dissimilarity

a measure of distance between pairs of objects (pixels) in the region of interest.

$$Dissimilarity = \sum_i \sum_j |i-j| p(i,j) \quad \textbf{(15)}$$

3.3.4 Intensity-Based Features (IBF)

First-order histogram gives distinct statistical properties such as four statistical moments of the intensity histogram of an image. These rely on specific pixel values and not on the inter action or co-occurrence of neighboring pixel values. The first-order histogram statistics features are *:*

3.3.5 Mean (m):

The mean (m) is defined as the sum of the intensity values of pixels divided by the number of pixels in the SROI of an image. Mean represents the first central moment and indicates the average level value of (Segmented regions of interest (SROI)) where I (i, j) is the pixel intensity at position (i, j) and N represents the total number of pixels in a tumor region.

$$m = \frac{1}{N} \sum_i \sum_j I(i,j) \quad \textbf{(16)}$$

3.3.6 Variance

This parameter calculates the heterogeneity which strongly correlated with the standard deviations. It describes the circulation of the calculated gray level values. If the gray level values of the means are differed then the variance increased.

$$f_4 = \sum_i \sum_j (i-\mu)^2 p(i,j) \quad \textbf{(17)}$$

3.3.7 Standard Deviation (Std)

It is defined as the second central moment which describes probability distribution of observed population and it is a measure of homogeneity characteristics of an image. A high value represents better intensity level whereas low value indicates small intensity level.The standard deviation is used to describe the spread of gray level around the mean.

$$\text{Std} = \sqrt{\frac{\sum_i \sum_j (I(i,j)-m)^2}{N}} \quad \textbf{(18)}$$

3.3.8 Skewness

The third central moment gives the skewness of the intensity distribution. It is the measure of symmetry and asymmetry of the gray level values around the mean.

The measure of symmetry is given by this textural feature.

$$f_7 = \frac{1}{\sigma^3} \sum_i \sum_j (p(i,j)-\mu)^3 \quad \textbf{(19)}$$

3.3.9 Kurtosis

The forth central moment gives kurtosis. It gives the measure of closeness of an intensity distribution to the normal Gaussian shape. Kurtosis is a measure of whether the data are peaked or flat relative to the normal distribution.

$$f_8 = \frac{1}{\sigma^4}\sum_i\sum_j (p(i,j)-\mu)^4 - 3 \quad \textbf{(20)}$$

3.3.10 Local Homogeneity, Inverse Difference Moment (IDM)

IDM measures the local homogeneity of an image. The incidence of co-occurrence of pixel pairs is enhanced when they are close in gray-scale value and thus increases the IDM value. Because of the weighting factor (1 + (i − j)2)−1, it will get small contributions from inhomogeneous areas $i \neq j$. The result is a low IDM value for inhomogeneous images, and a relatively higher value for homogeneous images.

$$IDM = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1}\frac{1}{1+(i-j)^2}P(i,j) \quad \textbf{(21)}$$

3.3.11 Entropy

Entropy represents the amount of randomness in gray level intensity distribution of an image and corresponds to the grey levels which the individual pixels can adopt.

Suppose the images are dissimilar then values in the co_occurance matrix will be very small values, It implies that the value of the entropy is very high. Energy and Entropy both are inversely proportional to each other. The entropy is expected to be high if the gray levels are distributed randomly throughout the image.

$$f_6 = -\sum_i\sum_j p(i,j)\log(p(i,j)). \quad \textbf{(22)}$$

3.4 Rotation Invariant Circular Gabor Features (RICGF):

Gabor filter is a Gaussian kernel function modulated by a radially sinusoidal surface wave; therefore, it gives rotational invariant texture features which are given by:

$$g(x,y,\lambda,\theta,\psi,\sigma) = \exp\left(-\frac{D(x,y)^2}{2\sigma^2}\right).\cos\left(2\pi\frac{D(x,y)}{\lambda}+\psi\right)$$

$$\text{where } D(x,y) = \sqrt{(x-\bar{x})^2+(y-\bar{y})^2} \quad (23)$$

Here $(\bar{x},\bar{y})$ are coordinates of the center of Gaussian part in the filter λ l represents the wavelength of the sinusoidal factor σ is Gaussian width, and ψ is the phase offset of the sinusoidal factor. Similar to directional Gabor filters, five values of l viz. (2√2, 4, 4√2, 8, and 8√2) and two values of ψ, i.e. 0° and 90° are considered there by retrieving 10 features . Mean, standard deviation, skewness, and kurtosis are the four statistical parameters which are extracted for each filter output in the marked SROI and are taken as 40 features in the feature bank. The intensity and texture features summary is given in

Table Summary of intensity and texture features

Feature category	Features	Number of features
LoG	Four statistical parameters for the LoG filter output in the SROI region are retrieved at σ=0.25, 0.50, 1, and 2 thereby contributing 16 features in the feature pool. These parameters are: (1) mean intensity, (2) standard deviation, (3) Skewness, (4) Kurtosis	
GLCM	Following GLCM features at 0°, 45°, 90°, and 135° are calculated: (1) contrast, (2) homogeneity, (3) correlation, (4) energy	4×4=16
RILBP	For LBP, LBP-range, and LBP standard deviation filter output for radii-1, 2, and 4 pixels are calculated. These features are: (1) mean intensity, (2) standard deviation, (3) Skewness, (4) Kurtosis	4×3×3=36
IBF	Following features for the histogram of an image and range filter output within the SROI are calculated: (1) mean intensity, (2) standard deviation, (3) Skewness, (4) Kurtosis, (5) entropy	5+5=10
DGTF	The λ for 2√2, 4, 4√2, 8, 8√2) and θ for 0°, 22.5°, 45°, 67.5°, and 90° are varied. Four statistical parameters are calculated for each filter output in the marked SROI and are taken as 100 features in the feature bank. These parameters are: (1) mean intensity, (2) standard deviation, (3) Skewness, (4) Kurtosis	25×4=100
RICGF	RICGFs are calculated at λ=2√2, 4, 4√2, 8, 8√2) and two values of ψ, i.e., 0° and 90° four statistical parameters for each filter output in the marked SROI and are taken as 40 features in the feature bank. These features are: (1) mean intensity, (2) standard deviation, (3) Skewness, (4) Kurtosis	10×4=40
Total number of features:10+16+16+100+40+36=218		

3.5 Edge detection

The edge representation of an image significantly reduces the quantity of data to be processed, yet it retains essential information regarding the shapes of objects.

The major property of the edge detection technique is its ability to extract the exact edge line with good orientation

Edge detection is a fundamental tool for image segmentation. Edge detection methods transform original images into edge images benefits from the changes of grey tones in the image.

Edges are local changes in the image intensity. Edges typically occur on the boundary between two regions. The main features can be extracted from the edges of an image.

Edge detection is an active area of research as it facilitates higher level image analysis. There are three different types of discontinuities in the grey level like point, line and edges. Spatial masks can be used to detect all the three types of discontinuities in an image.

3.5.1 First order edge detection or gradient based edge operator

the first order derivative operators are very sensitive to noise and produce thicker edges. It is based on the use of a first order derivative, or gradient based, If I (i, j) be the input image, then image gradient is given by following formula :

$$\nabla I(i,j) = \hat{i}\frac{\delta I(i,j)}{\delta i} + \hat{j}\frac{\delta I(i,j)}{\delta j} \qquad (24)$$

where; $\frac{\delta I(i,j)}{\delta i}$ is the gradient in the i direction

$\frac{\delta I(i,j)}{\delta j}$ is the gradient in the j direction

The gradient magnitude can be computed by the formula:

$$|G| = \sqrt{(\frac{\delta I}{\delta i})^2 + (\frac{\delta I}{\delta j})^2} \qquad (25)$$

$$\text{Or} \quad |G| = \sqrt{G_i^2 + G_j^2} \qquad (26)$$

The gradient direction can be computed by formula:

$$\theta = \arctan\left(\frac{G_j}{G_i}\right) \qquad (27)$$

The magnitude of the gradient above gives edge strength and the gradient direction is always perpendicular to the direction of edge.

3.5.1.1 Robert operator

The Roberts edge detection is introduced by Lawrence Roberts (1965). It performs a simple, quick to compute, 2-D spatial gradient measurement on an image. This method emphasizes regions of high spatial frequency which often correspond to edges. Gives no information about edge orientation

This operator computes the sum of the squares of the difference between diagonally adjacent pixels through discrete differentiation and then calculate approximate gradient of the image. The input image is convolved with the default kernels of operator and gradient magnitude and directions are computed.

$$D_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \text{ And } D_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \qquad \textbf{(28)}$$

3.5.1.2 *Sobel* operator

The Sobel edge detection method is introduced by Sobel in 1970 ،The image is processed in the X and Y directions separately first, and then combined together to form a new image which represents the sum of the X and Y edges of the image. However, these images can be processed separately as well.

When using a Sobel Edge Detector, it is first best to convert the image from an RGB scale to a Grayscale image. Then from there, we will use what is called kernel convolution. A kernel is a 3 x 3 matrix consisting of differently (or symmetrically) weighted indexes.

X – Direction Kernel

-1	0	1
-2	0	2
-1	0	1

Y – Direction Kernel

-1	-2	-1
0	0	0
1	2	1

Sobel Operator uses a 3×3 mask and applied on part of the image. Given an image f (x, y), its gradient along x and y-axis are calculated according to (29) and (30).

$$g_x = \frac{\delta f}{\delta x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) \quad \textbf{(29)}$$

$$g_y = \frac{\delta f}{\delta y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7) \quad \textbf{(30)}$$

Then the gradient of image is defined as:

$$\nabla f(x,y) = \frac{\delta f}{\delta x}\hat{\imath} + \frac{\delta f}{\delta y}\hat{\jmath} = g_x\hat{\imath} + g_y\hat{\jmath} \quad \textbf{(31)}$$

where, $\hat{\imath}$ & $\hat{\jmath}$ are unit vectors along x and y axis respectively. The magnitude of gradient is given by,

$$g(x,y) = |\nabla f(x,y)| = \sqrt{{g_x}^2 + {g_y}^2} \quad \textbf{(32)}$$

3.5.1.3 *Prewitt* operator

The Prewitt edge detection is proposed by Prewitt in 1970 It is a gradient-based operator. Prewitt operator is similar to the Sobel operator and is used for detecting vertical and horizontal edges in images.

It is one of the best ways To estimate the magnitude and orientation of an edge. It computes the gradient approximation of image intensity function for image edge detection. At the pixels of an image, the Prewitt operator produces either the normal to a vector or the corresponding gradient vector. It uses two 3 x 3 kernels or masks which are convolved with the input image to calculate approximations of the derivatives one for horizontal changes, and one for vertical.

Prewitt detection is slightly simpler to implement computationally than the Sobel detection, but it tends to produce somewhat noisier results.

$$D_i = \begin{bmatrix} -1 & 0 & +1 \\ -1 & 0 & +1 \\ -1 & 0 & +1 \end{bmatrix} \text{ And } D_j = \begin{bmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

3.5.1.4 Kirsch Edge detection

Kirsch edge detection is introduced by Kirsch (1971). The masks of this Kirsch technique are defined by considering a single mask and rotating it to eight main compass directions: North, Northwest, West, Southwest, South, Southeast, East and Northeast:

$$E=\overset{k_0}{\begin{bmatrix}-3&-3&5\\-3&0&5\\-3&-3&5\end{bmatrix}}\quad NE=\overset{k_1}{\begin{bmatrix}-3&5&5\\-3&0&5\\-3&-3&-3\end{bmatrix}}\quad N=\overset{k_2}{\begin{bmatrix}5&5&5\\-3&0&-3\\-3&-3&-3\end{bmatrix}}\quad NW=\overset{k_3}{\begin{bmatrix}5&5&-3\\5&0&-3\\-3&-3&-3\end{bmatrix}}$$

$$W=\overset{k_4}{\begin{bmatrix}5&-3&-3\\5&0&-3\\5&-3&-3\end{bmatrix}}\quad SW=\overset{k_5}{\begin{bmatrix}-3&-3&-3\\5&0&-3\\5&5&-3\end{bmatrix}}\quad S=\overset{k_6}{\begin{bmatrix}-3&-3&-3\\-3&0&-3\\5&5&5\end{bmatrix}}\quad SE=\overset{k_7}{\begin{bmatrix}-3&-3&5\\-3&0&5\\-3&5&5\end{bmatrix}}$$

The edge magnitude is defined as the maximum value found by convolution of each mask with the image. The direction is defined by mask that produces the maximum magnitude. Example, mask k0 corresponds to a vertical edge, while mask k5 corresponds to a diagonal edge. Notice that the last four masks are actually the same as the first four, but flipped about a central axis

3.5.1.5 Robinson Edge detection

The Robinson method (Robinson 1977) is similar to Kirsch masks but is easier to implement because they rely only on coefficients of 0, 1 and 2. The masks are symmetrical about their directional axis, the axis with the zeros. One need only to compute the result on four masks and the result from other four can be obtained by negating the result from the first four. The masks are as follows:

$$E=\overset{r_0}{\begin{bmatrix}-1&0&1\\-2&0&2\\-1&0&1\end{bmatrix}}\quad NE=\overset{r_1}{\begin{bmatrix}0&1&2\\-1&0&1\\-2&-1&0\end{bmatrix}}\quad N=\overset{r_2}{\begin{bmatrix}1&2&1\\0&0&0\\-1&-2&-1\end{bmatrix}}\quad NW=\overset{r_3}{\begin{bmatrix}2&1&0\\1&0&-1\\0&-1&-2\end{bmatrix}}$$

$$W=\overset{r_4}{\begin{bmatrix}1&0&-1\\2&0&-2\\1&0&-1\end{bmatrix}}\quad SW=\overset{r_5}{\begin{bmatrix}0&-1&-2\\1&0&-1\\2&1&0\end{bmatrix}}\quad S=\overset{r_6}{\begin{bmatrix}-1&-2&-1\\0&0&0\\1&2&1\end{bmatrix}}\quad SE=\overset{r_7}{\begin{bmatrix}-2&-1&0\\-1&0&1\\0&1&2\end{bmatrix}}$$

The magnitude of the gradient is the maximum value gained from applying all eight masks to the pixel neighborhood, and the angle of the gradient can be approximated as the angle of the line of zeroes in the mask yielding the maximum response.

3.5.2 Second order derivative or Zero crossing

It is based on second order derivative, the Laplacian ∇^2 .In this operator a pixel is marked as an edge at a position where second derivative of an image becomes zero. The laplacian operator ∇^2 for a 2D image I (i, j) is defined by following formula:

$$\nabla^2 = I(i,j) = \frac{\delta^2}{\delta x^2} I(i,j) + \frac{\delta^2}{\delta y^2} I(i,j) \quad \textbf{(31)}$$

3.5.2.1 Marr-*Hildreth* operator

The Marr-Hildreth (1980) technique is a method of detecting edges in digital images that is continuous curves wherever there are well-built and fast variations in image brightness. It is a easy and it operates by convolving the image with the LoG function. Subsequently the zero-crossings are discovered in the filtered result to find the edges. The LoG method is sometimes as well referred to as the Mexican hat wavelet due to its image shape while turned up-side-down. Algorithm for the Marr-Hildreth edge detector is Smooth the image using a Gaussian Apply a two-dimensional Laplacian to the smoothed image (often the first two steps are combined into a single operation) Loop through the result and look for sign changes. If there is a sign change plus the slope across the sign change is greater than some threshold, mark as an edge.To get better results it is possible to run the result of the Laplacian through a hysteresis alike to Canny's edge detection.

3.5.2.2 Laplacian of Gaussian (LoG) operator

The Laplacian of Gaussian (LoG) was proposed by Marr(1982). The LoG of an image f(x,y) is a second order derivative defined as It has two effects, it smoothes the image and it computes the Laplacian, whch yields a double edge image. Locating edges then consists of finding the zero crossings between the double edges. The digital implementation of the Laplacian function is usually made through the mask below The Laplacian is generally used to found whether a pixel is on the dark or light side of an edge.The Marr-Hildreth edge detector was a very popular edge operator before Canny proposed his algorithm. It is a gradient based operator which uses the Laplacian to take the second derivative of an image. It works on zero crossing method. It uses both Gaussian and laplacian operator so that Gaussian operator reduces the noise and laplacian operator detects the sharp edges. The Gaussian function defined by the formula:

$$G(i,j) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp^{-\left(\frac{i^2+j^2}{2\sigma^2}\right)} \quad \textbf{(32)}$$

Where σ, is the standard deviation and the LoG operator is computed from

$$LoG = \frac{\delta^2}{\delta i^2} G(i,j) + \frac{\delta^2}{\delta j^2} G(i,j) \quad \textbf{(33)}$$

$$= \frac{i^2 + j^2 - 2\sigma^2}{\sigma^4} \exp^{-\left(\frac{i^2+j^2}{2\sigma^2}\right)}$$

The Marr–Hildreth operator, however, suffers from two main limitations. It generates responses that do not correspond to edges, so-called "false edges", and the localization error may be severe at curved edges.

3.6 Optimal Edge Detection

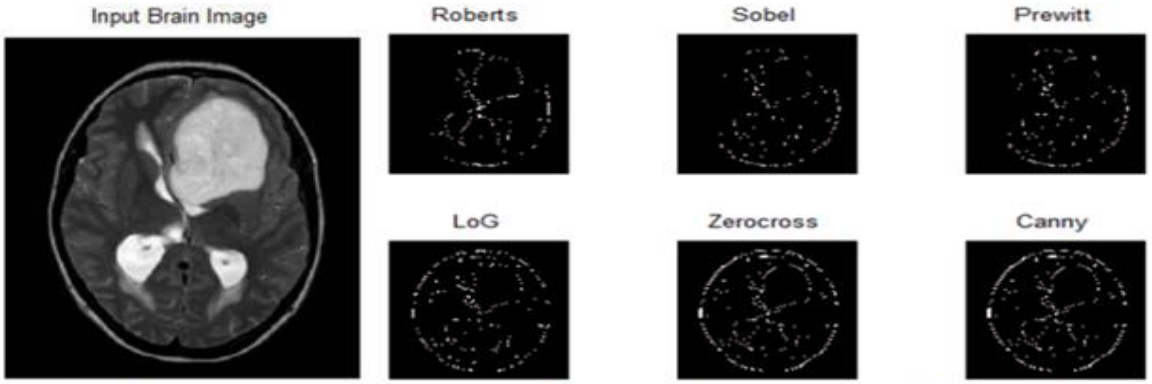
3.6.1.1 Canny Operator

It is a method to find edges by isolating noise from the image without affecting the features of the edges in the image and then applying the tendency to find the edges and the critical value for threshold.

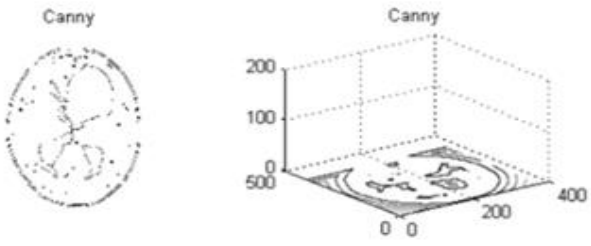
The canny edge detector first smoothens the image by application of Gaussian filter to smooth the MRI image for the removal of noise and finding the intensity gradient of the image. then it finds the image gradient to highlight regions with high spatial derivatives. After that it performs tracking along these regions and suppresses any pixel that is not at the maximum. The gradient array at this moment can further be reduced by hysteresis which is used to track along the remaining pixels that have not been suppressed.Hysteresis uses two thresholds and if the magnitude is below the first threshold, it is set to zero. If the magnitude is above the high threshold, it is made an edge .

Note:

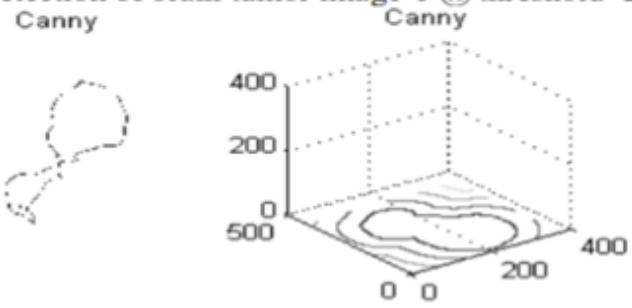
Non-maximum suppression is an edge thinning technique, with calculation of the gradient image, the edge that gets extracted from the gradient value is quite blurred. Hence a strong edge and weak edge concept, are used, the value to 0 for local minimal(weak edge) and 1 for local maximal(strong edge), which helps suppression of the gradient value that lies at zero (local minimal) and then the edge strength of the current pixel is compared with the edge strength of the pixel in the +ve and –ve gradient directions. After applying Non-Maximum suppression, edge pixel provides a spurious response. The value will be preserved, if edge strength of the pixel is larger (i.e local maximal) compared to the other pixels in the same direction. Then the local maximal region alone is considered as pixel, and the pixel gets clustered.



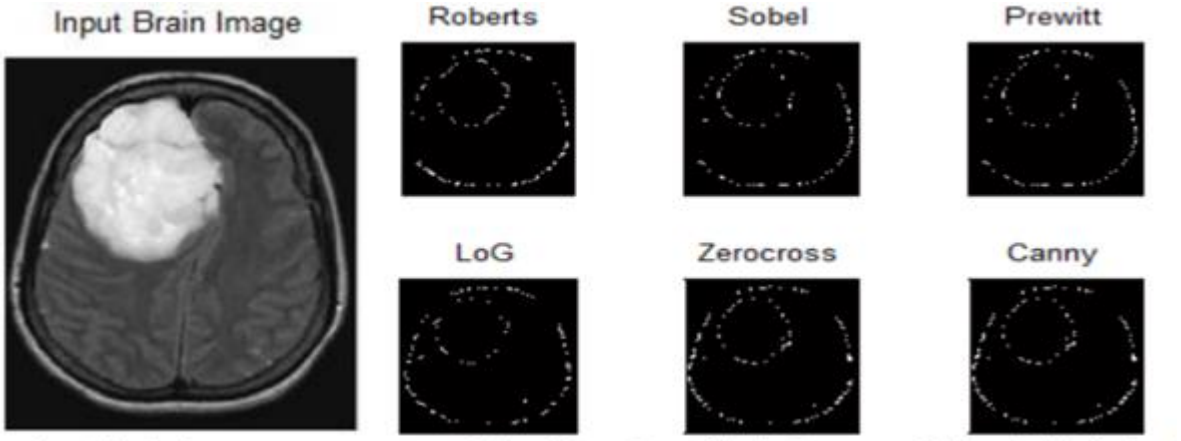
Input brain image Edge Detection of brain image @ threshold=0 pixel



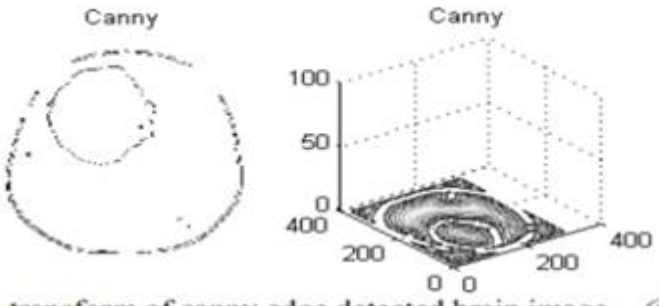
Edge Detection of brain tumor image-1 @ threshold=2000 pixel



Distance transform of canny edge detected brain tumor image @ threshold=2000 pixel

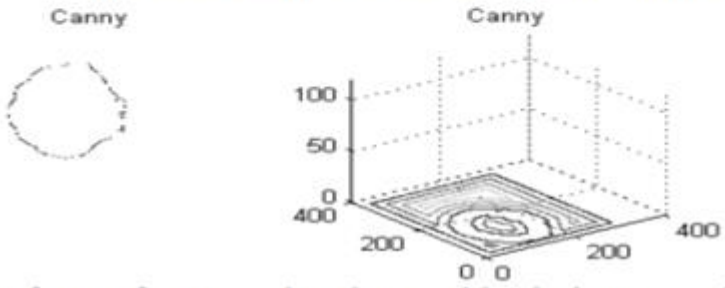


Input brain image Edge Detection of brain image @ threshold=0 pixel



Distance transform of canny edge detected brain image @ threshold=0

Edge Detection of brain tumor image @ threshold=2000 pixel



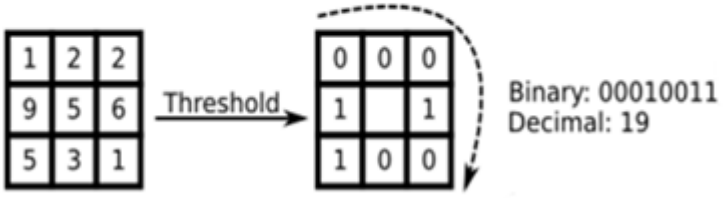
Distance transform of canny edge detected brain image @ threshold=2000

3.7 Local binary pattern

The local binary pattern is a simple, quick, and very efficient method for extracting feature from image texture. First a neighborhood of an image is selected. Then, the intensity of the points existing in this neighborhood is compared with the intensity of the pixel located in the center of the neighborhood and a binary code is considered for each pixel according to equation below. In order to make the algorithm rotation-invariant usually circular neighborhood is considered. The pixels whose coordinates are not exactly located on the neighborhood of pixel center are obtained by interpolation.In equation below, represents the number of neighborhood pixels of a considered center pixel, is the neighborhood radius, is the intensity of the neighborhood pixels, and is the intensity of the central pixel.

$$LBP_{P,R}=\sum_{i=0}^{P-1} s(g_i - g_c)2^i \qquad (34)$$

$$s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}$$



Binary code of each pixel in local binary patterns.

A measure called uniformity was proposed in the modified version of LBP [42]. A bit pattern is called uniform if it has at least two circular bit transitions from zero to one or vice versa. For instance,

the bit pattern (00000000) has a zero transition and it is uniform; whereas, the bit pattern (01010011) has six transitions and it is not uniform. In the uniform binary pattern mapping, there is a separate output label for each uniform pattern and all non-uniform patterns are assigned to a label. There are two reasons to remove non-uniform binary patterns and assign them to a label. First, it has been shown that most local binary patterns are uniform in natural images. The second reason is that considering uniform patterns instead of all possible patterns provides better detection results. There are signs that indicate uniform binary patterns have more robustness and less sensitivity to noise. In this method, each pixel is labeled by a code of the main texture, which is the best match with local neighbors. The main LBP operator was only defined with spatial information. Subsequently, it was extended for dynamic textures analysis . One of the methods in dynamic texture analysis considers each voxel in three orthogonal planes (XY-XY-YT). Here, X and Y are spatial coordination and T as the third dimensions represents time. More specifically, binary codes are extracted from all voxels in XY, YT, and XT planes and specified as XY-LBP, YT-LBP, and XT-LBP. Subsequently, histograms are computed and concatenated in a histogram. As we can see in figure below, each voxel is located at the intersection of the three orthogonal planes (XY, XT, and YT). In these planes, eight neighbors are delineated for each voxel. The local binary pattern for each voxel is separately extracted based on its location on each plane. The final feature vector is achieved by arranging the values of the local binary pattern.

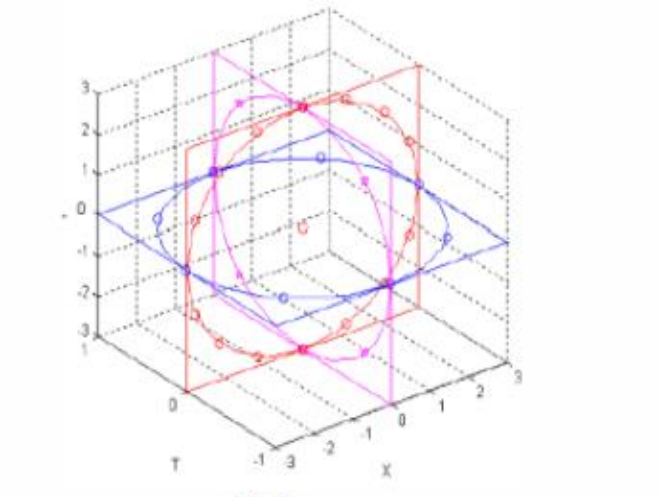


Fig1

Neighborhood point of each voxel in three orthogonal plane.

Figure below: part (a) presents a slice of the initial 3D MRI image with weighting. The uniform local binary pattern algorithm is applied to three orthogonal planes on the entire image. Parts (b), (c), and, (d) are slices of output images considering neighborhood in planes XY, XY, YT, respectively. The feature vector is obtained by concatenating these values together. As we can see, the local binary pattern algorithm has been successful in recognizing edges, flat regions, and the texture structure.

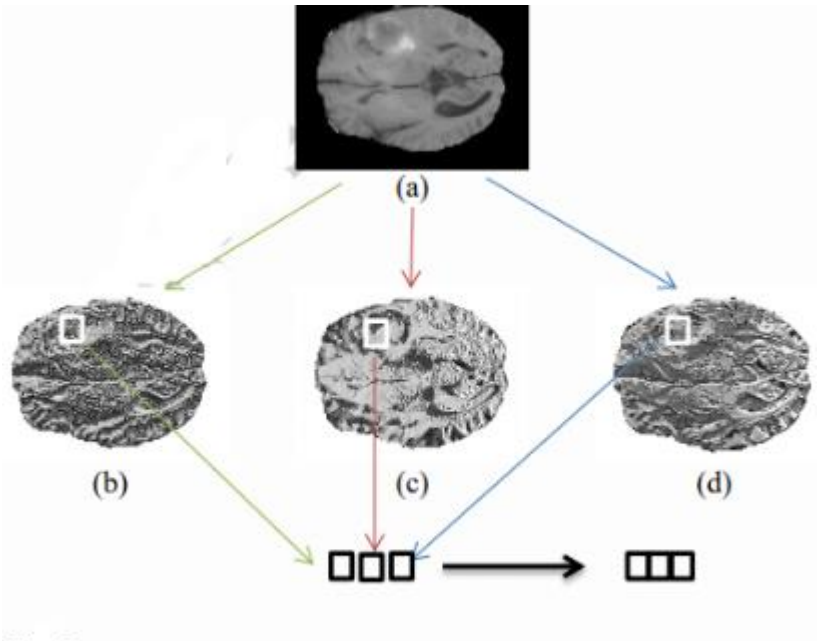


Fig 2. (a) MRI-T1, (b) LBP-XY plane, (c) LBP-XT plane, (d) LBP-YT plane.

3.8 HOG (Histogram Orientation Gradient):

Histogram of oriented gradients is a method in computer vision and image processing that is used to detect the object. In this method, the image is divided into small areas that are joined together, they are called cells. The histogram of oriented gradients is extracted from the cells. First, the image is filtered by Sobel kernel (figure below) to achieve the image gradient along the and directions.

$$[-1 \quad 0 \quad 1] \qquad \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

Horizontal mask Vertical mask

. Sobel mask.

Gx ,Gy are referred to the image gradient along the x and y directions. Magnitude and angle of the image gradient is calculated at each pixel according to equations below . In these equations, |G| is magnitude of gradient, ɵg is angle of gradient, i, and j represent the rows and columns of the image, respectively. To calculate the gradient histogram, angle is divided into n equal distances where n indicates the number of gradient directions or histogram bins. To calculate the histogram, each pixel inside a cell votes to one of the histogram bins based on its gradient angle. These votes are weighed in this pixel based on the gradient size.

$$|G(i,j)| = \sqrt{\left(G_x(i,j)\right)^2 + \left(G_y(i,j)\right)^2} \qquad (34)$$

$$\theta_G(i,j) = tan^{-1}\left(\frac{G_y(i,j)}{G_x(i,j)}\right) \qquad (35)$$

After calculating the gradient histogram in each cell, this histogram is allocated to the cell central pixel. Therefore for each pixel an n dimensional vector, which represents its surrounding gradient histogram, is calculated.

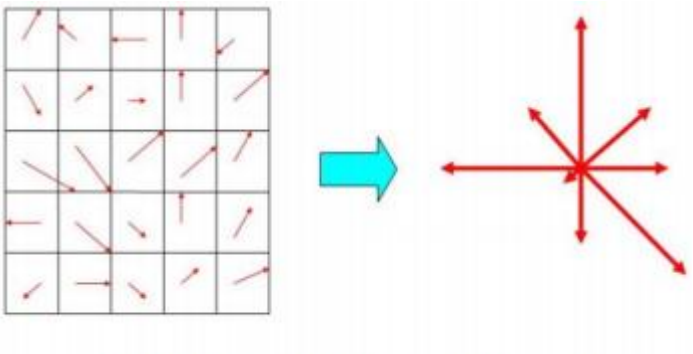


Fig3 A cell and histogram orientation gradiant.

The HOG method of feature extraction was proposed for 2D images but in this paper by considering the location of each voxel in the intersection of three orthogonal planes we extend the method to 3D images. Since 3D images are used in this research we use the idea presented in LBP-TOP in order to extend HOG to 3D images. As it is illustrated in figure below, for each cell we extract the gradient histogram in XY ,YZ and, XZ planes; therefore we have a vector with the length of 9 × 3 for each voxel or cell. Feature extraction step is performed on all MRI modalities.

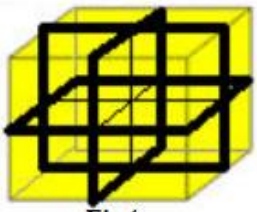


Fig4
Cells in three orthogonal plane.

3.9 Monogenic Signal Analysis

Monogenic signal analysis and its application in brain tumor image recognition are presented.

The monogenic signal is an image analysis approach based on analytic signal concept instigated by “Felsberg and Sommer” and this isotropic generalization has been used for various image analysis and computer vision applications. Monogenic signal is computed by convolving the two-dimensional image with a band-pass quadrature filter generating the local phase and local energy of the image. Monogenic signal analysis is an alternative characterization of any real-valued function using a complex-valued function which is conceptually simpler for the analysis based on analytic signal theory and Hilbert transform theory .The monogenic signal computes the image features of amplitude (A), phase (φ) and local orientation (θ) from the responses to three 2D spherical quadrature filters (SQFs). These features are determined using Riesz transform which generalizes the scalar-to-vector extension of Hilbert transform for multi-dimensional signals and is expressed as,

$$f_R(z) = \begin{bmatrix} f_m(z) \\ f_n(z) \end{bmatrix} = \begin{bmatrix} h_m * f(z) \\ h_n * f(z) \end{bmatrix} \quad \textbf{(36)}$$

where f (z) = z(m, n) is the two-dimensional input signal, hm is the first filter with frequency response Hm = − jωm/ || ω ||, hn is the second filter with frequency response Hn = − jωn/ || ω ||, with ω = (ωm, ωn). Spatial representation of the Riesz kernel is expressed as,

$$(h_m, h_n) = \left(\frac{m}{2\pi \ || \ z \ ||^3}, \frac{n}{2\pi \ || \ z \ ||^3} \right) \quad \textbf{(37)}$$

The two-dimensional signal f (z) is represented using the monogenic signal using Reisz transform

$$f_M(z) = (f(z), f_m(z), f_n(z)) \quad \textbf{(38)}$$

f(z) is the real part of the monogenic signal, fm(z) and fn(z) are the two imaginary parts. Alike the analytic signal representation, the two-dimensional image is then decomposed orthogonally into three different components: local amplitude A, local phase φ, and local orientation ∴, and computed as

$$\text{Local amplitude, } A = \sqrt{f^2 + f_m^2 + f_n^2} \quad \textbf{(39)}$$

$$\text{Local phase, } \phi = -\text{sign}(f_m) atan2 \left(\frac{\sqrt{f_m^2 + f_n^2}}{f} \right) \quad \textbf{(40)}$$

$$\text{Local orientation, } \theta = atan \left(\frac{f_n}{f_m} \right) \quad \textbf{(41)}$$

As discussed above, the local phase and local orientation information are essentially extracted by the two-dimensional monogenic image signal apart from the amplitude. Local orientation effectively captures the geometric characterization, whereas the structural characterization is captured by local phase using the monogenic signal analysis. Figures 3 and 4 depict three different components of local energy and phase each captured for healthy and cancer brain MR image, respectively. During the experimentation, we observed that orientation component was not contributing toward the enhancement of detection rate significantly, and hence, to minimize the feature vector dimension, this component is discarded. Considering the first component, the local energy, as seen in these Figs. 3 and 4, brain MR image with tumor shows significant energy levels compared to the normal brain MR image. Also, the structural variations present in the local phase component of the non-healthy MR image are prominent. Tumor region in the second image is clearly highlighted in all three local energy components. All components from the local phase analysis show notable textural and structural changes in brain tumor image as compared to the normal brain MR image. Additionally, the edge of tumorous region is emphasized remarkably in the second figure (image with tumor). These energy levels and structural dissimilarity motivated to employ monogenic signal analysis approach for brain tumor detection from MR images using GLCM and CLBP textural descriptors.

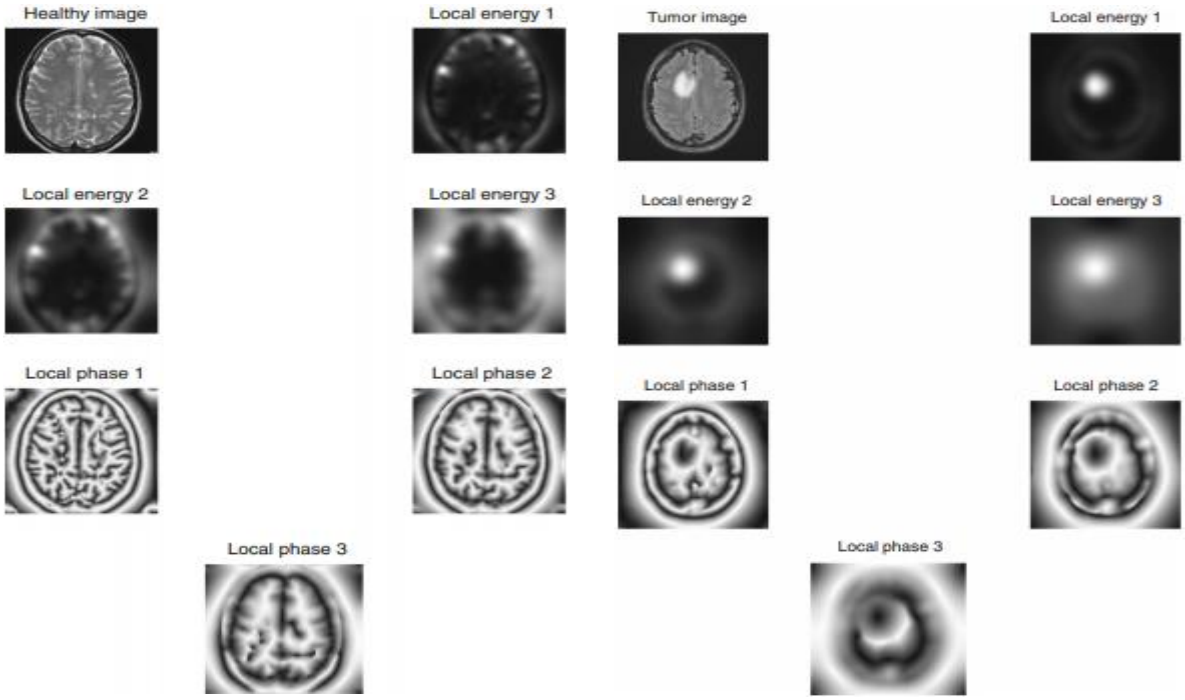


Fig. 4 Energy and orientation components of monogenic signal: (1) original healthy image, (2) local energy component 1, (3) local energy component 2, (4) local energy component 3, (5) local phase component 1, (6) local phase component 2, (7) local phase component *3*

Fig. *5* Energy and orientation components of monogenic signal: (1) original non-healthy image, (2) local energy component 1, (3) local energy component 2, (4) local energy component 3, (5) local phase component 1, (6) local phase component 2, (7) local phase component *3*

Before computing the Resize transform, band-pass filtering operation is applied on the input image because of its finite length. Supplementary advantage of applying bandpass filtering is the invariance–equivariance property of this operation, resulting in independent information extracted from local energy and phase components. Different types of band-pass filtering techniques available are: Gabor, Gaussian derivative, log-Gabor, Cauchy and Poisson filters. Log-Gabor filter is used in Riesz kernels to nullify the direct current (DC) component effect present due to the nonzero mean of Gabor filters that limit the bandwidth. Also, it has logarithmic frequency spacing permitting more information extraction in the high-frequency band. The multiresolution band-pass monogenic image after applying log-Gabor filter is represented as,

$$f_{\log_M} = \left(f_{\log}(z),\ f_{\log_m}(z),\ f_{\log_n}(z)\right) \qquad \textbf{(42)}$$

$$f_{\log_M} = \left(f_{\log}(z),\ h_m * f_{\log}(z),\ h_n * f_{\log}(z)\right) \qquad \textbf{(43)}$$

where flog(z) = f (z) ∗ F−1G(ω), F−1 represents the two-dimensional inverse Fourier transform and G(ω) is the frequency response of log-Gabor filter which is computed as,

$$G(\omega) = \exp\left\{-\frac{\log(\omega/\omega_0)^2}{2\,\log(\sigma/\omega_0)^2}\right\} \qquad \textbf{(44)}$$

where σ represents scaling factor and ω is center frequency. Band-pass representation of the image signal using the local amplitude Alog, orientation θlog and phase φlog is obtained as

$$\text{Local amplitude, } A_{\log} = \sqrt{f_{\log}^2 + f_{\log_{ev}}^2 + f_{\log_{od}}^2} \quad (47)$$

$$\text{Local phase, } \phi_{\log} = -\text{sign}(f_{\log_{od}})atan2\left(\frac{\sqrt{f_{\log_{ev}}^2 + f_{\log_{od}}^2}}{f_{\log}}\right) \quad (48)$$

$$\text{Local orientation, } \theta_{\log} = atan\left(\frac{f_{\log_{od}}}{f_{\log_{ev}}}\right) \quad (49)$$

Finally, two-dimensional image representation using the monogenic scale space is computed as {Alog1 , Alog2 ,..., AlogS , φlog1 , φlog1 ,...,φlogS , θlog1 , θlog2 ,...,θlogS }. In addition to the local energy and phase, additional parameters phase and orientation symmetry components are also extracted using quadrature filters from the monogenic image signal. Estimation of local symmetry and asymmetry structure around a point in an image is important. It is estimated by detecting the difference between sine and cosine of the local phase, feature symmetry S and asymmetry R as,

$$S(z) = \sum_i \frac{\lfloor |f_e, \lambda_i(z)| - |f_o, \lambda_i(z)| - T \rfloor}{A\lambda_i(z) + \epsilon} \quad (50)$$

$$R(z) = \sum_i \frac{\lfloor |f_0, \lambda_i(z)| - |f_e, \lambda_i(z)| - T \rfloor}{A\lambda_i(z) + \epsilon} \quad (51)$$

where f0, is the odd vector and fe is the even vector of monogenic signal. Local symmetry and asymmetry are computed using the magnitude of the monogenic image signal only. The second component, signed feature symmetry is extracted by retaining the sign during computation. Lastly the third component, oriented feature asymmetry is evaluated using the magnitude and direction of the asymmetry (orientation) of the edges present in the image.

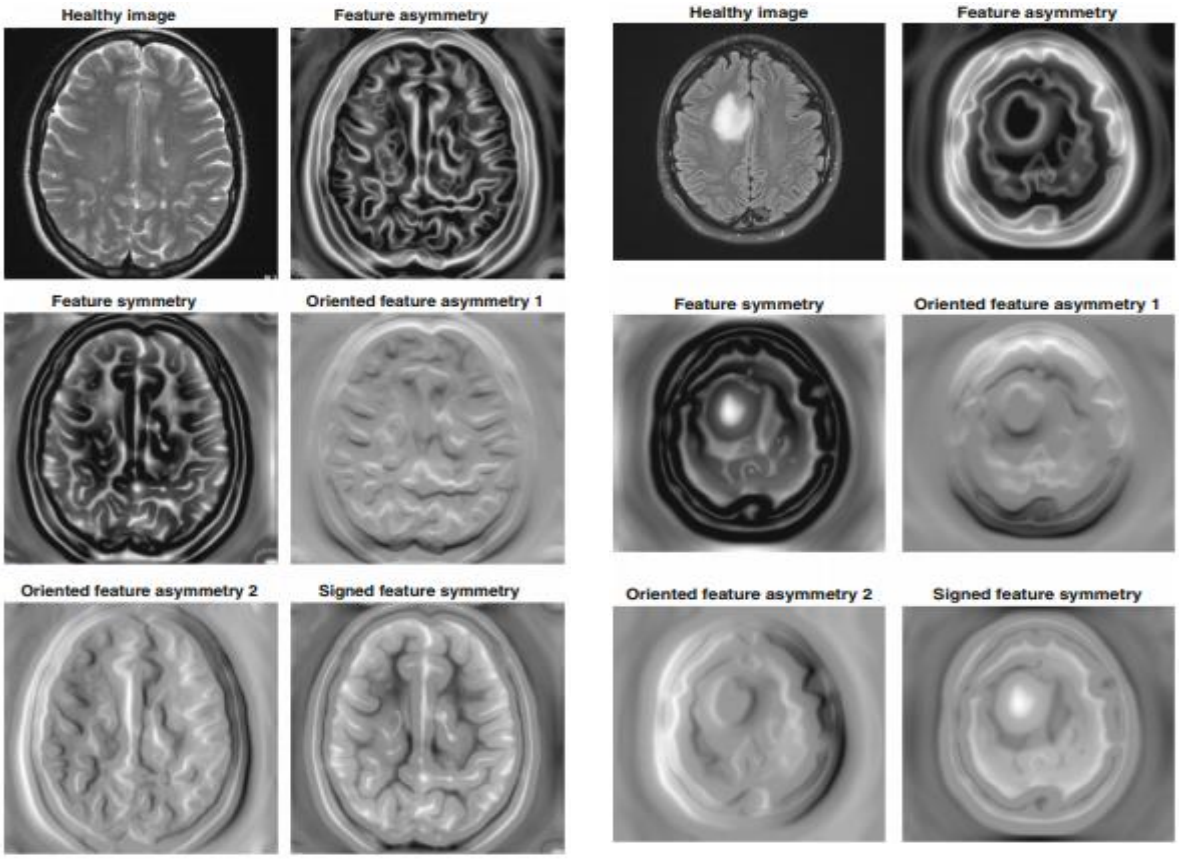


Fig. 6 Phase symmetry and asymmetry components of monginc signal: (1) original healthy image, (2) feature asymmetry, (3) f symmetry, (4) oriented feature asymmetry 1, (5) oriented feature. metry 2, (6) signed feature symmetry

Fig. 7 Phase symmetry and asymmetry components of monogenic signal: (1) original tumor image, (2) feature asymmetry. (3) feature symmetry, (4) oriented feature asymmetry 1, (5) oriented feature asym- metry 2, (6) signed feature symmetry

Figures 6 and 7 illustrate the phase symmetry and asymmetry components of monogenic signal for healthy and cancer brain MR images, respectively. In this study, five components are evaluated for brain tumor detection: (1) feature asymmetry, (2) feature symmetry, (3) oriented feature asymmetry component 1, (4) oriented feature asymmetry component 2 and (5) signed feature symmetry. It is evident from these figures that feature symmetry components show blob-like regions having similar appearance more prominently at the scale of interest. Feature asymmetry component collects edge features which form the object boundaries in the given image.

3.10 Combined Local Binary Patterns (CLBP):

The second texture descriptor considered in this study is a combination of local binary patterns (CLBP) descriptor that generalizes LBP effectively for the texture analysis. CLBP decomposes the input image into the difference, the sign and the magnitude component resulting in more complete characterization of textures compared to original LBP. Hence, CLBP is preferred as compared to LBP for feature extraction from monogenic image signal.

The first CLBP component denoted as sign component (CLBP-S) represents the sign (positive or negative) of difference between the center pixel and local pixel. The second component, CLBP-M indicates the magnitude of the difference between the center pixel and local pixel and CLBP-C indicates the difference between local pixel value and average central pixel value. CLBP includes additional information such as sign and magnitude from image textures useful for classification task.

Given an image pixel fi and its neighbors f j , the difference between fi and f j is computed as,

$$D_j = T_j * M_j \tag{52}$$

where $D_j = |M_j|$, $T_j = \begin{cases} 1 & D_j \geq 0 \\ -1 & D_j < 0 \end{cases}$ where T_j is the sign component and M_j is the magnitude component. Different CLBP components are obtained as,

$$\mathrm{CLBP}_{M_{P,J}} = \sum_{j=0}^{j-1} t(M_j, c)2^j \tag{53}$$

$$t(x, c) = \begin{cases} 1 & x \geq c \\ 0 & x < c \end{cases} \tag{54}$$

where c is the mean value.

$$\mathrm{CLBP}_{S_{P,J}} = \sum_{j=0}^{j-1} t(T_j, 0)2^j \tag{55}$$

$$\mathrm{CLBP}_{C_{P,J}} = t(f_i, c)2^j \tag{56}$$

In this study, two types of CLBP parameters are assessed during feature extraction: sign and magnitude from each of the monogenic image signal components.

Radial basis function (RBF) support vector machine classifier is employed in this study to classify the input MR image into normal or abnormal. RBF kernel function is used in this study as it achieves better results in nonlinear data patterns in the input feature space. RBF-SVM is implemented using the most popular LIBSVM package. To select γ and C parameters, a grid-search approach using K-fold cross validation is performed using the training data.

Feature extraction step generates large feature set as the algorithm consists of various components of monogenic signal analysis (local energy, phase and orientation). To reduce the dimensionality of feature vector and to remove redundant features, Fisher score-based feature selection is employed in this experiment. Feature selection approach selects important features and discards the features that are not contributing in the detection stage. As the Fisher method is ranking approach for feature selection, different threshold T has been used during the training phase to obtain optimal feature set. The proposed algorithm is evaluated using different parameters explained as follows.

– True Negative (TN) Healthy brain MRI image correctly identified as normal image.

– True Positive (TP) Abnormal brain MRI image correctly detected as abnormal.

– False Positive (FP) Healthy brain MRI image wrongly classified as abnormal image.

– False Negative (FN) Abnormal brain image incorrectly detected as healthy brain image.

– Sensitivity (Sen) = $\frac{TP}{TP+FN}$ **(57)**

– Specificity (Spec) = $\frac{TN}{TN+FP}$ **(58)**

– Accuracy (Acc) = $\frac{TP+TN}{TP+TN+FN+FP}$ **(59)**

– Precision (Prec) = $\frac{TP}{TP+FP}$ **(60)**

– F-score = $2 \times \frac{Prec \times Sen}{Prec+ Sen}$ **(61)**

3.11 Shape Feature

Feature extraction is very important technique in image processing. Different shape-based features are extracted from the tumor segmented image which is useful in further classification of the image. Shape features are extracted using region props. Shape features are center of gravity, Circularity ratio, Rectangularity

3.11.1 Centre of gravity

The center of gravity is also called centroid. Its position should be fixed in relation to the shape

$$g_x = \frac{1}{N}\sum_{i=1}^{N} xi$$
$$g_y = \frac{1}{N}\sum_{i=1}^{N} yi \qquad (62)$$

3.11.2 Circularity ratio

Circularity ratio stands for how a shape is like a circle. Circularity ratio is the ratio of the area of a shape to the shape's perimeter square

$$C_2 = \frac{A_s}{O^2} \qquad (63)$$

3.11.3 Rectangularity

Rectangularity stands for how rectangular a shape is, i.e. how much it fills its smallest bounding rectangle: where AS is the area of a shape; AR is the area of the smallest bounding rectangle.

$$\frac{A_s}{A_R} \qquad (64)$$

3.12 Feature Reduction

More number of features increases the computational time and memory storage and hence makes the classification of images more complicated. Principle component analysis (PCA) is an efficient tool to reduce the dimension of a data set consisting of many variables correlated with each other, either heavily or lightly, while retaining the variation present in the Dataset, up to the maximum extent. PCA is one of the most used linear dimensionality reduction technique. When using PCA, we take as input our original data and try to find a combination of the input features which can best summarize the original data distribution so that to reduce its original dimensions. PCA can do this by maximizing variances and Minimizing the reconstruction error by looking at pair wised distances.

3.12.1 Principal Component Analysis (PCA)

To reduce the number of unnecessary feature sand dimensions, we apply PCA in our work. PCA is used as statistical tools for providing orthogonal transformation of data to a new coordinate system and summarizing the variances that eliminate the least contributing components to the variation in data set. For the classification of brain MRI images, Independent component analysis (ICA) is perceived as an extension of principal component analysis (PCA) because of ICA better works on the data that have been already normalized by PCA . PCA is based on the information given by the second order statistics, whereas ICA goes up to high order statistics. PCA finds a set of the most representative projection vectors such that the projected samples retain the most information about original samples of brain MRI images. ICA captures both second and higher-order statistics and projects the input data onto the basis vectors that are as statistically independent as possible. Therefore the result obtained by ICA is assumed to be more meaningful than the one gained by PCA, and primary work for feature extraction and reduction in brain MRI image processing basically depends on PCA . On the other hand, the density vector is radially symmetric and suitably scaled. It is whitened (sphere),but the converse is not always true, because whitening is essentially de-correlation followed by scaling, for which the PCA technique can be used in brain MRI image preprocessing step. In ICA, the order of independent components cannot be determined that implies the correspondence between a physical signal and the estimated independent component is not one to-one for the classification of brain MRI images. This indeterminacy is particularly severe in many applications, where identification of the estimated components is of very high importance, and it makes permutation ambiguity ICA also shows scaling ambiguity where the energy of the independent components cannot be determined, and it scaled in source estimation. Therefore PCA method is best suited for the classification of brain tumor from scale invariant MRI images. Besides other limitations, ICA does not reflect time delays that occur in the signal hence, in our proposed method we use PCA instead of ICA for feature reduction for MRI brain image. The invariant features which are not considered by the PCA. The proposed method uses Independent Component analysis (ICA) for the reduction of the feature vector. ICA deals with higher order statistics and gives rise to independent components which can reveal interesting and fine features in the non-Gaussian hyper spectral image data sets. Hence ICA works more effectively than PCA in reducing the feature vector and also during the classification of brain tumors.

3.12.2 Independent component analysis (ICA)

This topic is based on .ICA is one of the higher-order methods. The projections of ICA are linear but not surely orthogonal. These are statistically independent to each other, which is a stronger condition than uncorrelated ness. The random variables x={x1,....xp}are said to be uncorrelated, if for ∀I ≠ j, 1 ≤ i, j≤p

$$Cov(x_i,x_j)=E\{(x_i-\mu_i)(x_j-\mu_j)\}=E(x_i\,x_j)-E(x_i)E(x_j)=O \quad (65)$$

In converse, the main requirement of independent property is factorization of the multivariate probability density function, which can be written as $f(x_p\ldots\ldots\ldots x_p)=f_1(x_1)\ldots\ldots f_p(x_p)$ Independence property always entails uncorrelated ness, but the vice versa is not there. The two are equivalent only if the function $f(x_p\ldots\ldots\ldots x_p)$ $f(x_p\ldots\ldots\ldots x_p)$ is multivariate normal. Based on Fodor, the noise-free

ICA representation for a p-dimensional random vector x follow to evaluate the components of k-dimensional vector s and the p× k full column rank matrix A

$(x_p\ldots\ldots\ldots x_p)^T=A_{(p\times k)}(s_p\ldots\ldots\ldots s_k)^T$ such that conferring to one of the definitions of Independence the components of k-dimensional vector s are as independent as possible. "At least one of the hidden independent components s_i has to be non-Gaussian to ensure the identify ability of the model" This method uses 13 features like Mean, Standard-Deviation, Entropy, RMS, Variance, Smoothness, Kurtosis, Skewness, IDM, Contrast, Correlation, Energy and Homogeneity.

3.13 Discrete Wavelet Transform (DWT)

DWT is an operation of wavelet transform using a discrete set of the wavelet scales and translation following some specific price- dent The proposed system extracts the wavelet co-efficient from brain MRI images using the DWT coefficient. Localized frequency information generates by DWT, which is altered and scaled from exact wavelets. The basic wavelet formation is offered as follows: if x(t) is a square integrating function, then the

Continuous wavelet transformation of x (t) corresponding to a given wavelet Ψ (t) is constructed as

$$W_{\Psi}(a,b) = \int_{-\infty}^{\infty} x(t)\psi_{a,b}(t)dt$$

Then,

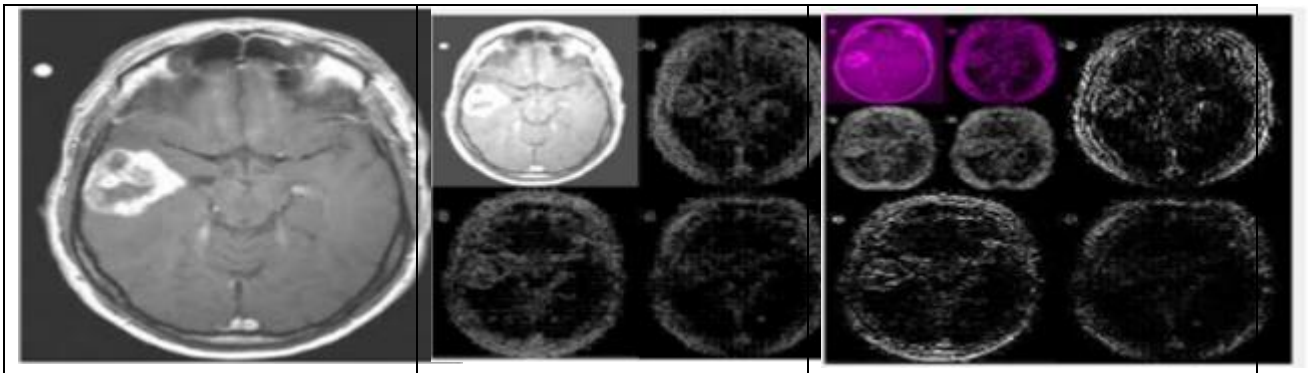
$$\psi a,b(t)\frac{1}{\sqrt{a}}\left(\frac{t-a}{b}\right) \quad \textbf{(66)}$$

In equation 5, a is the dilation factor and b is the translation parameter (both are real positive numbers). The wavelet Ψa,b(t) is determined from the original wavelet Ψ(t) by translation and dilation. Equation 5 can be discretized by constraining a and b to a discrete lattice (a = 2b&a > 0) to deliver the DWT, which can be presented as follows:

$$caj,k(n) = DS\left[\sum_{n} x(n)g_j^*(n-2^j k)\right]$$

$$cdj,k(n) = DS\left[\sum_{n} x(n)h_j^*(n-2^j k)\right] \quad \textbf{(67)}$$

In the proposed algorithm, DWT is used for the single-level composition of a 2D image. Among the four sub band (LL, LH, HL, HH), here LL sub band is considered as an approximation element of the image and used as an input for the next 2D DWT.

		
(A)	(B)	(C)

The procedures of 2-level 2D DWT:(A) normal Benign brain MRI;(B) Level 1 wavelet coefficients;(C) Level 2 wavelet coefficients.

3.14 Log-Polar Transformation (LPT)

LPT is an effective and validate mechanism for the conversion of image geometry from the Cartesian domain to log-polar domain retaining the rotation and scaling invariant properties. The polar coordinates (r, θ) can be represented by:

$$(r,\theta) = \sqrt{(x-x_c)^2 + (y-y_c)^2}, \tan^{-1}\frac{(y-y_c)}{(x-x_c)} \quad \textbf{(68)}$$

The formulas for transformation from log-polar to Carte-Sian Co-ordinates are

$$x = e^{p}\cos\theta$$

$$y = e^{p}\sin\theta \quad \textbf{(69)}$$

LPT is a conformal mapping from the points on the Carte- Sian plane (x, y) to points in the log-polar plane (log(r), θ).(log(αx), log(αy)) = ((logα + log x), (logα + logy)) However, when the original image is translated by (Δx, Δy), the corresponding log-polar coordinates is represented by

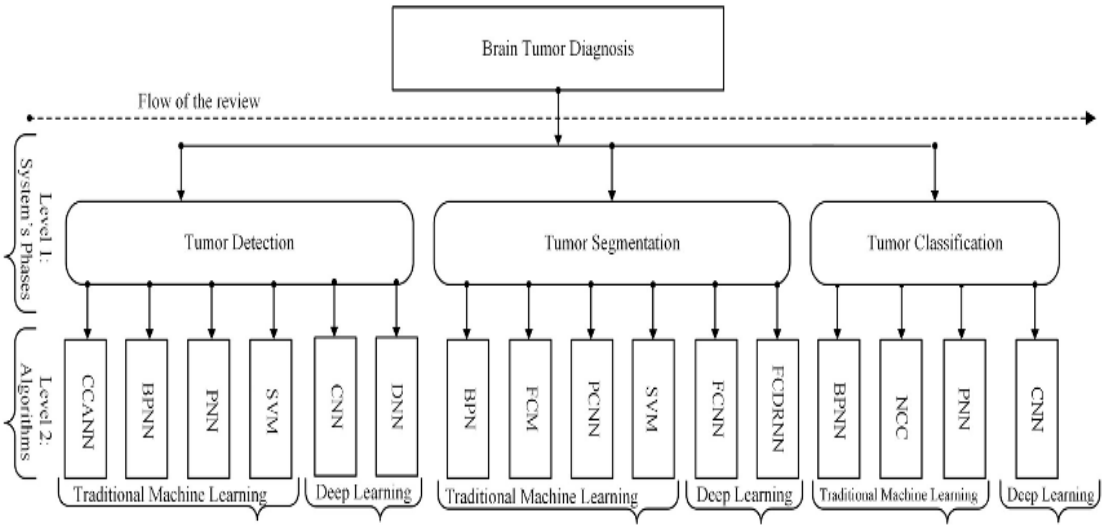
$$r^{1} = \log\sqrt{(e^{p}\cos\theta - \Delta x)^2 + (e^{p}\sin\theta - \Delta y)^2}$$

$$\theta = \tan - 1\frac{(e^{p}\sin\theta - \Delta y)^2}{(e^{p}\cos\theta - \Delta x)^2} \quad \textbf{(70)}$$

3.15 Mining Association rules

Association rule mining has been extensively investigated in the data mining literature. Many efficient algorithms have been proposed, the most popular being apriority and FP-Tree growth Association rule mining typically aims at discovering associations between items in a transactional database. Given a set of transactions D = {T1,...,Tn} and a set of items I = {i1,...,in} such that any transaction T in D is a set of items in I, an association rule is an implication A⇒B where the antecedent A and the consequent B are subsets of a transaction T in D, and A and B have no common items. For the association rule to be acceptable, the conditional probability of B given A has to be higher than a threshold called minimum confidence. Association rules mining is normally a two-step process, wherein the first step frequent item-sets are discovered and in the second step association rules are derived From the frequent item-sets. In our approach, we used the FP tree algorithm in order to discover association rules among the features extracted from the MRI database and the category to which each image belongs. In other words, a rule would describe frequent sets of features per category normal and abnormal (benign and malign) based on the association rule

discovery algorithm. Once the association rules are found, they are used to construct a classification system that categorizes the brain tumor as normal, malignant or benign.



The flow of the article with respect to the tumor detection, segmentation, and classification processes, with two levels of information. Level 1 is used for representing the main brain tumor diagnosis operations, while level shows the identified traditional machine and deep learning techniques.

3.16 Morphological Features

The morphology of the tissue has a significant role in defining that the tissue is regular or irregular. Morphological analysis of the nervous system is important in studies of physiological and pathological conditions. By extracting morphological features from an image, the image morphology is converted into a set of quantitative values which are mostly used in classification subdivision (Segmentation) etc. , where subdivision refers to the classification of an image into different regions The morphological operator feature selection method is more efficient in analysis of high classification and accuracy to detect the tumor area in the brain image .The morphological operator function is a proficient algorithm in which tumor segmentation is performed and its features such as centroid, surrounding and area are calculated from segmented tumors. Furthermore, because of the diverse shape, size and appearance of the tumor, accurate measurement in the brain diagnosis is very difficult, so the quality of the scanned image is enhanced and the morphological operator is scanned and applied to detect tumors of images. Navajo et al. Made it possible to convert the image to the polar region and apply the morphological filter before the process step of reshaping the radial pattern of artifacts into a vertical pattern.

3.17 Texture Feature

Selection of feature extraction process is one of the most important aspects for attaining high recognition performance. Texture is one of the important properties used to recognize the region of interest in the image, whether the image is a “photomicrograph” or a “satellite image”. Texture features are computed from a Gray Level Co-occurrence Matrix (GLCM) that captures the three-dimensional relationship between the pixels of the image. GLCM is an N × N square matrix and N is the number of diverse gray levels in the image. The element “p” (i, j, d, θ) of the GLCM of the image represents the relative frequency, “i” is the gray level of the pixel “p” at the position (x, y), “j” is the distance from “p” at the gray level orientation “θ” of the pixel located in “d” . The texture features extracted from gray level image are “(i) contrast, (ii) correlation (iii) Dissimilarity (iv) energy (v) entropy (vi) Homogeneity (vii) mean (viii) variance (xi) standard deviation”.

3.18 Scale Invariant Feature Transform (SIFT) Features

Lowe proposed Scale Invariant Feature Transform (SIFT), which has been broadly and effectively applied to object detection and face recognition/verification It has also been used to analyses the problems of panoramas reconstruction. Due to the robust nature of rotation, scaling, lighting change, noise and distorting effects, SIFT features have been used in various research fields. These characteristics of SIFT features are suitable for classification of tumor region. In the first step of extracting SIFT features, the key points are localized in an image and SIFT algorithm is applied to a set of training images representing the typical tumor cases. Only the key points that are detected from the tumor region are added to the training dataset. Therefore, the SIFT feature vector is calculated from the neighborhood around a key point at different scales, the localized structure of the tumor is recorded by SIFT. The second step is to find the best candidate match for each key point in the image by recognizing its nearest neighbor in the set of key points gained from training images. The nearest neighbor is well defined as the key point with the minimum Euclidean distance for the invariant descriptor vector.

3.19 Elliptic Fourier Descriptors (EFDs)

The EFDs features are suitable for distinguishing the images having epileptic shape. EFDs features are introduced by Nicolet al. [48] to classify the sold objects such as tank, windmill etc. In the Elliptical Fourier Descriptor (EFD), the shape of the target contour is represented, which is used as a recognition feature, and these features are also widely used in the

application of pattern recognition systems. In this study, the Elliptical Fourier Descriptor (EFD) is used to always guarantee a closed curve regardless of their value, and in this experiment, EFD is more economical (number of coefficients the less contour outlines the other choices (e.g., various other Fourier descriptors) that gave satisfactory outline approximations. To calculate the function of EFD, you need two steps. In the first step, the epileptic object is detected in the white image cluster. In a second step, the epileptic objects are sorted based on their area and the EFDS of the top E object up to the desired level H. The EFDS is closed by a sequence of 8 standardized line segments, depending on the chain code by approximating the shape of the outline; it is invariant to the translation, rotation, start point of the outline. To extract the EFD, H harmonic levels and four Fourier coefficients for each harmonic level, i.e. a, b, c and d, are calculated below.

$$a_i = [a_1, a_2, a_3, \dots \dots \dots a_H]i^T \quad \text{where i=1,2,3,... E}$$
$$b_i = [b_1, b_2, b_3, \dots \dots \dots b_H]i^T$$
$$c_i = [c_1, c_2, c_3, \dots \dots \dots c_H]i^T$$
$$d_i = [d_1, d_2, d_3, \dots \dots \dots d_H]i^T \quad \textbf{(71)}$$

Where a_i, b_i, c_i and d_i vectors contain a, b, c and d Fourier coefficients of i^{th} primitive up to harmonic level "H". The average of these vectors is computed as below:

$$\bar{a} = \frac{1}{E}\sum_{i=1}^{E} a_i, \bar{b} = \frac{1}{E}\sum_{i=1}^{E} b_i, \bar{c} = \frac{1}{E}\sum_{i=1}^{E} c_i \text{ and } \bar{d} = \frac{1}{E}\sum_{i=1}^{E} d_i \quad \textbf{(72)}$$

The final Fourier feature vector is obtained by combining the above average vectors:

$$v = [\bar{a}^T, \bar{b}^T, \bar{c}^T, d^T]^T \quad \textbf{(73)}$$

3.20 Entropy Features

The biological signals are the output of multiple interacting components of biological systems that exhibit patterns in a complex rhythm. These rhythms and patterns are altered due to malfunctioning in structural components and reduced interactions in coupling functions. To understand the dynamics of these rhythmical patterns, it requires robust methods from complexity measures and information theoretic approaches. Recently researchers used complexity based measures and Wavelet packet entropy methods to quantify and analyses the dynamics of physiological systems. The entropy features are extracted using sample entropy,

wavelet entropic measures such as, Shannon, norm, threshold, sure and log energy to extract the useful information hidden in the MRIs of brain tumor subjects.

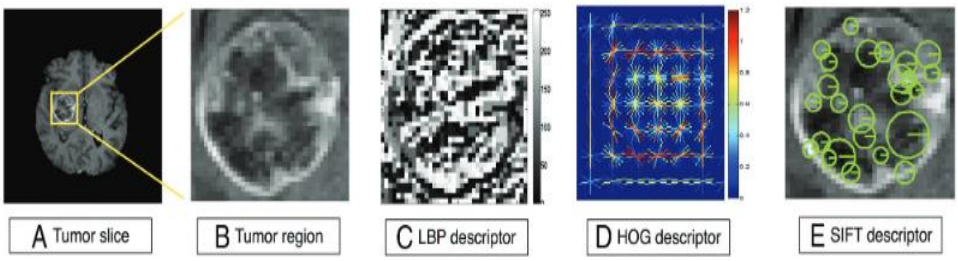
3.21 Quantitative Image Feature Extraction

While the quality, resolution, and flexibility of MR imaging tech-neology has great increased in past decades, the interpretation of images remains largely descriptive, subjective, and non-quantitative. Thus, the central goal of radionics is the development of image analytic techniques that can reproducibly extract objective, quantitative data from MR imaging scans. Linking these quantitative features and underlying tissue dynamics that govern tumor growth and response to therapy has the potential to rapidly ex-panda the scope of cancer imaging research.1Here, we focus on many related computer vision techniques that are particularly useful in quantitative cancer imaging science.

3.22 Computational Image Descriptors

Radionics relies on computational techniques in computer vision to extract many quantitative features from radiologic images. The extracted quantitative features are typically within a defined ROI that could include the whole tumor or specific regions within it. Computational image descriptors quantify visual characteristics at different scales from ROIs, which can be readily translated into radiologic image analysis pertaining to tumor volumetric shapes and visual appearance dynamics. For example, the scale-invariant feature transform (SIFT) is computed through key point detection using a Difference of Gaussian function and local image gradient measurement with radius and scale selections this permits a quantitative measurement of the tumor shape so that subtle variations during treatment (i.e., increasingly round or increasingly elliptic) can be observed and quantified. Several recent studies have demonstrated the accuracy and reproducibility of computational image extraction approaches to capture characteristics of tumor shape and texture information from brain tumor MR imaging. Thus, these Approaches have the potential for large-scale, rapid through-Put and reproducible evaluation and may be applied to routine Clinical imaging studies that are widely available. Here, we describe 2 primary image feature extraction strategies with local- or global-level computations in the context of computer vision. First, local-level feature extraction provides an image descriptor used to compare a pixel being tested with its immediate pixel neighborhood .This allows identification of a small, but biologically important, tumor niche area (a small number of pixels) within an otherwise homogeneous, larger tumor region. This can be achieved; for example, with local binary pat-terns (LBP).These are local image

descriptors sensitive to small monotonic gray-level differences that may not be apparent to a human observer. In contrast, global-level feature extraction



Visualization of computational image feature descriptors. A T1-weighted brain tumor section (A and B) is displayed, and feature visualizations (C–E) are given of LBP (C), HOG (D), and SIFT (E) descriptors. LBP quantifies local pixel structures through a binary coding scheme. HOG computes block-wise histogram gradients with multiple orientations. SIFT detects distributed key points with radius on tumor images.

These multipara metric features create a rich image-driven data base to characterize tumors in MR imaging at different scales.

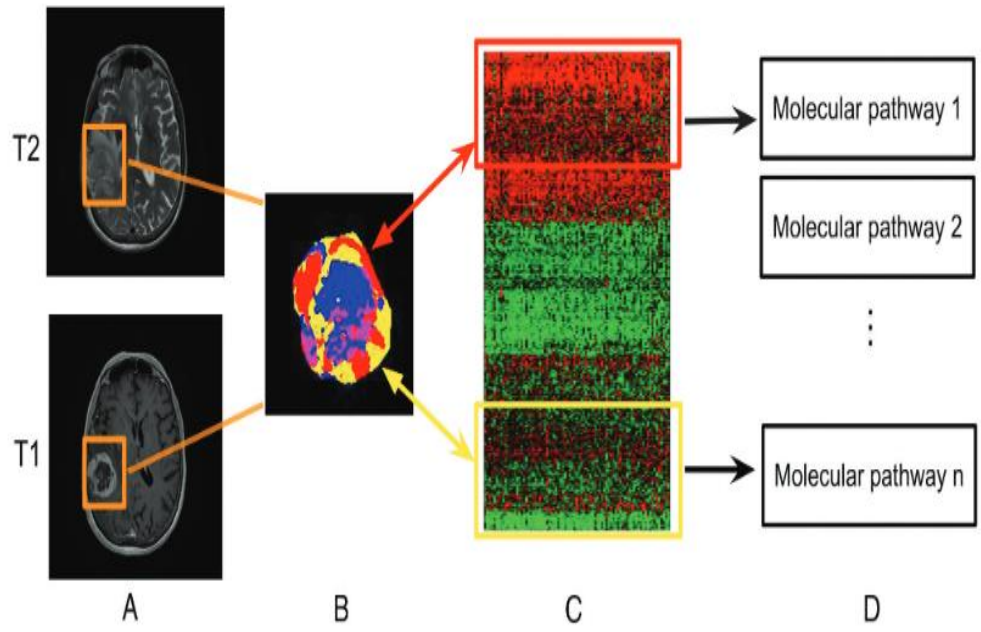
3.23 Biologically Inspired Feature Descriptor

Biologically inspired feature descriptors build on specific biologic hypotheses that transfer the recognized radiology knowledge into quantitative representation, as opposed to pure computational approaches for feature extraction. Under-standing disease characteristics are necessary to propose biologically inspired features because they can be disease-specific. For example, a recent study suggested that MR imaging–derived pharmacokinetic features (e.g., extracellular space per unit volume of tissue) were potential biomarkers for separating outcomes of treatment with concurrent radiation therapy and chemotherapy. Biologically inspired MR imaging features can be used to de-fine organ-level tumor data variation and distribution, offering an opportunity to observe spatial variations and temporal evolution of tumors. For example, a spatial distance measurement was defined to quantitatively explore brain tumor heterogeneity. The proposed spatial distances suggested that the variations among biologically defined tumor sub regions can reflect distinct prognostic information. Also, early temporal changes and spatial het-erogeneity during radiation therapy in heterogeneous regions of high and low perfusion in gliomas might predict different physiologic responses to radiation therapy. Other work has pro-posed a novel concept of imaging habitats that quantifies distinctive tumor sub regions by their local contrast enhancement, edema, and cellularity in MR imaging. Moreover, a recent study

measured the relationships between MR signal and cell density using radio graphically localized biopsies, revealing that T2-FLAIR and ADC sequences were inversely correlated with cell density. The Table highlights several feature descriptors with their

clinical potentials. Biologically inspired features correlate with corresponding

MR imaging sequences because different MR imaging sequences come with various clinical imaging protocols. As a result, selection of MR imaging sequences can directly affect image feature definition and corresponding biologic interpretation. For exam-ple,1 study used apparent diffusion coefficient histograms for the early prediction of drug treatment responses of glioblastomas. In this study, diffusion- and T1-weighted data were used separately for ADC computation and tumor segmentation. The ADCVwas used to describe the diffusion processes that reflect different biomedical mechanisms. These biologically inspired features are quantitative rather than qualitative semantic features annotated by radiologists to describe the tumor environment. The VA-SARI semantic feature set for example, is used to describe the morphology of brain tumors (e.g., tumor lo-cation, shape, and geometric properties) on contrast-enhanced MR imaging



Linking sub regional imaging to molecular profiles in glioblastoma. In this example, tumor sub regions (B) are defined by jointly clustering on contrast-enhanced T1WI and T2WI (A). These sub regions correspond to red (high T1WI and high T2WI), yellow (high T1WI and low T2WI), blue (low T1WI and high T2WI), and pink (low T1WI and low T2WI) areas. The

defined tumor sub regions enable quantitative spatial characterization, offering a means to noninvasively assess specific molecular activities (C) with enriched molecular pathways (D).

Examples of quantitative features with their potential clinical relevance

Quantitative Feature Descriptors	Potential Clinical Relevance
Histogram of contrast-enhanced tumor MRI[45]	Distinguish molecular subtypes
Contrast information between co-occurring subregions[5]	Survival predictor
Pretreatment ADC histograms[82]	Indicator to bevacizumab treatment
HOG[34]	Measure tumor microenvironment
LBP[27]	Measure tumor microenvironment
SIFT[22]	Measure tumor spatial characteristics

3.24 Moment Invariant Feature Extraction

An analysis of object classification or recognition methods based on image moments. The various types of moments are complex moment, geometric moment and moment based invariants with respect to various image degradations and distortions, which can be used as shape descriptors for classification. There is a description about image classification based on moment invariants. They reviewed efficient numerical algorithms, used for moment computation and demonstrated some practical examples of moment invariance based real-time applications. The feature extraction of MR images is done with the consideration of moment invariant functions. Generally, moments are given as a projection of the image function into a polynomial basis. Such projections are known as image moments and the respective functions are called moment invariants. In practice, the interpretation of an image obtained by the MRI system provides the degraded version of the original scene. Those degradations have occurred during image acquisition by factors like lens aberration, the motion of the scene, imaging geometry, wrong focus and random sensor error. The dexterity of invariants with respect to these factors is a crucial part. In our proposed work, we provide a moment invariant mechanism in feature extraction. Images under each moment are too sensitive to local changes, but they are very robust to noise. Accordingly, invariants are applied to intensity changes, convolution, rotational images and contrast images. During MRI, brain is scanned to give distinctive brain images for accurate prediction and classification of brain tumor. Through

differentiating the intensity values of images in increasing order, we evaluate the moment invariance.

$$\varphi_1 = \mu_{20} + \mu_{02}$$

$$\varphi_2 = (\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2$$

$$\varphi_3 = (\mu_{30} - 3\mu_{12})^2 + (3\mu_{21} - \mu_{03})^2$$

$$\varphi_4 = (\mu_{30} + \mu_{12})^2 + (\mu_{21} + \mu_{03})^2$$

$$\varphi_5 = (\mu_{30} - 3\mu_{12})(\mu_{30} + \mu_{12})((\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2) + (3\mu_{21} - \mu_{03})(\mu_{21} + \mu_{03})(3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2)$$

$$\varphi_6 = (\mu_{20} - \mu_{03})((\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2) + 4\mu_{11}(\mu_{30} + \mu_{12})(\mu_{21} + \mu_{03})$$

$$\varphi_7 = (3\mu_{21} - \mu_{03})(\mu_{30} + \mu_{12})((\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2) - (\mu_{30} - 3\mu_{12})(\mu_{21} + \mu_{03})(3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2)$$

(74)

Where φ represents an invariant value of extracting feature of a particular brain slice, which is obtained by μ value, differential values of image intensities. From the above equations, the distinctive features of images are extracted based on 7invariants of rotation using 3rd order differentiations. In our first classification method, the outcome will be given for classification precisely to the trained neural network classifier and preceded with the section 3.4. The next method proceeds with the following section and classifies the brain images into normal, benign and malignant.

3.25 Binary Association Rule Generation

The conceit of feature selection involves in reducing the inputs to an endurable size for effective processing and analysis. A quality pattern has been discovered with substantial features from large training dataset using binary association rule. The rule pursues in discovering the association among features extracted from

the MRI image gallery. Moreover, it contrives strong rules in a database for analysis using different measures of intrusiveness. The problem of binary association rule generation is given as: Let D= {t1, t2... tm} be a set of transactions and I= {i1, i2... in} be a set of items. It is conspicuous that each transaction has a subset of the items in I. Inherently, the aforementioned rule is defined as an implication of the form, where (X is the antecedent of the rule and Y is the consequent of the rule). The association rules are confined such that the antecedent of the rules

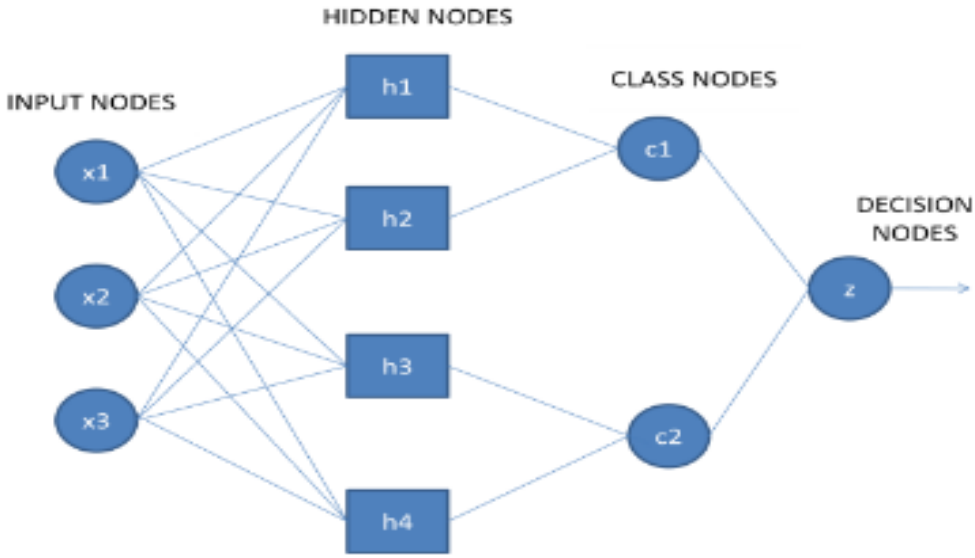
is comprised conjunction of features from the magnetic resonance brain image whereas the consequent of the rule is constantly the class label to which the brain image concerns. The method draws in finding rules that provide minimum confident and minimum support values specified by the user.

3.26 DT-CWT

The multidimensional (M-D) dual-tree CWT is no detachable but is based on a ciphering adequate, separable filter bank (FB). The theory behind the dual-tree transforms shows how compound wavelets with good properties can be designed, and illustrates a range of applications in signal and image processing. We use the manifold number symbol C in CWT to avoid distractions with the often-used acronym CWT for the (different) continuous wavelet transform

3.27 PNN

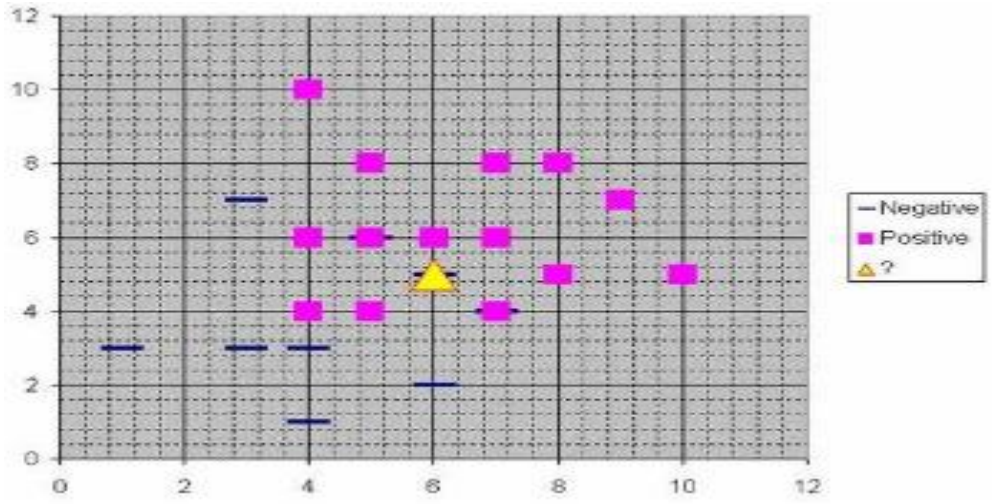
Probabilistic (PNN) and General Regression Neural Networks (GRNN) have collateral architectonics, but there is a integral inequality: Probabilistic networks perform classification where the target variable is categorical, whereas general regression neural networks percolate regression where the intention variable is continuous. If you select a PNN/GRNN network, DTREG will erratically favored the factual type of network Based on the type of target variable



PNN architecture

3.27.1 How PNN network work: Probabilistic

Neural networks are rudimentary coincidental to K-Nearest Neighbor (kNN) models. The basic idea is that a conjectured argot value of an item is inclined to be about the look-alike as other items that have close values of the conjecture variables. Consider this figure:



working with PNN

Predicate that each case in the training set has two forecaster variables, x and y. The cases are sketch using their x,y accommodates as shown in the figure. Also predicate that the forecaster variable has two pigeonholed, positive which is Symbolize by a square and negative which is symbolize by a dash. Now, suppose we are trying to forecast the value of a New case delineated by the triangle with forecaster values x=6, y=5.1.Should we forecast the target as positive or negative? Notice that the triangle is position almost exactly on top of a dash illustrating a negative value. But that dash is in a fairly noteworthy position set side by side to the other dashes which are clustered below the squares and left of center. So it could be that the substrata Negative value is an odd case.

3.28 SFCM

Neighboring pixels possess highly correlated intensities, and Probability of their belongingness to the same cluster high .therefore spatial relationship is important of clustering. In Other words, these neighboring Pixels possess similar feature Values, and the probability that they belong to the same cluster is great. This geographical relationship is paramount in Clustering, but it is not utilized in a standard FCM algorithm. To exploit the spatial information, a spatial function is defined as

$$hij = \sum_{k \in NB(xj)} (u_{ik}) \quad \textbf{(75)}$$

Where NB (x j) impersonates a square window focalized on pixel xj in the geographical domain. A 5! 5 windows were used brought out this work. Just like the membership function, the geographical function impersonates the anticipation that pixel xj belongs to it cluster. The geographical function of a pixel for a cluster is large if the superiority of its propinquity belongs to the same clusters. The geographical function is assimilated into membership function as follows:

$$u_{ij} = u_{ij}^{p} \; h_{ij}^{q} \; / \sum_{k=1}^{c} u_{kj}^{p} \; h_{kj}^{q} \qquad \textbf{(76)}$$

P and q are parameters to control the relative importance of both functions. In unvarying region, the geographical functions simply fortify the original membership, and the clustering result remains unchanged. However, for a clamorous pixel, these formulas diminish the weighting of a clamorous cluster by the labels of its neighboring pixels. As a result, misclassified pixels from clamorous or specious splotch can effortlessly be reformed. The geographical FCM with parameter p and q is symbolize SFCM p, q. Note that SFCM1,0 is interchangeable to the decorous FCM.

3.28.1 SFCM ALGORITHM

1. As in the standard FCM, initialize cluster centers v is fuzzification parameter m and the value ϵ>0
2. Calculate the membership matrix $u = [u_{ij}]$ in the feature space according to equation
3. Map the membership from the feature space to the spatial domain and calculate the spatial function according to algorithm (1). Clustering is procedure with the new membership that is the result of spatial function from the equation (2).
4. Update the cluster centers with equation.
5. If $|U^{(1+1)} - U^{(1)}| \leq \epsilon$, where is the iteration number, then go to step2
6. Calculate final cluster centers
7. Perform the maximum membership segmentation.

4 Chapter 4. Deep Learning and Convolutional Neural Network

4.1 INTRODUCTION

Since the 1950s, a small subset of Artificial Intelligence (AI), often called Machine Learning (ML), has revolutionized several fields in the last few decades. Neural Networks (NN) are a subfield of ML, and it was this subfield that spawned Deep Learning (DL). Since its inception DL has been creating ever larger disruptions, showing outstanding success in almost every application domain. Fig.4.1 shows, the taxonomy of AI. DL (using either deep architecture of learning or hierarchical learning approaches) is a class of ML developed largely from 2006 onward. Learning is a procedure consisting of estimating the model parameters so that the learned model (algorithm) can perform a specific task. For example, in Artificial Neural Networks (ANN), the parameters are the weight matrices DL on the other hand consists of several layers in between the input and output layer which allows for many stages of non-linear information processing units with hierarchical architectures to be present that are exploited for feature learning and pattern classification Learning methods based on representations of data can also be defined as representation learning Recent literature states that DL based representation learning involves a hierarchy of features or concepts, where the high-level concepts can be defined from the low-level ones and low-level concepts can be defined from high-level ones. In some articles DL has been described as a universal learning approach that is able to solve almost all kinds of problems in different application domains.

Fig. 4.1. AI: Artificial Intelligence ML, NN, DL, and Spinking Neural Networks (SNN)

4.2 Types of DL approaches

Like machine learning, deep learning approaches can be categorized as follows: supervised, semi-supervised or partially supervised, and unsupervised. In addition, there is another category of learning called Reinforcement Learning (RL) or Deep RL (DRL) which are often discussed under the scope of semi supervised or sometimes under unsupervised learning approaches.

4.2.1 Supervised Learning

Supervised learning is a learning technique that uses labeled data. In the case of supervised DL approaches, the environment has a set of inputs and corresponding outputs. For example, if for input $\mathbf{x_t}$, the intelligent agent predicts $\hat{\mathbf{y}} = \mathbf{f}(\mathbf{x_t})$, the agent will receive a loss value . The agent will then iteratively modify the network parameters for better approximation of the desired outputs. After successful training, the agent will be able to get the correct answers to questions from the environment. There are different supervised learning approaches for deep leaning including Deep Neural Networks (DNN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) including Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRU).

4.2.2 Semi-supervised Learning

Semi-supervised learning is learning that occurs based on partially labeled datasets (often also called reinforcement learning). of this study surveys DRL approaches. In some cases, DRL and Generative Adversarial Networks (GAN) are used as semi-supervised learning techniques. Additionally, RNN including LSTM and GRU are used for semi-supervised learning as well

4.2.3 Unsupervised learning

Unsupervised learning systems are ones that can without the presence of data labels. In this case, the agent learns the internal representation or important features to discover unknown relationships or structure within the input data. Often clustering, dimensionality reduction, and generative techniques are considered as unsupervised learning approaches. There are several members of the deep learning family that are good at clustering and non-linear dimensionality reduction, including Auto Encoders (AE), Restricted Boltzmann Machines (RBM), and the recently developed GAN. In addition, RNNs, such as LSTM and RL, are also used for unsupervised learning in many application domains

4.2.4 Deep Reinforcement Learning (DRL)

Deep Reinforcement Learning is a learning technique for use in unknown environments. DRL began in 2013 with Google Deep Mind. From then on, several advanced methods have been proposed based on RL. Here is an example of RL: if environment samples inputs: $x_t \sim \rho$, agent predict $\hat{y}_t = f(x_t)$, agent receive cost: $c_t \sim P(c_t | x_t, \hat{y}_t)$ where P is an unknown probability distribution, the environment asks an agent a question, and gives a noisy score as the answer. Sometimes this approach is called semi-supervised learning as well. There are many semi-supervised and un-supervised techniques that have been implemented based on this concept. In RL, we do not have a straight forward loss function, thus making learning harder compared to traditional supervised approaches. The fundamental differences between RL and supervised learning are: first, you do not have full access to the function you are trying to optimize; you must query them through interaction, and second, you are interacting with a state-based environment: input x_t depends on previous actions. Depending upon the problem scope or space, you can decide which type of RL needs to be applied for solving a task. If the problem has a lot of parameters to be optimized, DRL is the best way to go. If the problem has fewer parameters for optimization, a derivation free RL approach is good

4.3 Feature Learning

A key difference between traditional ML and DL is in how features are extracted. Traditional ML approaches use handmade features by applying several feature extraction algorithms including Scale Invariant Feature Transform (SIFT), Speeded Up Robust Features (SURF), GIST, RANSAC, Histogram Oriented Gradient (HOG), Local Binary Pattern (LBP), Empirical mode decomposition (EMD) for speech analysis, and many more. Finally, the leaning algorithms including support vector machine (SVM), Random Forest (RF), Principle Component Analysis (PCA), Kernel PCA (KPCA), Linear Decrement Analysis (LDA), Fisher Decrement Analysis (FDA), and many more are applied for classification on the extracted features. Additionally, other boosting approaches are often used where several learning algorithms are applied on the features of a single task or dataset and a decision is made according to the multiple outcomes from the different algorithms. On the other hand, in the case of DL, the features are learned automatically and are represented hierarchically in multiple levels. This is the strong point of deep learning against traditional machine learning approaches. The following table shows the different feature-based learning approaches with different learning steps.

DIFFERENT FEATURE LEARNING APPROACHES

Approaches	Learning steps				
Rule based	Input	Hand-design features	Output		
Traditional Machine Learning	Input	Hand-design features	Mapping from features	Output	
Representation Learning	Input	Features	Mapping from features	Output	
Deep Learning	Input	Simple features	Complex features	Mapping from features	Output

4.3.1 DEEP NEURAL NETWORK (DNN)

4.3.1.1 The History of DNN

Computational neurobiology has conducted significant research on constructing computational models of artificial neurons. Artificial neurons, which try to mimic the behavior of the human brain, are the fundamental component for building ANNs. The basic computational element (neuron) is called a node (or unit) which receives inputs from external sources, and has some internal parameters (including weights and biases that are learned during training) which produce outputs. This unit is called a perceptron. The basic block diagram of a perceptron for NNs is shown in the following diagram.

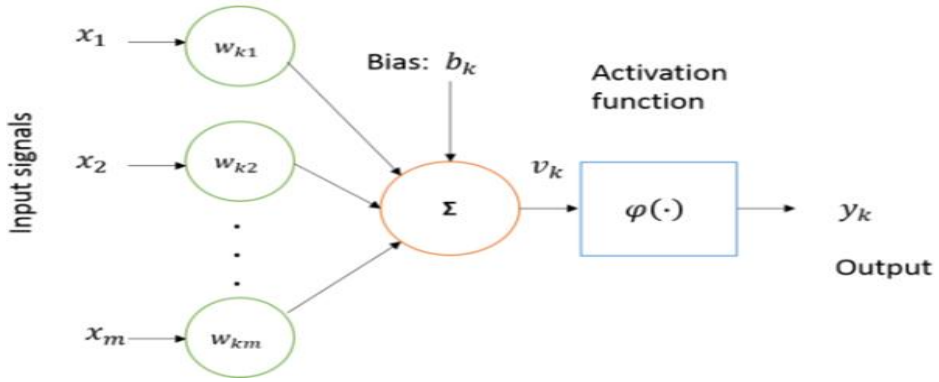


Fig.4.2. Basic model of neuron

Fig.4.2. shows the basic nonlinear model of a neuron, where $x_1, x_2, x_3, \cdots x_m$ are input signals; $w_{k1}, w_{k2}, w_{k3}, \cdots w_{km}$ are synaptic weights; v_k is the linear combination of input signals; $\varphi(\cdot)$ is the activation function (such as sigmoid), and y_k is the output. The bias b_k is added with a linear combiner of out puts v_k, which has the effect of applying an affine transformation, producing the outputs y_k.The neuron functionality can be represented mathematically as follows:

$$v_k = \sum_{j=1}^{m} w_{kj}\, x_j \qquad (1)$$

$$y_k = \varphi(v_k + b_k) \qquad (2)$$

ANNs or general NNs consist of Multilayer Perceptron's (MLP) which contain one or more hidden layers with multiple hidden units (neurons) in them. The NN model with MLP is shown in Fig.4.3.

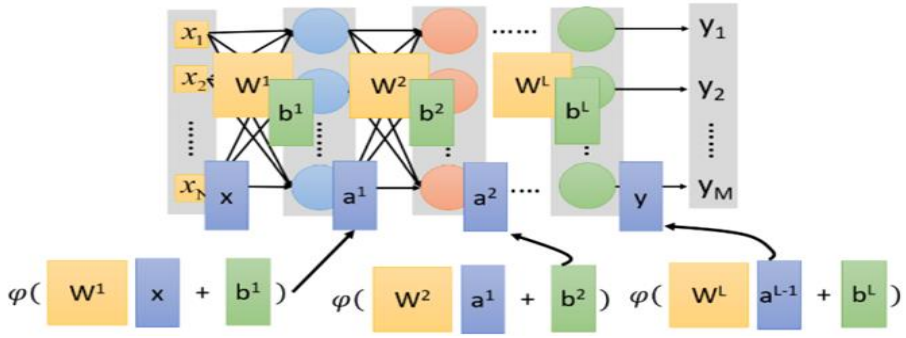


Fig.4.3. Neural network model with multiple layers perceptron

The multilayer perceptron can be expressed mathematically (which is a composite function) as*:*

$$\mathbf{y = f(x) = \varphi(W^L \dots \varphi(w^2\varphi(w^1x + b^1) + b^2) \dots + b^L} \qquad \textit{(3)}$$

4.3.1.2 Gradient descent

The gradient descent approach is a first order optimization algorithm which is used for finding the local minima of an objective function. This has been used for training ANNs in the last couple of decades successfully. Algorithm I explains the concept of gradient descent:

Algorithm I. Gradient descent

Inputs: loss function ε, learning rate η, dataset X, y and the model $\mathcal{F}(\theta, x)$

Outputs: Optimum θ which minimizes ε

REPEAT until converge:

$$\tilde{y} = \mathcal{F}(\theta, x)$$

$$\theta = \theta - \eta \cdot \frac{1}{N}\sum_{i=1}^{N} \frac{\partial \varepsilon(y, \tilde{y})}{\partial \theta}$$

End

4.3.1.3 Stochastic Gradient Descent (SGD)

Since a long training time is the main drawback for the traditional gradient descent approach, the SGD approach is used for training Deep Neural Networks (DNN) Algorithm II explains SGD in detail.

Algorithm II. Stochastic Gradient Descent (SGD)

Inputs: loss function ε, learning rate η, dataset X, y and the model $\mathcal{F}(\theta, x)$

Outputs: Optimum θ which minimizes ε

REPEAT until converge:

Shuffle X, y;

For each batch of x_i, y_i in X, y **do**

$\tilde{y}_i = \mathcal{F}(\theta, x_i)$;

$\theta = \theta - \eta \cdot \frac{1}{N} \sum_{i=1}^{N} \frac{\partial \varepsilon(y_i, \tilde{y}_i)}{\partial \theta}$

End

4.3.1.4 Back-propagation

DNN are trained with the popular Backpropagation (BP) algorithm with SGD. The pseudo code of the basic Back propagation is given in Algorithm III. In the case of MLPs, we can easily represent NN models using computation graphs which are directive acyclic graphs. For that representation of DL, we can use the chain-rule to efficiently calculate the gradient from the top to the bottom layers with BP as shown in Algorithm III for a single path network. For example:

$$\mathbf{y = f(x) = \varphi(W^L \dots \varphi(w^2 \varphi(w^1 x + b^1) + b^2) \dots + b^L} \quad \textit{(4)}$$

This is composite function for *L* layers of a network. In case of *L* = 2, then the function can be written

$$\mathbf{y = f(x) = f(g(x))} \quad \textit{(5)}$$

According to the chain rule, the derivative of this function can be written as

$$\frac{\partial y(x)}{\partial x} = \frac{\partial f(x)}{\partial x} = \mathbf{f'(g(x)) \cdot g'(x)} \quad \textit{(6)}$$

4.3.1.5 Momentum

Momentum is a method which helps to accelerate the training process with the SGD approach. The main idea behind it is to use the moving average of the gradient instead of using only the current real value of the gradient. We can express this with the following equation mathematically:

$$\mathbf{v_t = \gamma\, v_{t-1} - \eta\, \nabla F(\theta_{t-1})} \quad \textit{(7)}$$

$$\mathbf{\theta_t = (\theta_{t-1} + v_t)} \quad \textit{(8)}$$

Here γ is the momentum and η is the learning rate for the t^{th} round of training. The main advantage of using momentum during training is to prevent the network from getting stuck in local minimum. The values of momentum are $\gamma \in (0,1]$. It is noted that a higher momentum value overshoots its minimum, possibly making the network unstable. In general, γ is set to 0.5 until the initial learning stabilizes and is then increased to 0.9 or higher

Algorithm III. Back-propagation

Input: A network with l layers, the activation function σ_l , the outputs of hidden layer $h_l = \sigma_l(W_l^T h_{l-1} + b_l)$ and the network output $\tilde{y} = h_l$

Compute the gradient: $\delta \leftarrow \frac{\partial \varepsilon(y_i, \tilde{y}_i)}{\partial y}$

For $i \leftarrow l$ to 0 **do**

Calculate gradient for present layer:

$$\frac{\partial \varepsilon(y,\tilde{y})}{\partial W_l} = \frac{\partial \varepsilon(y,\tilde{y})}{\partial h_l}\frac{\partial h_l}{\partial W_l} = \delta \frac{\partial h_l}{\partial W_l}$$

$$\frac{\partial \varepsilon(y,\tilde{y})}{\partial b_l} = \frac{\partial \varepsilon(y,\tilde{y})}{\partial h_l}\frac{\partial h_l}{\partial b_l} = \delta \frac{\partial h_l}{\partial b_l}$$

Apply gradient descent using $\frac{\partial \varepsilon(y,\tilde{y})}{\partial W_l}$ and $\frac{\partial \varepsilon(y,\tilde{y})}{\partial b_l}$

Back-propagate gradient to the lower layer

$$\delta \leftarrow \frac{\partial \varepsilon(y,\tilde{y})}{\partial h_l}\frac{\partial h_l}{\partial h_{l-1}} = \delta \frac{\partial h_l}{\partial h_{l-1}}$$

End

4.3.1.6 Learning rate (η)

The learning rate is an important component for training DNN. The learning rate is the step size considered during training which makes the training process faster. However, selecting the value of the learning rate is sensitive. For example: if you choose a larger value for η, the network may start diverging instead of converging. On the other hand, if you choose a smaller value for η, it will take more time for the network to converge. In addition, it may easily get stuck in local minima. The typical solution for this problem is to reduce the learning rate during training . There are three common approaches used for reducing the learning rate during training: constant, factored, and exponential decay. First, we can define a constant ζ which is applied to reduce the learning rate manually with a defined step function. Second, the learning rate can be adjusted during training with the following equation:

$$\boldsymbol{\eta_t = \eta_0 \beta^{t/\epsilon}} \qquad \textit{(9)}$$

Where η_t is the t^{th} round learning rate, η_0 is the initial learning rate, and β is the decay factor with a value between the range of (0,1). The step function format for exponential decay is:

$$\boldsymbol{\eta_t = \eta_0 \beta^{[t/\epsilon]}} \qquad \textit{(10)}$$

The common practice is to use a learning rate decay of β = 0.1 to reduce the learning rate by a factor of 10 at each stage.

4.3.1.7 Weight decay

Weight decay is used for training deep learning models as a L2 regularization approach, which helps to prevent over fitting the network and model generalization. L2 regularization for $\mathcal{F}(\theta, x)$ can be define as:

$$\boldsymbol{\Omega} = \|\boldsymbol{\theta}\|^2 \quad \textit{(11)}$$

$$\hat{\boldsymbol{\varepsilon}}(\mathbf{F}(\boldsymbol{\theta}, \mathbf{x}), \mathbf{y}) = \boldsymbol{\varepsilon}(\mathbf{F}(\boldsymbol{\theta}, \mathbf{x}), \mathbf{y}) + \frac{\mathbf{1}}{\mathbf{2}}\boldsymbol{\lambda}\,\Omega \quad \textit{(12)}$$

The gradient for the weight θ is: $\frac{\partial \frac{1}{2}\lambda\,\Omega}{\partial \boldsymbol{\theta}} = \boldsymbol{\lambda}\,.\,\boldsymbol{\theta}$ *(13)*

General practice is to use the value λ = 0.0004. A smaller λ will accelerate training. Other necessary components for efficient training including data preprocessing and augmentation, network initialization approaches, batch normalization, activation functions, regularization with dropout, and different optimization approaches. In the last few decades, many efficient approaches have been proposed for better training of deep neural networks. Before 2006, attempts taken at training deep architectures failed: training a deep supervised feed-forward neural network tended to yield worse results (both in training and in test error) then shallow ones (with 1 or 2 hidden layers). Hinton's revolutionary work on DBNs spearheaded a change in this in 2006. Due to their composition, many layers of DNNs are more capable at representing highly varying nonlinear functions compared to shallow learning approaches Moreover, DNNs are more efficient for learning because of the combination of feature extraction and classification layers. The following sections discuss in detail about different DL approaches with necessary components.

4.3.2 CONVOLUTIONAL NEURAL NETWORKS (CNN)

4.3.2.1 CNN overview

This network structure was first proposed by Fukushima in 1988. It was not widely used however due to limits of computation hardware for training the network. In the 1990s, LeCun et al. applied a gradient-based learning algorithm to CNNs and obtained successful results for the handwritten digit classification problem. After that, researchers further improved CNNs and reported state-of-the-art results in many recognition tasks. CNNs have several advantages over DNNs, including being more similar to the human visual processing system, being highly optimized in structure for processing 2D and 3D images, and being effective at learning and extracting abstractions of 2D features. The max pooling layer of CNNs is effective in absorbing shape variations. Moreover, composed of sparse connections with tied weights, CNNs have significantly fewer parameters than a fully connected network of similar size. Most of all, CNNs are trained with the gradient-based learning algorithm, and suffer less from the

diminishing gradient problem. Given that the gradient-based algorithm trains the whole network to minimize an error criterion directly, CNNs can produce highly optimized weights. Fig. shows the overall architecture of CNNs consist of two main parts: feature extractors and a classifier. In the feature extraction layers, each layer of the network receives the output from its immediate previous layer as its input, and passes its output as the input to the next layer. The CNN architecture consists of a combination of three types of layers: convolution, max-pooling, and classification. There are two types of layers in the low and middle-level of the network: convolutional layers and max-pooling layers. The even numbered layers are for convolutions and the odd numbered layers are for max-pooling operations. The output nodes of the convolution and maxpooling layers are grouped into a 2D plane called feature mapping. Each plane of a layer is usually derived of the combination of one or more planes of previous layers. The nodes of a plane are connected to a small region of each connected planes of the previous layer. Each node of the convolution layer extracts the features from the input images by convolution operations on the input nodes. Higher-level features are derived from features propagated from lower level layers. As the features propagate to the highest layer or level, the dimensions of features are reduced depending on the size of kernel for the convolutional and max-pooling operations respectively. However, the number of feature maps usually increased for representing better features of the input images for ensuring classification accuracy. The outputs of the last layer of the CNN are used as the input to a fully connected network which is called classification layer. Feed-forward neural networks have been used as the classification layer as they have better performance . In the classification layer, the desired numbers of features are selected as inputs with respect to the dimension of the weight matrix of the final neural network. However, the fully connected layers are expensive in terms of network or learning parameters. Nowadays, there are several new techniques including average pooling and global average pooling that are used as an alternative of fullyconnected networks. The score of the respective class is calculated in the top classification layer using a soft-max layer. Based on the highest score, the classifier gives output for the corresponding classes. Mathematical details on different layers of CNNs are discussed in the following section.

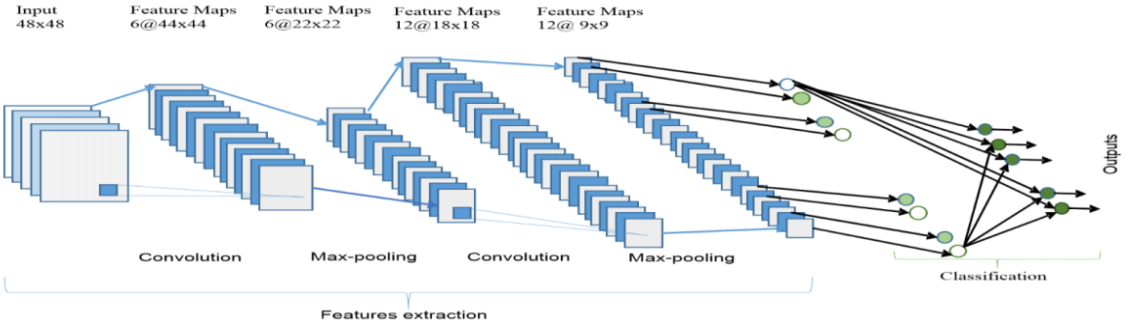


Fig.4.4. The overall architecture of the CNN include an input layer. Multiple altemating convolution and max-pooling, one fully connected layer and one classification layer

4.3.2.2 Convolution Layer

In this layer, feature maps from previous layers are convolved with learnable kernels. The output of the kernels go through a linear or non-linear activation function such as a(sigmoid, hyperbolic tangent, Softmax, rectified linear, and identity functions) to form the output feature maps. Each of the output feature maps can be combined with more than one input feature map. In general, we have that

$$\mathbf{x_j^l = f\left(\sum_{i\epsilon M_j} x_i^{l-1} * k_{ij}^l + b_j^l\right)} \qquad \textit{(14)}$$

Where $\mathbf{x_j^l}$ is the output of the current layer, $\mathbf{x_i^{l-1}}$ is the previous layer output, $\mathbf{k_{ij}^l}$ is the kernel for the present layer, and $\mathbf{b_j^l}$are biases for the current layer. $\mathbf{M_j}$represents a selection of input maps. For each output map, an additive bias *b* is given. However, the input maps will be convolved with distinct kernels to generate the corresponding output maps. The output maps finally go through a linear or non-linear activation function (such as sigmoid, hyperbolic tangent, soft max, rectified linear, or identity functions).

4.3.2.3 Sub-sampling Layer

The subsampling layer performs the down sampled operation on the input maps. This is commonly known as the pooling layer. In this layer, the number of input and output feature maps does not change. For example, if there are *N* input maps, then there will be exactly *N* output maps. Due to the down sampling operation the size of each dimension of the output maps will be reduced, depending on the size of the down sampling mask. For example: if a 2×2 down sampling kernel is used, then each output dimension will be the half of the corresponding input dimension for all the images. This operation can be formulated as

$$\mathbf{x_j^l = down\left(x_j^{l-1}\right)} \qquad \textit{(15)}$$

Where down($\mathbf{x_j^{l-1}}$) represents a sub-sampling function. Two types of operations are mostly performed in this layer: average pooling or max-pooling. In the case of the average pooling approach, the function usually sums up over N×N patches of the feature maps from the

previous layer and selects the average value. On the other hand, in the case of max-pooling, the highest value is selected from the N×N patches of the feature maps. Therefore, the output map dimensions are reduced by n times. In some special cases, each output map is multiplied with a scalar. Some alternative sub-sampling layers have been proposed, such as fractional max-pooling layer and subsampling with convolution.

4.3.2.4 Classification Layer

This is the fully connected layer which computes the score of each class from the extracted features from a convolutional layer in the preceding steps. The final layer feature maps are represented as vectors with scalar values which are passed to the fully connected layers. The fully connected feed-forward neural layers are used as a soft-max classification layer. There are no strict rules on the number of layers which are incorporated in the network model. However, in most cases, two to four layers have been observed in different architectures including LeNet, AlexNet, and VGG Net. As the fully connected layers are expensive in terms of computation, alternative approaches have been proposed during the last few years. These include the global average pooling layer and the average pooling layer which help to reduce the number of parameters in the network significantly. In the backward propagation through the CNNs, the fully connected layers update following the general approach of fully connected neural networks (FCNN). The filters of the convolutional layers are updated by performing the full convolutional operation on the feature maps between the convolutional layer and its immediate previous layer. Fig.4.5. shows the basic operations in the convolution and sub-sampling of an input image.

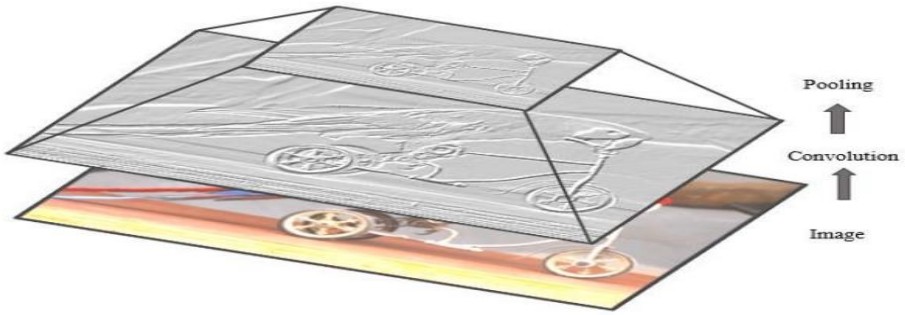


Fig.4.5. Example of convolution and pooling operation.

4.3.2.5 Network parameters and required memory for CNN

The number of computational parameters is an important metric to measure the complexity of a deep learning model. The size of the output feature maps can be formulated as follows:

$$\mathbf{M = \frac{(N-F)}{S} + 1} \qquad \textit{(16)}$$

Where *N* refers to the dimensions of the input feature maps, *F* refers to the dimensions of the filters or the receptive field, *M* refers to the dimensions of output feature maps, and *S* stands for the stride length. Padding is typically applied during the convolution operations to ensure the input and output feature map have the same dimensions. The amount of padding depends on the size of the kernel. Equation 17 is used for determining the number of rows and columns for padding.

$$\mathbf{P = (F - 1) /2} \qquad \textit{(17)}$$

Here *P* is the amount of padding and *F* refers to the dimension of the kernels. Several criteria are considered for comparing the models. However, in most of the cases, the number of network parameters and the total amount of memory are considered. The number of parameters ($\mathbf{Parm_l}$) of $\mathbf{l^{th}}$layer is calculated based on the following equation:

$$\mathbf{Parm_l = (F \times F \times F\, M_{l-1}) \times F\, M_l} \qquad \textit{(18)}$$

If bias is added with the weights, then the above equation can be written as follows:

$$\mathbf{Parm_l = (F \times (F + 1) \times F\, M_{l-1}) \times F\, M_l} \qquad \textit{(19)}$$

Here the total number of parameters of l^{th} layer can be represented with P_l , FM_l is for the total number of output feature maps, and $FM_{l-1}$is the total number of input feature maps or channels. For example, let's assume the l^{th} layer has $FM_{l-1} = 32$ input features maps, $FM_l = 64$ output feature maps, and the filter size is F = 5. In this case, the total number of parameters with bias for this layer is

$$\mathbf{parm_i = (5 * 5 * 33) * 64 = 528{,}000}$$

Thus, the amount of memory (Mem_l) needs for the operations of the l^{th}layer can be expressed as:

$$\mathbf{Mem_l = (N_l \times N_l \times FM_l)} \qquad \textit{(20)}$$

4.3.3 Popular CNN architectures

We will now examine several popular state-of-the-art CNN architectures. In general, most deep convolutional neural networks are made of a key set of basic layers, including the convolution layer, the sub-sampling layer, dense layers, and the soft-max layer. The architectures typically consist of stacks of several convolutional layers and max-pooling layers followed by a fully connected and Soft Max layers at the end. Some examples of such models are Le Net, Alex Net, VGG Net, NiN and all convolutional (All Conv) .Other alternative and more efficient advanced architectures have been proposed including Google Net with

Inception, Residual Networks, Dense Net, and Fractal Net. The basic building components (convolution and pooling) are almost the same across these architectures. However, some topological differences are observed in the modern deep learning architectures. Of the many DCNN architectures, Alex Net, VGG, Google Net, Dense CNN and Fractal Net are generally considered the most popular architectures because of their state-of-the-art performance on different benchmarks for object recognition tasks. Among all of these structures, some of the architectures are designed especially for large scale data analysis (such as Google Net and Res Net), whereas the VGG network is considered a general architecture. Some of the architectures are dense in terms of connectivity, such Dense Net. Fractal Network is an alternative of Res Net.

4.3.3.1 Le Net (1998)

Although Le Net was proposed in the 1990s, limited computation capability and memory capacity made the algorithm difficult to implement until about 2010. LeCun, however, proposed CNNs with the back-propagation algorithm and experimented on handwritten digits dataset to achieve state of-the-art accuracies. His architecture is well known as LeNet5 .The basic configuration of LeNet-5 is (see Fig.4.6.) convolution (conv) layers, 2 sub-sampling layers, 2 fully connected layers, and an output layer with Gaussian connection. The total number of weights and multiply and Accumulates (MACs) are 431k and 2.3M respectively. As computational hardware started improving in capability, CNNs stated becoming popular as an efficient learning approach in the computer vision and machine learning communities.

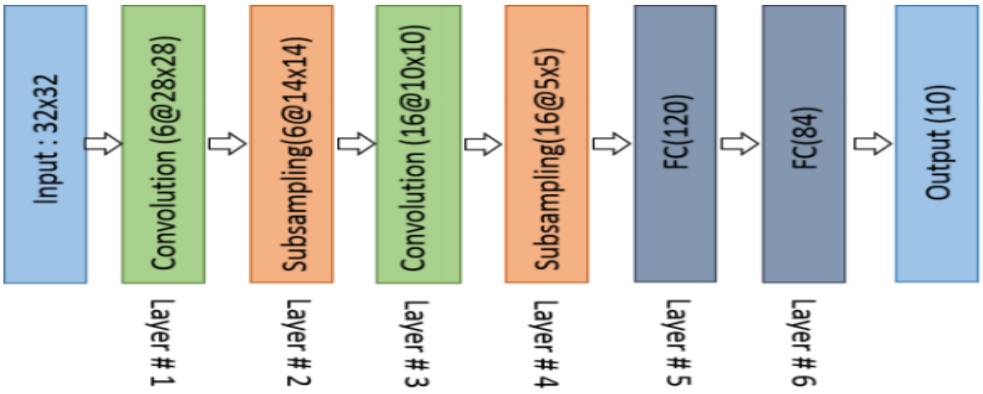


Fig.4.6. Architecture of LeNet

4.3.3.2 Alex Net (2012)

In 2012, Alex Krizhevesky and others proposed a deeper and wider CNN model compared to Le Net and won the most difficult Image Net challenge for visual object recognition called the Image Net Large Scale Visual Recognition Challenge (ILSVRC) in 2012. Alex Net achieved state-of-the-art recognition accuracy against all the traditional machine learning and computer vision approaches. It was a significant breakthrough in the field of machine learning

and computer vision for visual recognition and classification tasks and is the point in history where interest in deep learning increased rapidly. The architecture of Alex Net is shown in Fig. The first convolutional layer performs convolution and max pooling with Local Response Normalization (LRN) where 96 different receptive filters are used that are 11×11 in size. The max pooling operations are performed with 3×3 filters with a stride size of 2. The same operations are performed in the second layer with 5×5 filters. 3×3 filters are used in the third, fourth, and fifth convolutional layers with 384, 384, and 296 feature maps respectively. Two fully connected (FC) layers are used with dropout followed by a Soft max layer at the end. Two networks with similar structure and the same number of feature maps are trained in parallel for this model. Two new concepts, Local Response Normalization (LRN) and dropout, are introduced in this network. LRN can be applied in two different ways: first applying on single channel or feature maps, where an N×N patch is selected from same feature map and normalized based one the neighborhood values. Second, LRN can be applied across the channels or feature maps (neighborhood along the third dimension but a single pixel or location).

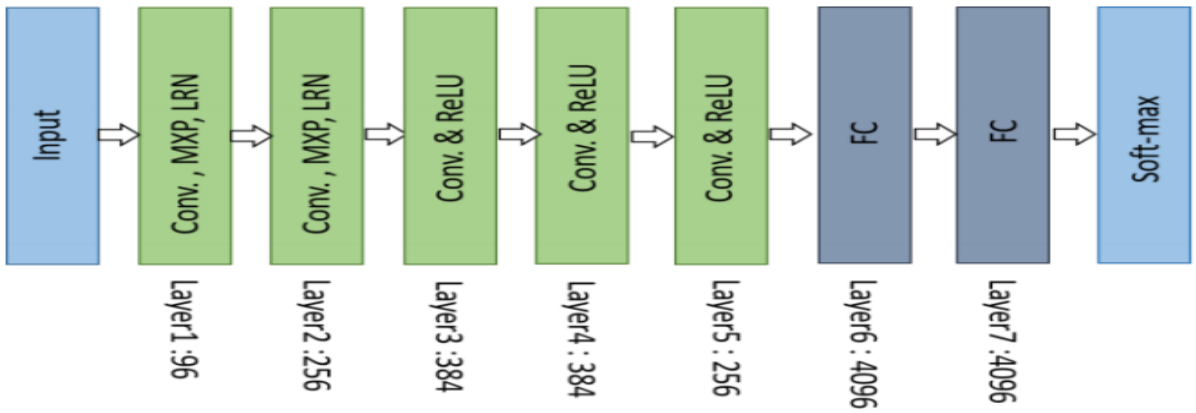


Fig.4.7. Architecture of AlexNet: Convolution, max-pooling, LRN and fully connected (FC) layer

Alex Net has 3 convolution layers and 2 fully connected layers. When processing the Image Net dataset, the total number of parameters for Alex Net can be calculated as follows for the first layer: input samples are 224×224×3, filters (kernels or masks) or a receptive field that has a size 11, the stride is 4, and the output of the first convolution layer is 55×55×96. According to the equations we can calculate that this first layer has 290400 (55×55×96) neurons and 364 (11 ×11×3 = 363 + 1 bias) weights. The parameters for the first convolution layer are 290400×364 = 105,705,600. Table II shows the number of parameters for each layer in millions. The total numbers of weights and MACs for the whole network are 61M and 724M respectively.

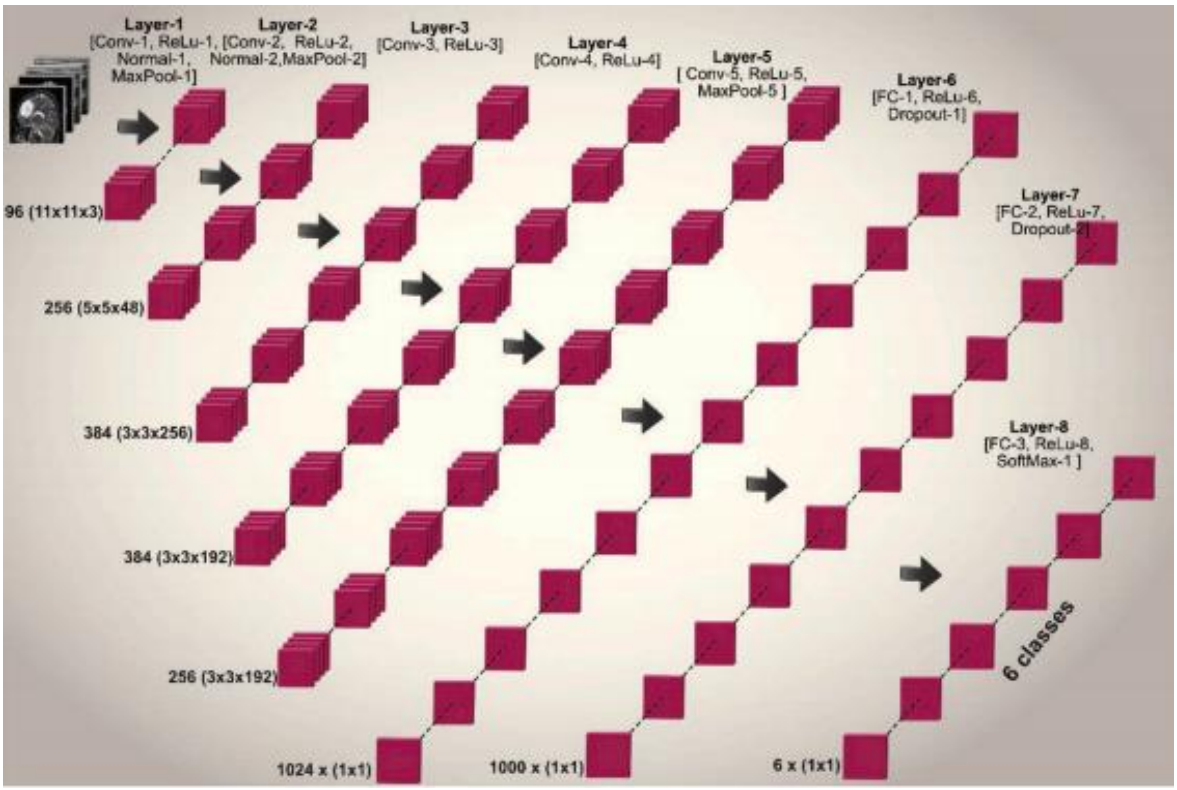


Fig.4.8. processing the Image Net dataset in Alex Net

4.3.3.3 Improved Alex Net

we propose to use batch normalization (BN) to improve the robustness of Alex Net for detecting abnormal brain. The distribution of the brain MRIs is complex because of the high variance of human brains. As a result, the distributions of inputs of the layers in Alex Net are different from layer to layer. This can make the parameter training extremely hard and time-consuming, which requires good initialization. In order to overcome this internal covariate shifting, BN is invented. The intuition behind BN is simple. As CNNs are trained in minibatch mode, BN uses normalization transform on the activations of layers to keep the means and variances fixed. For a random variable x and its values in a mini-batch S:

$$\mathbf{S} = [\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, \dots, \mathbf{x_n}] \qquad \textit{(21)}$$

The mean $\boldsymbol{\mu_S}$ and variance $\boldsymbol{\sigma_S^2}$ of x can be obtained by:

$$\boldsymbol{\mu_S} = \frac{\mathbf{1}}{\mathbf{n}} \sum_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{x_i} \qquad \textit{(22)}$$

$$\boldsymbol{\sigma_S^2} = \frac{\mathbf{1}}{\mathbf{n}} \sum_{\mathbf{i=1}}^{\mathbf{n}} (\mathbf{x_i} - \boldsymbol{\mu_S})^2 \qquad \textit{(23)}$$

So, the normalized values $\widehat{\mathbf{x_i}}$ can be obtained by:

$$\widehat{\mathbf{x_i}} = \frac{\mathbf{x_i} - \boldsymbol{\mu_S}}{\sqrt{\boldsymbol{\sigma_S^2} + \boldsymbol{\varepsilon}}} \qquad \textit{(24)}$$

Where **ε** denotes a constant value to increase the numerical stability. Nevertheless, the normalized activations may not be the learning target of the layers in some cases. So, a transformation is added to the result:

$$\mathbf{y_i} = \boldsymbol{\gamma}\widehat{\mathbf{x}_\mathbf{i}} + \alpha \qquad (25)$$

Where γ and **α** are two learnable parameters of minibatch S. With BN, the training speed of deep CNNs can be accelerated and gradients are less dependent on the initial values of parameters. Furthermore, BN can serve as a regularization, which improves the generalization ability of deep networks.

4.3.3.4 ELM

The improved Alex Net can yield good classification performance, but its classification is dependent on the last several layers (mostly fully connected layers). We proposed to replace these layers with a more efficient classifier model: extreme learning machine, to further improve the detection accuracy. ELM is a training algorithm for single hidden layer feedforward network (SLFN), proposed by Guang-Bin and Qin-Yu .An SLFN contains merely three layers, namely input layer, hidden layer and output layer, shown in Fig. The w and $\boldsymbol{\beta}$ are the input and output weights, respectively, and b denotes the bias in hidden nodes. The x and o represent the input and output, respectively. The advantage of ELM is that it is trained without iteration, which makes it converges extremely faster than traditional BPNN , and the generalization ability of ELM is also promising .The training algorithm of ELM contains only three steps. Given a training set M:

$$\mathbf{M} = [(\boldsymbol{x}_1, \boldsymbol{t}_1), (\boldsymbol{x}_2, \boldsymbol{t}_2), (\boldsymbol{x}_3, \boldsymbol{t}_3), \ldots, (\boldsymbol{x}_n, \boldsymbol{t}_n)] \qquad (26)$$

Where $\boldsymbol{x}_i$ represents the input vector and t_i denotes the label, ELM firstly initializes the input weight w and bias b with random values. Then, the output matrix of hidden layer H can be calculated:

$$\mathbf{H} = \sum_{i=1}^{\hat{N}} g_i(\boldsymbol{w}_i \boldsymbol{x}_j + \boldsymbol{b}_i), j = 1, \ldots, n \qquad (27)$$

Where $\hat{N}$ denotes the number of hidden nodes and g(.) denotes the activation function in hidden layer. Finally, our target is to achieve the ELM output equals to the actual sample labels

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T} \qquad (28)$$

Where $\mathbf{T} = (\boldsymbol{t}_1, \boldsymbol{t}_2, \boldsymbol{t}_3, \ldots, \boldsymbol{t}_n)^T$ .So, the $\boldsymbol{\beta}$ can be obtained by Moore–Penrose pseudo-inverse:

$$\boldsymbol{\beta} = \mathbf{H}^{\dagger}\mathbf{T} \qquad (29)$$

Where $H^{\dagger}$ represents the pseudo-inverse of H. The training algorithm is summarized in Table From the above analysis, it is clear that the training of ELM is simple to implement. So, ELM is widely applied in practical applications, like recognition, prediction

Table Training algorithm of ELM

Pseudocode of ELM	
Input	Training set in Eq. ***(26)***
Step1	Initialize the input weight and bias randomly
Step2	Calculate the output matrix of hidden layer by Eq. ***(27)***
Step3	Obtain the output weight by Eq. ***(29)***
Output	Trained ELM model

And clustering. Therefore, we employ ELM in this study to replace the last several layers for brain MRIs classification. However, the random input weight and bias can have a bad effect on the robustness of the ELM performance; we hope to further optimize these parameters. So, chaotic bat algorithm is proposed to handle this problem.

4.3.3.5 SNN

We also employed Schmidt neural network (SNN) and random vector functional link (RVFL) net as classifiers to compare with ELM. SNN and RVFL are both random neural networks, but their structures are different. SNN was proposed by Schmidt and Kraaijveld . There are three layers in SNN, shown in Fig. The weights from input layer to hidden layer were randomly assigned, and there are biases in both hidden layer and output layer. The output of SNN can be expressed as

$$\sum_{i=1}^{\hat{N}} \left[\boldsymbol{\beta}_i g\left(\boldsymbol{w}_i \boldsymbol{x}_j + b_i\right) + \boldsymbol{b}\right] = \boldsymbol{o}, \quad \boldsymbol{j} = 1, \ldots, \boldsymbol{N} \tag{30}$$

Where N denotes the number of hidden nodes. The training of SNN is similar to training of ELM, which can be implemented by pseudo-inverse to get the output weights β.

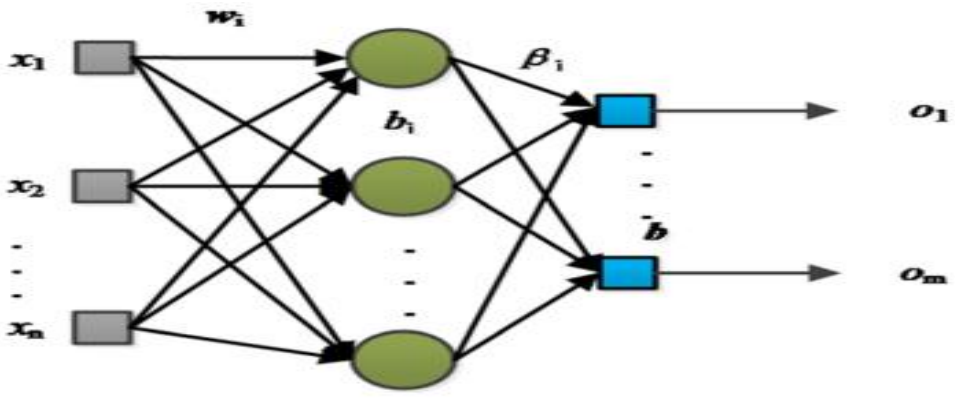


Fig.4.9. Architecture of SNN

4.3.3.6 RVFL

RVFL was proposed by Pao and Park, which is different from ELM and SNN. RVFL firstly maps the input features to enhancement space with random weights and biases. Then, the input features and enhanced features are concatenated to form the feature vector. This structure looks like the shortcut shown in Fig.4.10, which is similar to the modules in ResNet. Finally, output weights b are obtained by pseudo-inverse like ELM and SNN.

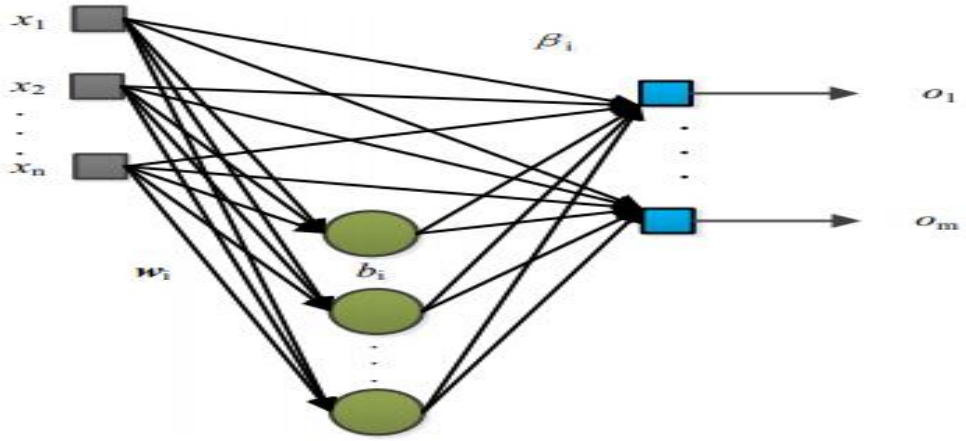


Fig.4.10. Architecture of RVFL

4.3.3.7 Chaotic bat algorithm

Chaotic bat algorithm (CBA) belongs to a swarm intelligent optimization method, which is evolved from bat algorithm. Inspired by the echolocation behavior of bats, CBA uses a set of bats with potential solutions to search the solution space by certain strategies. In every iteration, the parameters of the bats will be updated including the position, velocity and frequency based on the optimal solution found so far. The bat algorithm is better than traditional PSO for optimization, and we introduce chaotic map to improve its searching ability. Chaotic map is used in updating the positions of bats in our CBA. There are various chaotic maps, and we choose four maps for optimization: sine map, cosine map, Gaussian map and logistic map . The formulae are presented below.

Sine map:

$$x_{k+1} = \mu \sin(\pi x_k) \qquad (31)$$

Where k denotes the iteration time and μ the parameter ranging from 0 to 1. Where μ represents the parameter ranging from 0 to 1.

Cosine map:

$$x_{k+1} = \mu \sin(\pi x_k) \qquad (32)$$

aussian map:

$$x_{k+1} = \exp(-\alpha x_k^2) + \beta \qquad (33)$$

Where α and β are two parameters of real values.

Logistic map:

$$x_{k+1} = rx_k(1 - x_k) \qquad (34)$$

Where r denotes the parameter of positive integer values.

CBA firstly initializes the parameters in the bats with random values. Then, in each iteration, all the bats search the solution space with their velocities and update the solutions using chaotic maps. The best solution in that iteration is obtained by sorting. The iterations will continue until the stop criterion is met. A brief diagram of the CBA is given in Fig.4.11.

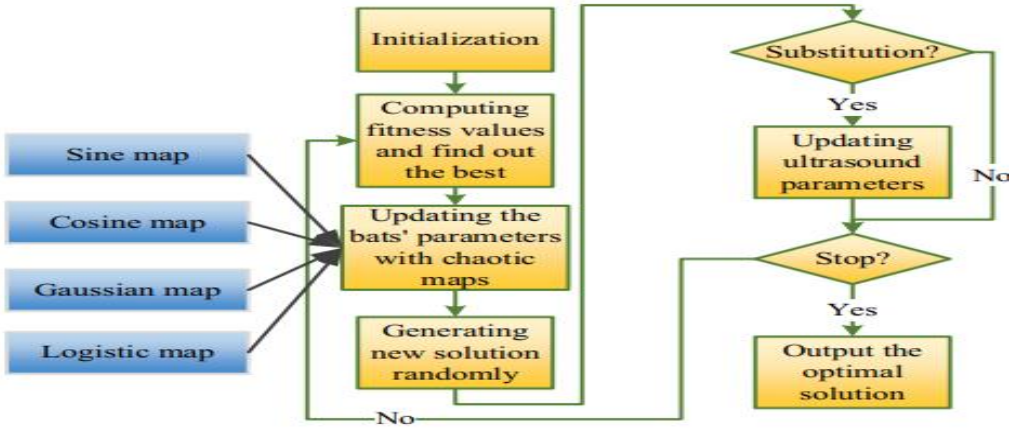


Fig.4.11. CBA optimization

4.3.3.8 BN-Alex Net-ELM-CBA

We propose the abnormal brain detection method based on batch normalized Alex Net, extreme learning machine and chaotic bat algorithm, abbreviated as BN-Alex Net-ELMCBA. First of all, we employ a pre-trained Alex Net for extracting image feature from brain MRIs. We add the batch normalization layers in the Alex Net model to handle the internal covariate shifting problem. We also modify the last three layers because the original output contains 1000 nodes, but our brain images have only two categories: abnormal and healthy. Totally, six BN layers are added into Alex Net, mainly located after the convolution and pooling layers. The original fully connected layer 'fc8' is also replaced by two new fully connected layers. Because the output matrix dimension of 'drop7' is 4096 9 1, and the original 'fc8' in Alex Net contains 1000 nodes, but our abnormal brain detection is a binary problem. So, we use two layers 'fc8' and 'fc9' to gradually shrink the dimensions, and the dimensions for 'fc8' and 'fc9' are 256 9 1 and 2 9 1, respectively. We also construct a transfer Alex Net (T-Alex Net) for performance comparison. T-Alex Net is built by removing all the batch normalization layers in BN-Alex Net. The three deep CNN structures are given in Fig.4.12

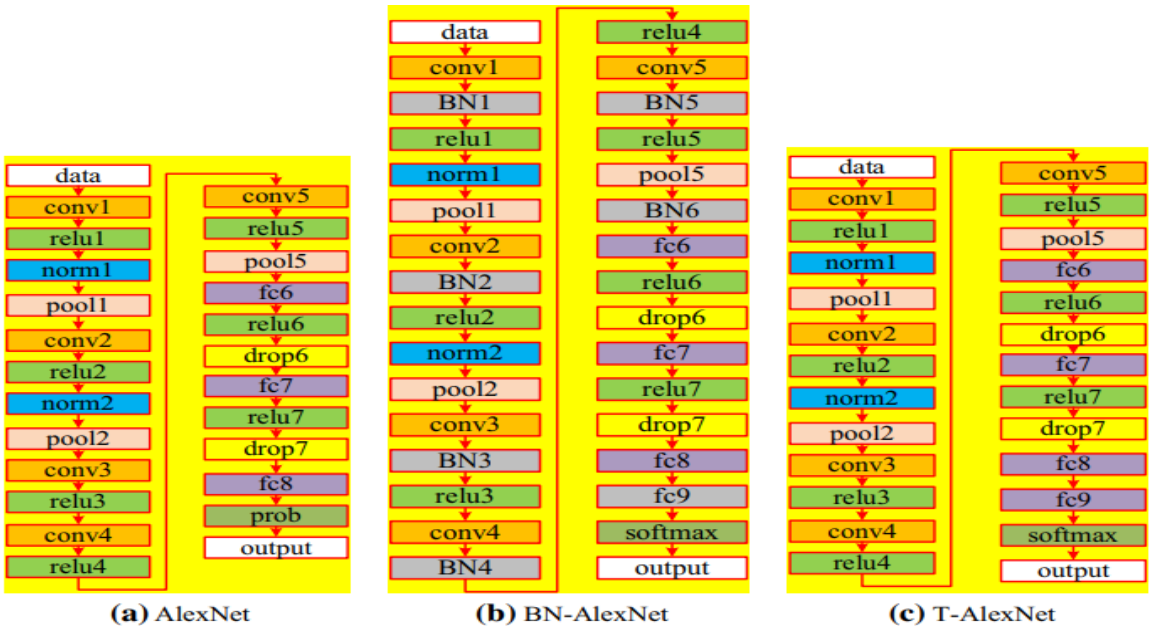


Fig.4.12. The structures of AlexNet and BN-AlexNet

Then, the last several layers in BN-Alex Net are replaced by an ELM classifier. To obtain the optimal layers to be substituted, we proposed to search it by the classification performance. We test the accuracy of our system with n replaced layers based on 5 9 hold-out validation and select the best one. The searching algorithm is given in Table.

Table: Searching algorithm for optimal layers to be replaced

Pseudocode of searching algorithm for the optimal layers to be replaced	
Input	Training set, testing set and the trained BN-AlexNet
Step1	For n = 2 to 6, run Step2 to Step5
Step2	Replace the last n layers in BN-AlexNet with ELM model
Step3	Train the parameters in ELM using chaotic bat algorithm on training set
Step4	Obtain the BN-AlexNet-ELM-CBA classification performance on testing set
Step5	Run from Step2 to Step5 for five times and compute the average accuracy with that n value
Step6	Compare the average performance with different n values and find the best number
Output	The optimal number of layers to be replaced

In chaotic bat algorithm optimization for the ELM, the bats contain the input weight **w** and bias **b** of the ELM. The fitness function f(.) of CBA is the squared error of predicted labels and actual labels:

$$f(\boldsymbol{w},\boldsymbol{b}) = \sum_{i=1}^{n} (\boldsymbol{o}_i - \boldsymbol{t}_i)^2 \qquad (35)$$

Where $\boldsymbol{o}_i$ and $\boldsymbol{t}_i$ stand for the output of ELM and the image label, respectively, and **n** denotes the number of training samples. The solutions in bats are updated with their velocities and chaotic maps:

$$x_i^t = x_i^{t-1} + v_i^t + \lambda \times \text{chaotic}\left(x_i^{t-1}\right) \qquad (36)$$

Where x_i^t denotes the solution of the ith bat in tth iteration, and λ is the weighting parameter ranging from 0 to 1. In this paper, k is set as 0.3. All the evaluation is carried out based on 5 9 hold-out validation, i.e., we run the systems for five times and calculate the average classification performance for comparison. The pseudocode of our BN Alex Net-ELM-CBA is presented in Table 3, and a brief diagram is illustrated in Fig. Our method provides a general framework using off-the-shelf deep learning models. The system can be used in other image classification tasks by simple parameter tuning.

Table: Training of BN-AlexNet-ELM-CBA

Pseudocode of BN-AlexNet-ELM-CBA	
Input	Training set and testing set
Step1	Modify the architecture of a pre-trained AlexNet and obtain the BN-AlexNet
Step2	Train the BN-AlexNet using training set
Step3	Replace the last *n* layers in BN-AlexNet by ELM structure
Step4	Train the weight and bias in ELM on training set by chaotic bat algorithm
Step5	Obtain the BN-AlexNet-ELM-CBA generalization ability on testing set
Step6	Run from Step1 to Step5 for five times
Output	The optimal number of layers to be replaced

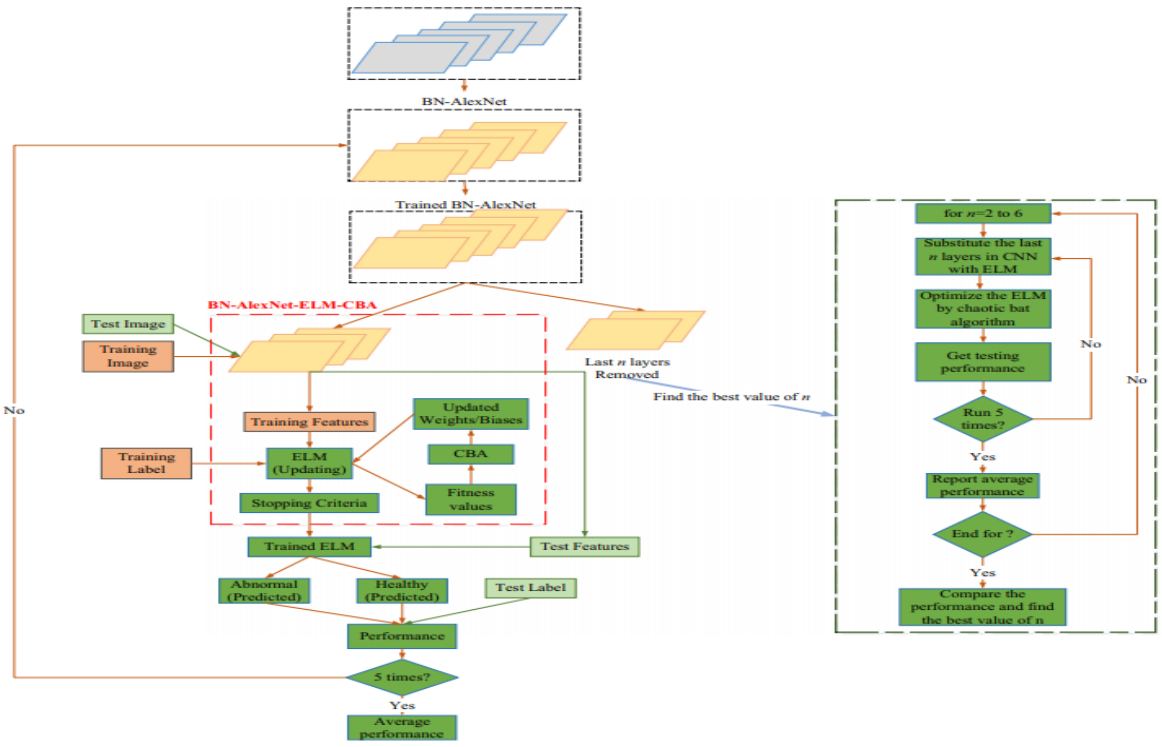


Fig.4.14. Flowchart of our BN-AlexNet-ELM-CBA

4.3.3.9 ZF Net / Clarifai (2013)

In 2013, Matthew Zeiler and Rob Fergue won the 2013 ILSVRC with a CNN architecture which was an extension of Alex Net. The network was called ZF Net, after the authors' names. As CNNs are expensive computationally, an optimum use of parameters is needed from a model complexity point of view. The ZF Net architecture is an improvement of Alex Net, designed by tweaking the network parameters of the latter. ZF Net uses 7x7 kernels instead of 11x11 kernels to significantly reduce the number of weights. This reduces the number of network parameters dramatically and improves overall recognition accuracy.

4.3.3.10 Network in Network (NiN)

This model is slightly different from the previous models where a couple of new concepts are introduced. The first concept is to use multilayer perception convolution, where convolutions are performed with a 1×1 filters that help to add more nonlinearity in the models. This helps to increase the depth of the network, which can then be regularized with dropout. This concept is used often in the bottleneck layer of a deep learning model. The second concept is to use Global Average Pooling (GAP) as an alternative of fully connected layers. This helps to reduce the number of network parameters significantly. GAP changes the network structure significantly. By applying GAP on a large feature map, we can generate a final low dimensional feature vector without reducing the dimension of the feature maps.

4.3.3.11 VGGNET (2014)

The Visual Geometry Group (VGG) was the runner up of the 2014 ILSVRC. The main contribution of this work is that it shows that the depth of a network is a critical component to achieve better recognition or classification accuracy in CNNs. The VGG architecture consists of two convolutional layers both of which use the ReLU activation function. Following the activation function is a single max pooling layer and several fully connected layers also using a ReLU activation function. The final layer of the model is a Soft max layer for classification. In VGG-E the convolution filter size is changed to a 3x3 filter with a stride of 2. Three VGG-E models, VGG11, VGG-16, and VGG-19; were proposed the models had 11, 16,and 19 layers respectively All versions of the VGG-E models ended the same with three fully connected layers. However, the number of convolution layers varied VGG-11 contained 8 convolution layers, VGG-16 had 13 convolution layers, and VGG-19 had 16 convolution layers. VGG-19, the most computational expensive model, contained 138Mweights and had 15.5M MACs.

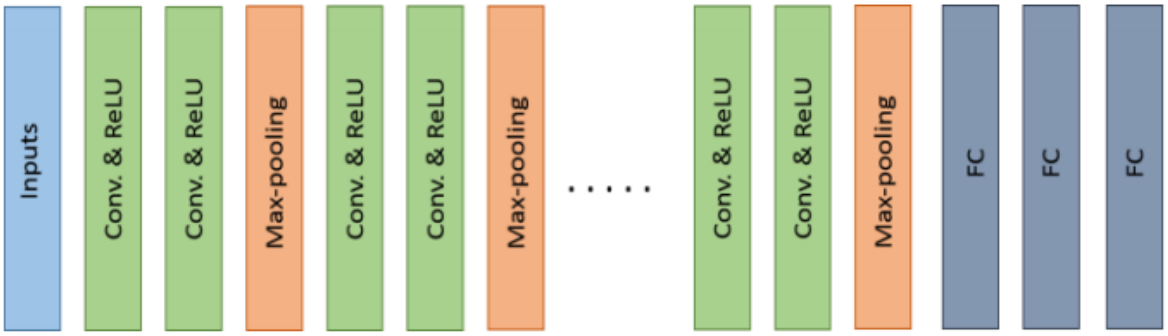


Fig.4.13 Basic building block of VGG network: Convolution (Conv) and FC for fully connected layers

4.3.3.12 VGG16 and VGG19

The visual representation of both models is shown in Figures, respectively. The VGG16 model consisted of 12 convolution layers, 15 ReLu activation layers, five max-pooling layers, three fully connected (FC) layers, and one Soft max layer, as a classification layer. The input layer size was 224 × 224 × 3. The number of filters in the first convolution layer was 64, and the filter size was 3 × 3 × 3, along with a stride of 1 × 1. In the next convolution layer, the number of filters was not updated but the filter size was updated to 3 × 3 × 64. Further, the dimension of learnable weights was 3 × 3 × 64 × 64, which were 3 × 3 × 3 × 64 in the first convolution layer. The learnable weights of each convolution layer were updated according to the number of filters and the filter size. In the first max-pooling layer, a 2 × 2 filter size was opted along with the same stride 2 × 2. After the convolution layers, three FC layers were added. The learnable weights dimension of the first FC layer was 4096 × 25088. After a 50% dropout, the weights matrix size of the second FC layer was 4096 × 4096. Another dropout layer was added and a ratio of 50% was set. The resultant weight matrix used as an input of the third layer (denoted as FC8) returned a weight matrix of dimension 1000 × 4096. Finally, the Soft max function and the classification layers were added for the final classification.

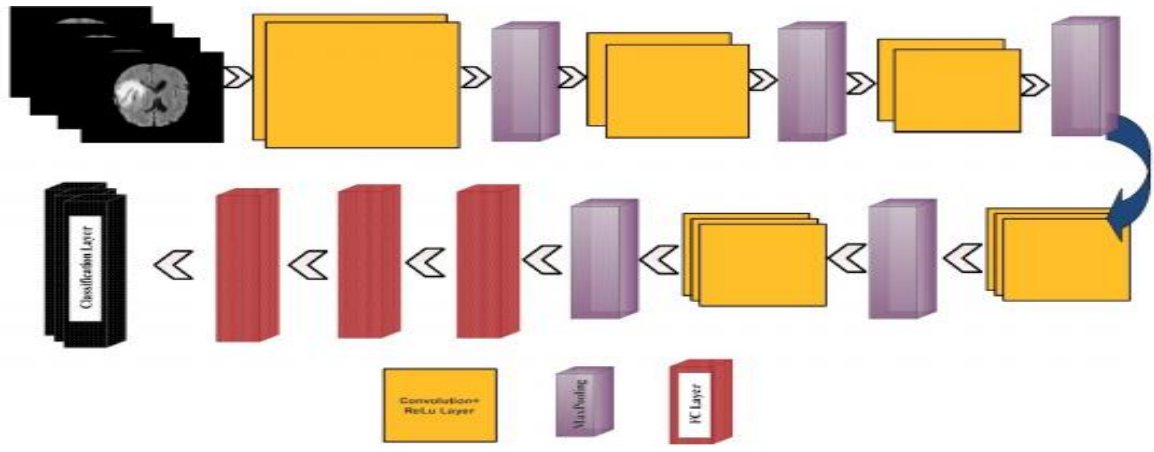


Fig.4.14. A layered wise architecture of the VGG16 deep learning model

VGG19 model consists of a series of 16 convolution layers, 19 ReLu activation layers, four max-pooling layers, three FC layers, and one Soft max layer as a classification layer. The input layer size was 224 × 224 × 3. The number of filters in the first convolution layer was 64, and the filter size was 3 × 3 × 3. This filter size was updated according to the number of filters. In the first max-pooling layer, a 2 × 2 filter size was opted along with the same stride. After the convolution layers, three FC layers were added. The weights dimension of the first FC layer was 4096 × 25088. After a 50% dropout, the weights matrix size of the second FC layer was 4096 × 4096. The resultant weight matrix used as an input of the third layer (denoted as FC8) returned a weight matrix of dimension 1000 × 4096.

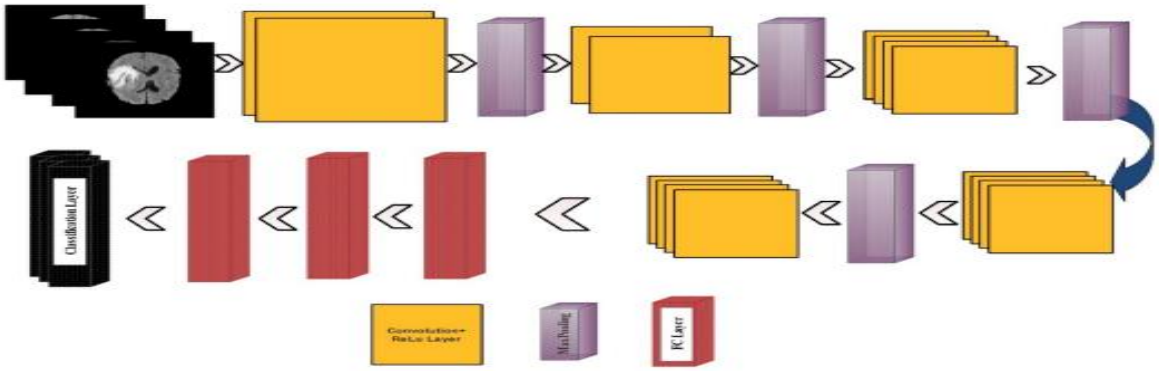


Fig.4.15. A layered wise architecture of the VGG19 deep learning model

4.3.3.13 Google Net (2014)

Google Net, the winner of ILSVRC 2014, was a model proposed by Christian Szegedy of Google with the objective of reducing computation complexity compared to the traditional CNN. The proposed method was to incorporate "Inception Layers" that had variable receptive fields, which were created by different kernel sizes. These receptive fields created operations that captured sparse correlation patterns in the new feature map stack.

The initial concept of the Inception layer can be seen in Fig. Google Net improved the state-of-the-art recognition accuracy using a stack of Inception layers seen in Fig. The difference between the naïve inception layer and final Inception Layer was the addition of 1x1 convolution kernels. These kernels allowed for dimensionality reduction before computationally expensive layers. Google Net consisted of 22 layers in total, which was far greater than any network before it. However, the number of network parameters Google Net used was much lower than its predecessor Alex Net or VGG. Google Net had 7M network parameters when Alex Net had 60M and VGG-19 138M. The computations for Google Net also were 1.53G MACs far lower than that of Alex Net or VGG

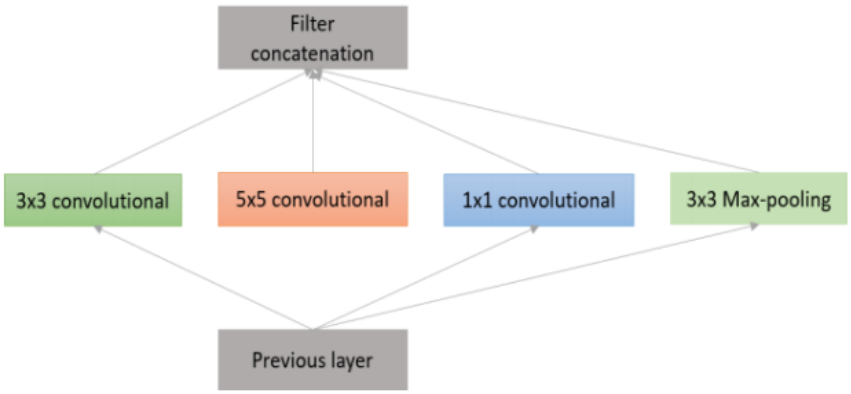


Fig.4.16. Inception layer: naive version

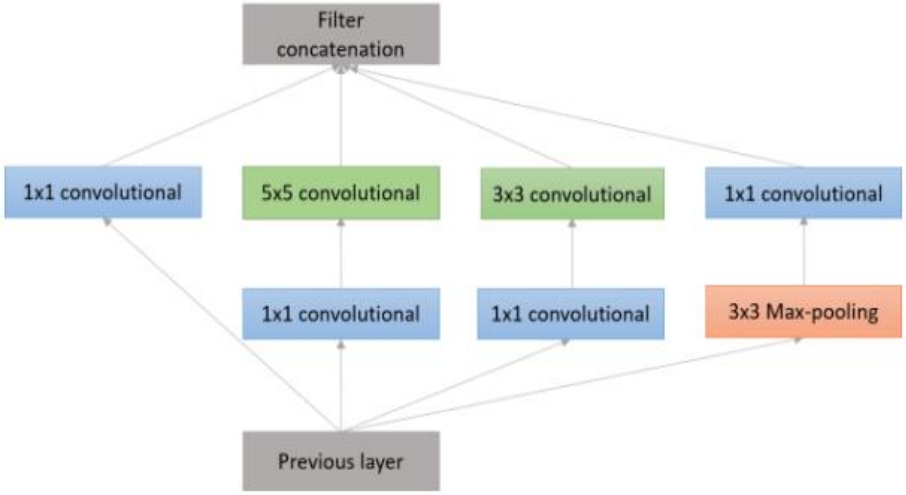


Fig.4.17. Inception layer with dimension reduction

4.3.3.14 Residual Network (Res Net in 2015)

The winner of ILSVRC 2015 was the Residual Network architecture, Res Net. Res Net was developed by Kaiming He with the intent of designing ultra-deep networks that did not suffer from the vanishing gradient problem that predecessors had. Res Net is developed with many different numbers of layers; 34, 50,101, 152, and even 1202. The popular ResNet50 contained 49 convolution layers and 1 fully connected layer at the end of the network. The total numbers of weights and MACs for the whole network are 25.5M and 3.9M respectively.

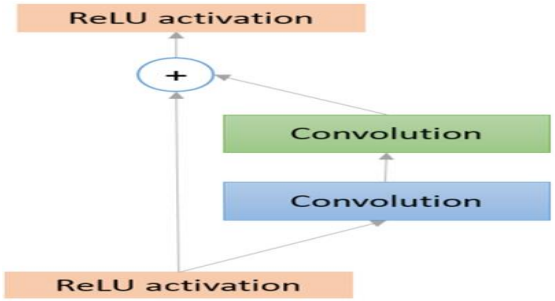


Fig.4.18. Basic diagram of Residual block

The basic block diagram of the Res Net architecture is shown in Fig. Res Net is a traditional feed forward network with a residual connection. The output of a residual layer can be defined based on the outputs of $(l-1)^{th}$ which comes from the previous layer defined as x_{l-1}. $\mathcal{F}(x_{l-1})$ is the output after performing various operations (e.g. convolution with different size of filters, Batch Normalization (BN) followed by an activation function such as a ReLU on x_{l-1}). The final output of residual unit is x_l which can be defined with the following equation:

$$x_l = \mathcal{F}(x_{l-1}) + x_{l-1} \qquad (37)$$

The residual network consists of several basic residual blocks. However, the operations in the residual block can be varied depending on the different architecture of residual networks. The wider version of residual network was proposed by Zagoruvko el at. In 2016. Another improved residual network approach known as aggregated residual transformation was proposed in 2016. Recently, some other variants of residual models have been proposed based on the Residual Network architecture. Furthermore, there are several advanced architectures that have been proposed with the combination of Inception and Residual units. The basic conceptual diagram of Inception-Residual unit is shown in the following Fig.4.19

- *تتكون الشبكة المتبقية من عدة كتل أساسية متبقية. ومع ذلك ، يمكن أن تتنوع العمليات في الكتلة المتبقية اعتمادًا على البنية المختلفة للشبكات المتبقية. اقترح* Zagoruvko el at *النسخة الأوسع من الشبكة المتبقية. في عام 2016. تم اقتراح نهج شبكة متبقية محسّن آخر يُعرف باسم التحويل المتبقي التجميعي في عام 2016. مؤخرًا ، تم اقتراح بعض المتغيرات الأخرى للنماذج المتبقية بناءً على بنية الشبكة المتبقية. علاوة على ذلك ، هناك العديد من البنى المتقدمة التي تم اقتراحها مع الجمع بين وحدات التأسيس والوحدات المتبقية. يظهر الرسم التخطيطي المفاهيمي الأساسي للوحدة الاستهلالية-المتبقية في الشكل*

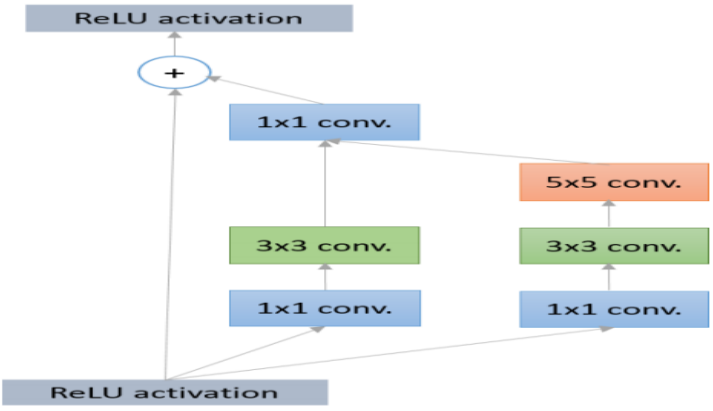


Fig.4.19. Basic block diagram for Inception Residual unit

Mathematically, this concept can be represented as

$$x_l = \mathcal{F}(x_{l-1}^{3\times3} \odot x_{l-1}^{5\times5}) + x_{l-1} \qquad (38)$$

Where the symbol ⊙ refers the concentration operations between two outputs from the 3×3 and 5×5 filters. After that the convolution operation is performed with 1×1 filters. Finally, the

outputs are added with the inputs of this block of x_{l-1} .The concept of Inception block with residual connections is introduced in the Inception-v4 architecture . The improved version of the Inception-Residual network known as Poly Net was recently proposed

Deep CNN can be trained more effectively. Traditionally, a CNN contains three types of layers: convolution layer, pooling layer and fully connected layer. Convolution operation generates feature maps from the input by a set of filters with trainable weights. Figure presented a toy example of convolution, in which a 4 × 4 image was convoluted by a 2 × 2 filter with stride 2, and the feature map is 2 × 2. Given an image **I** in size of (**M**, **N**) and a filter **F** in size of (**p**, **q**), the convolution formula can be expressed as

$$conv=(I*F)(x,y)=\sum M\sum NI(x-p,y-q)F(p,q) \qquad (39)$$

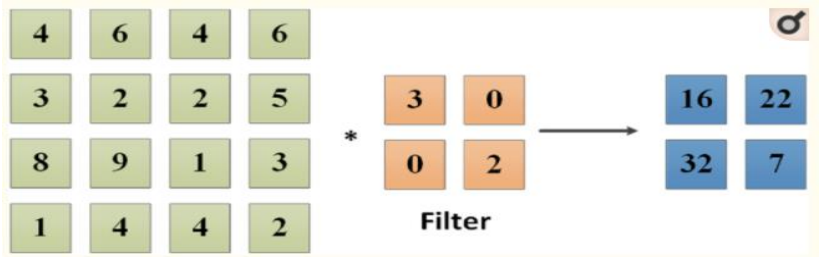


Fig.4.20. Convolution operation

The filters sweep from the top left to bottom right with some stride, and the feature maps can be computed. But the map size will shrink inevitably and the edge information can be lost during convolution. As in Figure , the input was 4 × 4, but the feature map was 2 × 2. To handle this problem, zero padding can be employed. Zero padding adds zero valued pixels around the input image before convolution, shown in Figure . A 4 × 4 image with 1 × 1 padding was convoluted by a 3 × 3 filter with 1 stride, and a 4 × 4 feature map was obtained. The detailed relationship of input size and feature map size is given below:

{height map = (height input – heigh filter + 2 × padding) stride + 1widthmap = (width input -width filter + 2 × padding) stride + 1 }

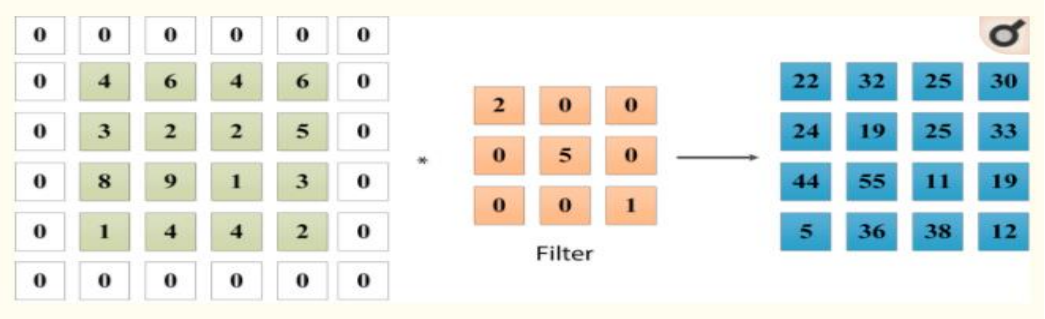


Fig.4.21. Convolution with zero padding

Pooling is a simple but effective operation, and pooling layers are usually placed after convolution layers. Pooling generates feature maps with a local perceptive field. Pooling layers can extract main features from the input and reduce the dimension, which helps accelerate the training and improve the generalization ability. An example is given in Figure , which illustrates three pooling strategies: min, average and max with the local perceptive field of 2 × 2.

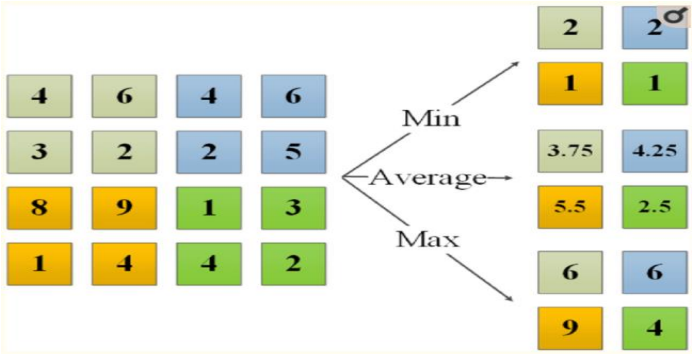


Fig.4.22. Pooling methods

The fully connected layers are usually arranged at the end of a CNN structure, which was used for classification and recognition. Every node in fully connected layers is linked to all the nodes in the adjacent fully connected layers with trainable weights, shown in Figure .

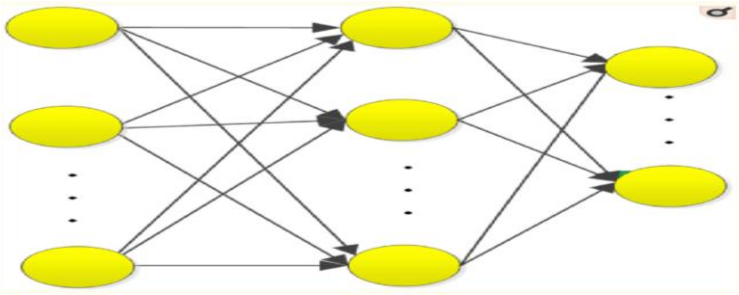


Fig.4.21. Fully connected layer

The activation function is an important part of neural networks, which offers nonlinearity. The network may become a linear system composed of matrix multiplications without activation functions. In classical BPNN, the sigmoid function was the most used, which can be expressed as:

$$S(x) = 11 – e - x \qquad (40)$$

It is widely used because BPNN is trained by gradient descent algorithms, and the gradient of sigmoid function is computationally effective, which can be calculated by:

$$S'(x)=S(x)(1-S(x)) \qquad (41)$$

However, in deep learning architecture, the sigmoid function doesn't work anymore because it causes the gradient vanishing problem in which the learning speed was very slow. So, in deep CNN, the rectified linear unit (ReLU) is often a good option. The formula of ReLU is:

$$ReLU(x)=max(x,0) \quad (42)$$

For positive input, the output of ReLU is always 1, and for negative input, the output is always 0. ReLU is simple to compute and it can accelerate the forward progress in training. For the final classification layer, Softmax function is often used:

$$softmax(x)i=exp(xi)\sum j=1nexp(xj) \quad (43)$$

Soft max function maps the input vector into probabilities, so the overflow can be avoided. Res Net was proposed by introducing the residual learning method. The idea is that network layers are capable of approximating any function asymptotically. For instance, **f**(x) denotes the learned mapping of several layers, then it is equivalent to train these layers to approximate the residual function: **r**(**x**) = **f**(**x**)-**x**. So, the target function becomes:

$$f(x)=r(x)+x \quad (44)$$

The analysis suggested that nonlinear layers may be difficult to reach identity mapping. So, if the identity mapping is optimal, the weights of theses stacked layers will be simply driven to zero with residual learning block, shown in Figure .

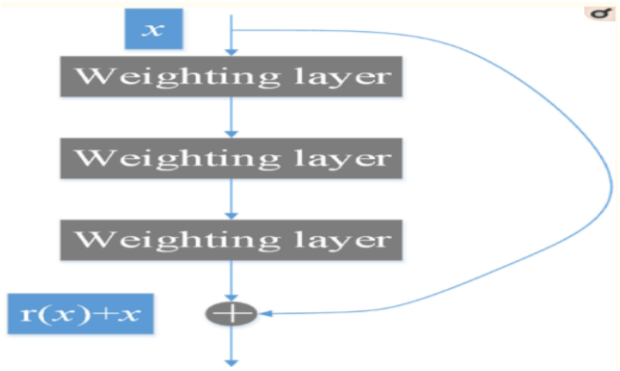


Fig.4.22. Residual block

Res Net has been widely evaluated on a variety of open datasets including CIFAR 10, Image Net classification, and COCO. The residual learning mechanism is a generic, easy to optimize and effective training method for deep CNN architectures.

4.3.3.15 ResNet50

ResNet50 is a 50-layer Residual Network with 26M parameters. The residual network is a deep convolutional neural network model that is introduced by Microsoft in 2015 .In Residual network rather than learning features, we learn from residuals that are subtraction of features learned from the layer's inputs. Res Net used the skip connection to propagate information across layers. Res Net connects nth layer input directly to some (n+x)th layer which enables additional layers to be stacked and a to establish a deep network. We used a pre-trained

ResNet50 model in our experiment and fine-tuned it. In Figure, the architecture of ResNet50 is shown.

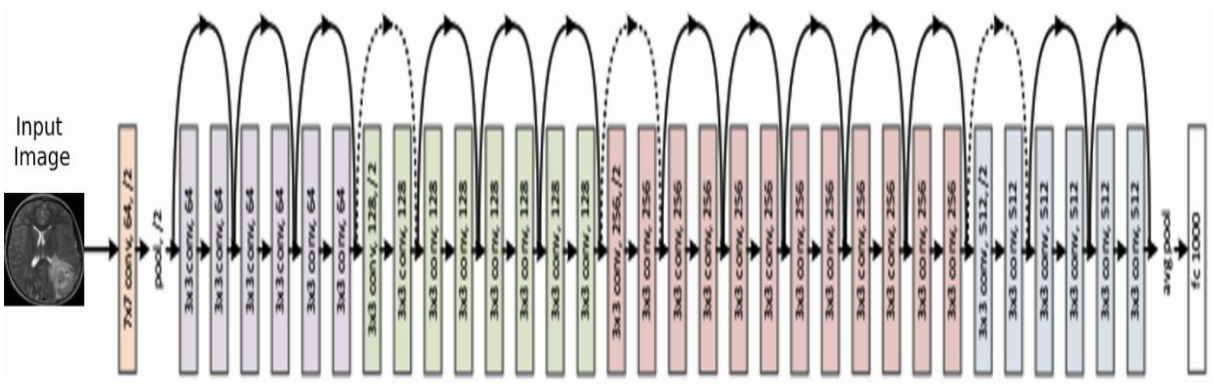


Fig.4.23. ResNet-50 model architecture.

4.3.3.16 ResNet-101

The design of Res Nets was inspired by VGG-19 model. It is one of the deepest proposed architectures for Image Net (Object detection and Image classification Challenge). Usually, in a CNN, several layers are connected to each other and are trained to perform various tasks. The network learns several levels of features at the end of its layers. The sizes of convolutional layers in this model have mostly 33 filters. In Res Net, the layers have the same number of filters for same output feature map size and the number of filters is doubled if the feature map size is halved so as to maintain the time complexity for every layer. It executes downsampling directly by convolving layers with a stride of two. This Res Net terminates with a global average pooling layer and a Soft Max activated fully connected layer. Res Net Module is illustrated in Fig(a). Residual learning can be easily interpreted as subtraction of input features learned from that layer. This is done by Res Net using shortcut connections to each pair of 33 filters, directly connecting the input of k^{th} layer to $(k+x)^{th}$ layer. The motive behind bypassing layers is to keep away the problem of vanishing gradients by reutilizing activations from the preceding layer till the layer next to the present one has learned its weights. While training the network, weights will amplify the layer next to the present one and will also adjust to mute the preceding layer. It has been observed that it is easier to train this network than training simple deep convolutional neural networks. It also resolves the problem of accuracy degradation. ResNet-101 is 101-layer Residual Network and is a modified version of the 50-layer Res Net.

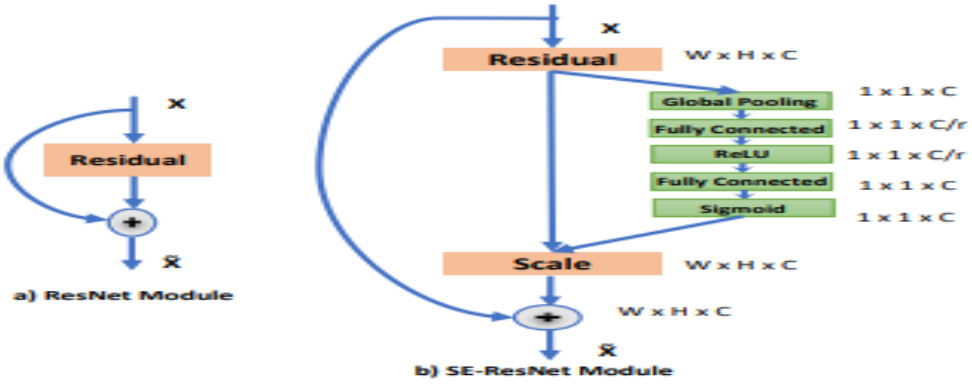


Fig.4.24. Block Diagram of a) Residual Module b) SE-ResNet module

4.3.3.17 Squeeze and Excitation block with Res Net

It was found that Squeeze and Excitation (SE) blocks combined with Res Net architecture, has performed outstandingly good in classifying the brain tumors. By stacking the SE blocks one after the other, SE-ResNet-101 architecture is constructed that generalise extremely well across our brain tumor dataset. For given transformation of a convolution or a set of convolutions, equation (45) is as follows:

$$Z_{tr} : X \rightarrow Y,\ X \epsilon R^{W_0 \times H_0 \times C_0},\ Y \epsilon R^{W \times H \times C} \qquad \textit{(45)}$$

Here width, height, and the number of channels is represented by W, H, and C respectively after taking X and Y as input and output feature maps and r signifies the reduction ratio. The input features X are first passed through a squeeze operation, which is used to average the feature maps in a spatial plane. After this, two fully connected layers with ReLU and Sigmoid activations are used separately for excitation operation. The Squeeze and Excitation block with ResNet-101.

4.3.3.18 Training of Model

SE-ResNet-101 model was trained from scratch and finely tuned to just fit training data, first without data augmentation containing 3064 samples in total, and then using data augmentation which increased the data size to 7771. The proposed method for classification of brain tumor used the SE-ResNet101 architecture with single channel SE block. The original dataset contained images of size 512×512×1 which were then pre-processed using the above-described methods to generate images of dimension 256 × 256 × 1 which focused only on the segmented tumors or ROI. The optimizer used for training purpose was Adam with a learning rate of 0.005 at first and then reduced by a factor of 0.2 up to 0.001 after every 3^{rd} epoch when the validation loss was not improved. Early Stopping, a Keras library module, was used to stop the training of model if validation loss didnt improve during 5 epochs consecutively. These regularization methods were used to prevent over fitting of our model. At first, the network

was trained without data augmentation till 12,196 iterations and then with data augmentation, the pre-trained model was again trained for 13,898 iterations. The preprocessed dataset was divided into training and testing parts as described. The test dataset was used for evaluating the performance of the model. The total duration of training was 7 hours with a batch size of 5 for 26,094 iterations. The model got automatically updated weights while training on the suitable features.

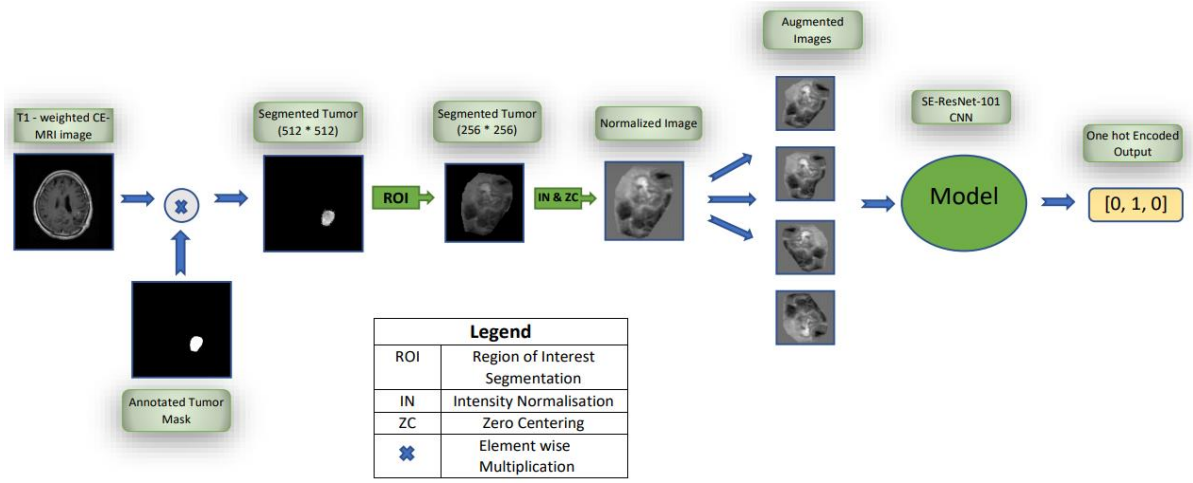


Fig.4.25. Block Diagram of the proposed methodology

4.3.3.19 Densely Connected Network (Dense Net)

Dense Net developed by Gao Huang and others in 2017, which consists of densely connected CNN layers, the outputs of each layer are connected with all successor layers in a dense block . Therefore, it is formed with dense connectivity between the layers rewarding it the name “Dense Net”. This concept is efficient for feature reuse, which dramatically reduces network parameters. Dense Net consists of several dense blocks and transition blocks, which are placed between two adjacent dense blocks. The conceptual diagram of a dense block is shown in Fig.4.26.

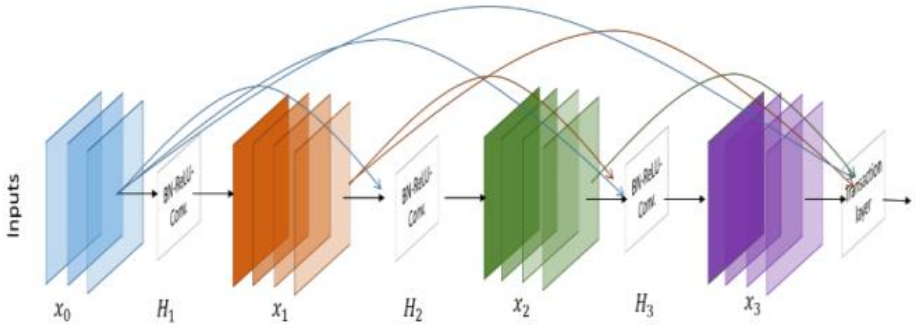


Fig.4.26. A 4-layer Dense block with block with growth rate of k=3

Each layer takes all the preceding feature maps as input. When deconstructing Fig. the l^{th} layer received all the feature maps from previous layers of $x_0, x_1, x_2 \cdots x_{l-1}$ as input:

$$x_l = H_l([x_0, x_1, x_2 \cdots x_{l-1}]) \qquad (46)$$

Where $[x_0, x_1, x_2 \cdots x_{l-1}]$ are the concatenated features for layers 0, ⋯ ⋯, $l - 1$ and H_l (·) is considered as a single tensor. It performs three different consecutive operations: Batch Normalization (BN), followed by a ReLU and a 3 × 3 convolution operation. In the transaction block, 1 × 1 convolutional operations are performed with BN followed by a 2 × 2 average pooling layer. This new model shows state-of the-art accuracy with a reasonable number of network parameters for object recognitions tasks.

4.3.3.20 Fractal Net (2016)

This architecture is an advanced and alternative architecture of Res Net model, which is efficient for designing large models with nominal depth, but shorter paths for the propagation of gradient during training. This concept is based on drop path which is another regularization approach for making large networks. As a result, this concept helps to enforce speed versus accuracy tradeoffs. The basic block diagram of Fractal Net is shown in Fig.4.27.

• *هذه البنية هي بنية متطورة وبديلة لنموذج* Res Net *، وهي فعالة في تصميم نماذج كبيرة بعمق اسمي ، ولكن مسارات أقصر لنشر التدرج أثناء التدريب. يعتمد هذا المفهوم على مسار الإسقاط وهو نهج تنظيم آخر لإنشاء شبكات كبيرة. نتيجة لذلك ، يساعد هذا المفهوم في فرض مقايضات السرعة مقابل الدقة. يظهر الرسم التخطيطي الأساسي لشبكة* Fractal *في الشكل.*Net

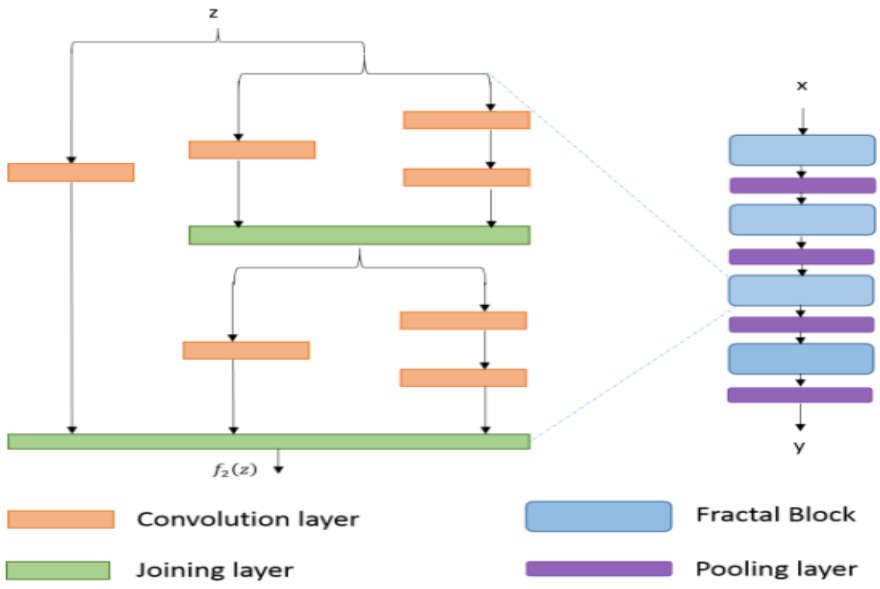


Fig.4.27. The detailed FractalNet module on the left and FractalNet on the right

4.3.4 Capsule Net

CNNs are an effect methodology for detecting features of an object and achieving good recognition performance compared to state-of-the-art hand-crafted feature detectors. There are limits to CNNs, which are that it does not take into account special relationships, perspective, size and orientation, of features, For example: if you have a face image, it does not matter the

placement of different components (nose, eye, mouth etc.) of the faces neurons of a CNN will wrongly active and recognition as face without considering special relationships (orientation, size). Now, imagine a neuron which contains the likelihood with properties of features (perspective, orientation, size etc.). This special type of neurons, capsules, can detect face efficiently with distinct information. The capsule network consists of several layers of capsule nodes. The first version of capsule network (Caps Net) consisted of three layers of capsule nodes in an encoding unit.

This architecture for MNIST (28×28) images, the 256 9×9 kernels are applied with a stride 1, so the output is (28 − 9 + 1 = 20) with 256 feature maps. Then the outputs are feed to the primary capsule layer which is a modified convolution layer that generates an 8-D vector instead of a scalar. In the first convolutional layer, 9×9 kernels are applied with stride 2, the output dimension is ((20 − 9)/2+ 1 = 6). The primary capsules are used 8×32 kernels which generates 32×8×6×6 (32 groups for 8 neurons with 6×6 size).

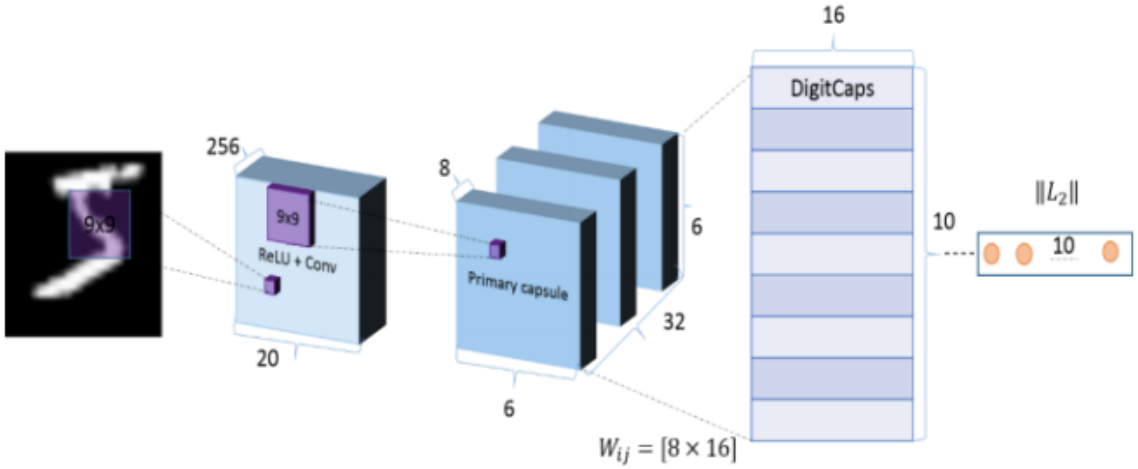


Fig.4.28. A CapsNet encoding unit with 3 layers. The instance of each class is represented with a vector of a capsule in DigitCaps layer that is used for calculating classification loss. The weights between primary capsule layer and DigitCaps layer are represented with W_{ij}

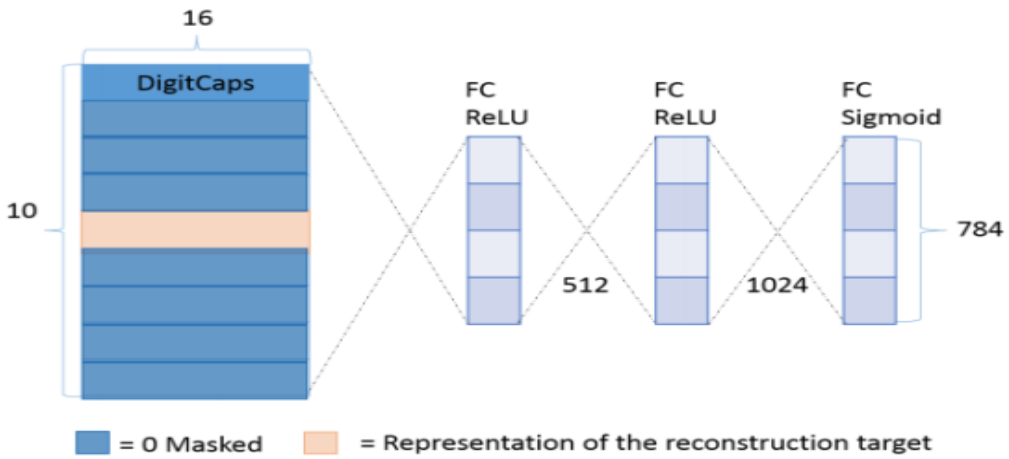


Fig.4.29. The decoding unit where a digit is reconstructed from DigitCaps layer representation. The Euclidean distance is used minimizing error between input sample and reconstructed sample from sigmoid layer. True labels are used for reconstruction target during training

The entire encoding and decoding processes of Caps Net is shown in Fig. and Fig. respectively. We used max pooling layer in CNN often that can handle translation variance. Even if a feature moves if it is still under a max pooling window it can be detected. As the capsule contains the weighted sum of features from the previous layer, therefore this approach is capable of detecting overlapped features which is important for segmentation and detection tasks. In the traditional CNN, we have used a single cost function to evaluate the overall error which propagates backward during training. However, in this case if the weight between two neurons is zero then the activation of a neuron is not propagated from that neuron. The signal is routed with respect to the feature parameters rather than a one size fits all cost function in iterative dynamic routing with agreement. For details about this architecture, please see . This new CNN architecture provides state-of-the-art accuracy for handwritten digits recognition on MNIST. However, from an application point of view, this architecture is more suitable for segmentation and detection tasks compare to classification tasks.

4.3.5 Comparison on different models

The comparison of recently proposed models based on error, network parameters, and maximum number of connections are given in Table II

TABLE II. THE TOP-5% ERRORS WITH COMPUTATIONAL PARAMETERS AND MACS FOR DIFFERENT DEEP CNN MODELS.

Methods	**LeNet-5**	**AlexNet**	**OverFeat (fast)**	**VGG-16**	**GoogLeNet**	**ResNet-50(v1)**
Top-5 errors	n/a	16.4	14.2	7.4	6.7	5.3
Input size	28x28	227x227	231x231	224x224	224x224	224x224
Number of Conv Layers	2	5	5	16	21	50
Filter Size	5	3,5,11	3,7	3	1,3,5,7	1,3,7
Number of Feature Maps	1,6	3-256	3-1024	3-512	3-1024	3-1024
Stride	1	1,4	1,4	1	1,2	1,2
Number of Weights	26k	2.3M	16M	14.7M	6.0M	23.5M
Number of MACs	1.9M	666M	2.67G	15.3G	1.43G	3.86G
Number of FC layers	2	3	3	3	1	1
Number of Weights	406k	58.6M	130M	124M	1M	1M
Number of MACs	405k	58.6M	130M	124M	1M	1M
Total Weights	431k	61M	146M	138M	7M	25.5M
Total MACs	2.3M	724M	2.8G	15.5G	1.43G	3.9G

4.3.6 Other models

There are many other network architectures such as fast region-based CNN and Exception which are popular in the computer vision community. In 2015 a new model was proposed using recurrent convolution layers named Recurrent Convolution Neural Network or RCNN. The improved version of this network is a combination of the two most popular architectures in the Inception network and Recurrent Convolutional Network, Inception Convolutional Recurrent Neural Networks(IRCNN).IRCNN provided better accuracy compared RCNN and inception network with almost identical network parameters. Visual Phase Guided CNN (ViP

CNN) is proposed with phase guided message passing structure (PMPS) to build connections between relational components, which show better speed up and recognition accuracy . Look up based CNNis a fast, compact, and accurate model enabling efficient inference. In 2016 the architecture known as fully convolutional network (FCN) was proposed for segmentation tasks where it is now commonly used. Other recently proposed CNN models include deep network with stochastic depth, deeply-supervised networks and ladder network The Question is, do deep nets really need to be deeper? Some papers have been published base on the justification for deeper networks and concluded that "Deeper is better" . Now the question is which one is better width versus depth? On the one hand, there is controversy whether deep or wide networks are better some studies can be seen in the following papers .As DL approaches are data driven techniques which require a lot of labeled samples for training for the supervised approach. Recently some frameworks have been developed for making efficient databases from labeled and un-labeled datasets . Hyper parameter optimization allows for variable levels of performance, which is helpful for creating models to pair with designing hardware for deep learning.

4.3.7 Applications of CNNs

Most of the techniques that have been discussed above are evaluated on computer vision and image processing tasks., which are applied for different modalities of computer vision and image processing.

4.3.7.1 CNNs for solving Graph problem

Learning graph data structures is a common problem with various different applications in data mining and machine learning tasks. DL techniques have made a bridge in between the machine learning and data mining groups. An efficient CNN for arbitrary graph processing was proposed in 2016

4.3.7.2 Image processing and computer vision

Most of the models, we have discussed above are applied on different application domains including image classification, detection, segmentation, localization, captioning, video classification and many more. There is a good survey on deep learning approaches for image processing and computer vision related tasks. Single image super-resolution using CNN methods. Image de-noising using block-matching CNN. Photo aesthetic assessment using A-Lamp: Adaptive Layout-Aware Multi-Patch Deep CNN. DCNN for hyper spectral imaging for segmentation using Markov Random Field (MRF). Image registration using CNN. The Hierarchical Deep CNN for Fast Artistic Style Transfer .Background segmentation using

DCNN. Handwritten character recognition using DCNN approaches .Optical image classification using deep learning approaches. Object recognition using cellular simultaneous recurrent networks and convolutional neural network

4.3.7.3 Speech processing

CNN methods are also applied for speech processing: speech enhancement using multimodal deep CNN, and audio tagging using Convolutional Gated Recurrent Network (CGRN)

4.3.7.4 CNN for medical imaging

A good survey on DL for medical imaging for classification, detection, and segmentation tasks. There are some papers published after this survey. MD Net, which was developed for medical diagnosis with images and corresponding text description .Cardiac Segmentation using short-Axis MRI. Segmentation of optic disc and retina vasculature using CNN. Brain tumor segmentation using random forests

4.4 ADVANCED TRAINING TECHNIQUES

What is missing in the previous section is the advanced training techniques or components which need to be considered carefully for efficient training of DL approaches. There are different advanced techniques to apply to train a deep learning model better. The techniques including input pre-processing, better initialization method, batch normalization, alternative convolutional approaches, advanced activation functions, alternative pooling techniques, network regularization approaches, and better optimization method for training. The following sections are discussed on individual advanced training techniques individually.

4.4.1 Preparing dataset

Presently different approaches have been applied before feeding the data to the network. The different operations to prepare a dataset are as follows; sample rescaling, mean subtraction, random cropping, flipping data with respective to the horizon or vertical axis, color jittering, PCA/ZCA whitening and many more.

4.4.2 Network initialization

The initialization of deep networks has a big impact on the overall recognition accuracy . Previously, most of the networks have been initialized with random weights. For complex tasks with high dimensionality data training a DNN becomes difficult because weights should not be symmetrical due to the back-propagation process. Therefore, effective initialization techniques are important for training this type of DNN. However, there are many efficient techniques that

have been proposed during last few years. In 1998, LeCun and Y. Bengio in 2010 proposed a simple but effective approach. In this method, the weights are scaled by the inverse of the square root of number of input neurons of the layer, which can be stated $1/\sqrt{N_l}$, where N_l is the number of input neurons of l^{th} layer. The deep network initialization approach of Xavier has been proposed based on the symmetric activation function with respect to the hypothesis of linearity. This approach is known as "Xavier" initialization approach. Recently in 2016, Dmytro M. et al. proposed Layer-sequential unit-invariance (LSUV), which is a data-driven initialization approach and provides good recognition accuracy on several benchmark datasets including Image Net . One of the popular initialization approaches has proposed by Kiming He in 2015 . The distribution of the weights of l^{th} layer will be normal distribution with mean zero and variance $\frac{2}{n_l}$ which can be expressed as follows.

$$w_l \sim \mathcal{N}\left(0, \frac{2}{n_l}\right) \qquad \textbf{(47)}$$

4.4.3 Batch Normalization

Batch normalization helps accelerate DL processes by reducing internal covariance by shifting input samples. What that means is the inputs are linearly transformed to have zero mean and unit variance. For whitened inputs, the network converges faster and shows better regularization during training, which has an impact on the overall accuracy. Since the data whitening is performed outside of the network, there is no impact of whitening during training of the model. In the case of deep recurrent neural networks, the inputs of the nth layer are the combination of n-1 th layer, which is not raw feature inputs. As the training progresses the effect of normalization or whitening reduce respectively, which causes the vanishing gradient problem. This can slow down entire training process and cause saturation. To better the training process during training batch normalization is then applied to the internal layers of the deep neural network. This approach ensures faster convergence in theory and during experiment on benchmarks. In batch normalization, the features of a layer are independently normalized with mean zero and variance one. The algorithm of Batch normalization is given in Algorithm IV.

Algorithm IV: Batch Normalization (BN)

Inputs: Values of x over a mini-batch: $\mathfrak{B} = \{x_{1,2,3.....,m}\}$

Outputs: $\{y_i = BN_{\gamma,\beta}(x_i)\}$

$\mu_{\mathfrak{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$ // mini-batch mean

$\sigma_{\mathfrak{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathfrak{B}})^2$ // mini-batch variance

$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathfrak{B}}}{\sqrt{\sigma_{\mathfrak{B}}^2 + \epsilon}}$ // normalize

$y_i = \gamma\hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i)$ // Scaling and shifting

The parameters γ and β are used for the scale and shift factor for the normalization values, so normalization does not only depend on layer values. If you use normalization techniques, the following criterions are recommended to consider during implementation:

- Increase learning rate
- Dropout (batch normalization does the same job)
- L_2 Weight regularization
- accelerating the learning rate decay
- Remove Local Response Normalization (LRN) (if you used it)
- Shuffle training sample more thoroughly
- Use less distortion of images in the training set

4.4.4 Alternative Convolutional methods

Alternative and computationally efficient convolutional techniques that reduces the cost of multiplications by factor of 2.5 have been proposed

4.4.5 Activation function

The traditional Sigmoid and Tanh activation functions have been used for implementing neural network approaches in the past few decades. The graphical and mathematical representation is shown in Fig.4.30.

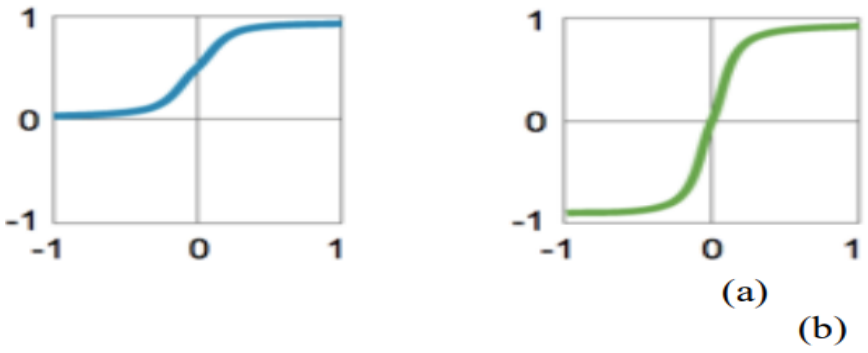


Fig.4.30. Activation function: (a)sigmoid function and (b)Hyperbolic transient

Sigmoid:

$$y = \frac{1}{1+e^{x}} \qquad \textbf{(48)}$$

TanH:

$$y = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \qquad \textbf{(49)}$$

The popular activation function called Rectified Linear Unit (ReLU) proposed in 2010 solves the vanishing gradient problem for training deep learning approaches. The basic concept is simple keep all the values above zero and sets all negative values to zero that is shown in Fig. The ReLU activation was first used in Alex Net, which was a breakthrough deep CNN proposed in 2012 by Hinton

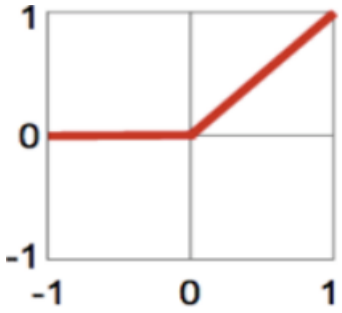


Fig.4.31. Pictorial representation of Rectified Linear Unit (ReLU)

Mathematically we can express ReLU as follows:

$$y = \max(0, x) \qquad \textbf{(50)}$$

As the activation function plays a crucial role in learning the weights for deep architectures. Many researchers focus here because there is much that can be done in this area. Meanwhile, there are several improved versions of ReLU that have been proposed, which provide even better accuracy compared to the ReLU activation function. An efficient improved version of ReLU activation function is called the parametric ReLU (PReLU) proposed by Kaiming He et al. in 2015. The Fig. shows the pictorial representation of Leaky ReLU and ELU activation functions. This technique can automatically learn the parameters adaptively and improve the accuracy at negligible extra computing cost

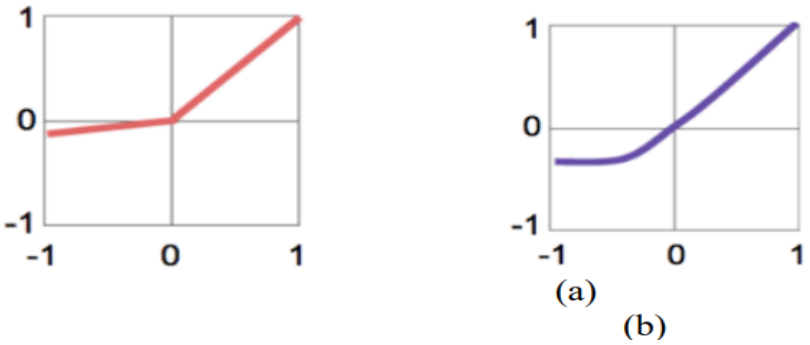


Fig.4.32. Diagram for (a) Leaky ReLU (b) Exponential Linear Unit (ELU)

Leaky ReLU:

$$y = \max(ax, x) \qquad (51)$$

Here a is a constant, the value is 0.1.

ELU:

$$y = \begin{cases} x, & x \geq 0 \\ a(e^x - 1), & x < 0 \end{cases} \qquad (52)$$

The recent proposal of the Exponential Linear Unit activation function, which allowed for a faster and more accurate version of the DCNN structure. Furthermore, tuning the negative part of activation function creates the leaky ReLU with Multiple Exponent Linear Unit (MELU) that are proposed recently. S shape Rectified Linear Activation units are proposed in 2015. A survey on modern activation functions was conducted in 2015

4.4.6 Sub-sampling layer or pooling layer

At present, two different techniques have been used for implementation of deep networks in the sub-sampling or pooling layer: average and max-pooling. The concept of average pooling layer was used for the first time in Le Net and Alex Net used Max-pooling layers instead in 2012. The conceptual diagram for max pooling and average pooling operation are shown in the Fig. 4.35. The concept of special pyramid pooling has been proposed by He et al. in 2014 which is shown in Fig.4.34

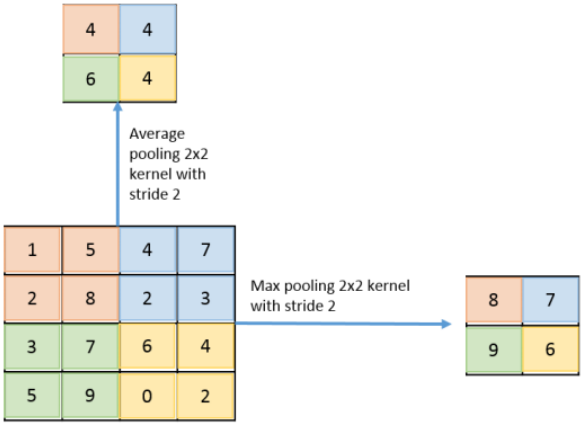


Fig.4.33. Average and max pooling operations.

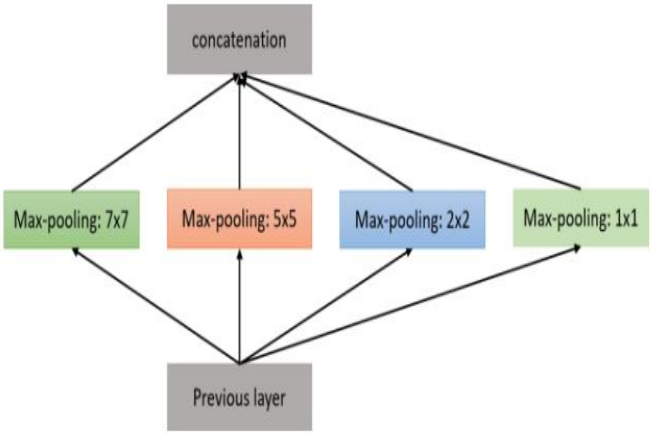


Fig.4.34. Spatial pyramid pooling.

The multi-scale pyramid pooling was proposed in 2015. In 2015, Benjamin G. proposed a new architecture with Fractional max pooling, which provides state-of-the-art classification accuracy for CIFAR-10 and CIFAR-100 datasets. This structure generalizes the network by considering two important properties for sub-sampling layer or pooling layer. First, the non-overlapped max-pooling layer limits the generalize of the deep structure of the network, this paper proposed a network with 3x3 overlapped max-pooling with 2- stride instead of 2x2 as sub-sampling layer . Another paper which has conducted research on different types of pooling approaches including mixed, gated, and tree as generalization of pooling functions.

4.4.7 Regularization approaches for DL

There are different regularization approaches that have been proposed in the past few years for deep CNN. The simplest but efficient approach called “dropout” was proposed by Hinton in 2012. In Dropout a randomly selected subset of activations are set to zero within a layer. The dropout concept is shown in Fig.4.35.

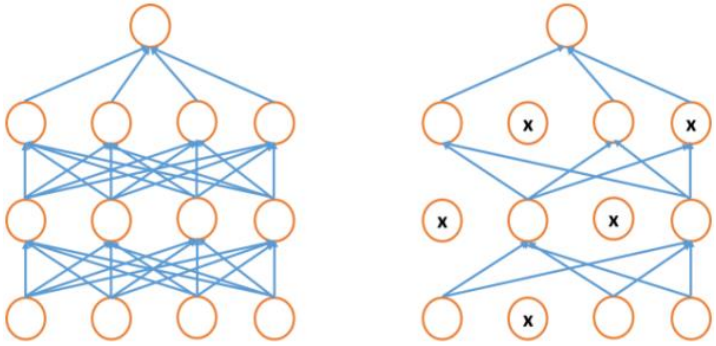


Fig.4.35. Pictorial representation of the concept Dropout

Another regularization approach is called Drop Connect, in this case instead of dropping the activation, the subset of weights within the network layers are set to zero. As a result, each layer receives the randomly selected subset of units from the immediate previous layer . Some other regularization approaches are proposed as well

4.4.8 Optimization methods for DL

There are different optimization methods such as SGD, Adagrad, AdaDelta, RMSprop, and Adam.Some activation functions have been improved upon such as in the case of SGD where it was been proposed with an added variable momentum, which improved training and testing accuracy. In the case of Adagrad, the main contribution was to calculate adaptive learning rate during training. For this method the summation of the magnitude of the gradient is considered to calculate the adaptive learning rate. In the case with a large number of epochs, the summation of magnitude of the gradient becomes large. The result of this is the learning rate decreases radically, which causes the gradient to approach zero quickly. The main drawback to this approach is that it causes problems during training. Later, RMSprop was proposed considering only the magnitude of the gradient of the immediate previous iteration, which prevents the problems with Adagrad and provides better performance in some cases. The Adam optimization approach is proposed based on the momentum and the magnitude of the gradient for calculating adaptive learning rate similar RMSprop. Adam has improved overall accuracy and helps for efficient training with better convergence of deep learning algorithms . The improved version of the Adam optimization approach has been proposed recently, which is called EVE. EVE provides even better performance with fast and accurate convergence.

5 Experimental results

In this section, detailed information about the experiment tall setup and the obtained results is given. The experimental setup includes the info about training the model and the software platform used in the present work.

5.1 Experimental setup

To evaluate the performance of the proposed pre-trained models a fixed partitioning scheme has been chosen. The datasets are partitioned into training and test sets using a ratio of 80:20. It means formerly, 80% of the dataset is employed for training and the rest 20% is employed for testing the transferred models. Additionally, in “Discussion” section when the comparison is made with the current state-of-the-art works we have followed the same settings existing in the literature employing both the fixed partition and the cross-validation schemes. The pre-trained models, the learning rate surges the network training time and increasing the rate leads training to get stuck at a suboptimal result.

Dataset link ((https://www.kaggle.com/sartajbhuvaji/brain-tumor-classification-mri)

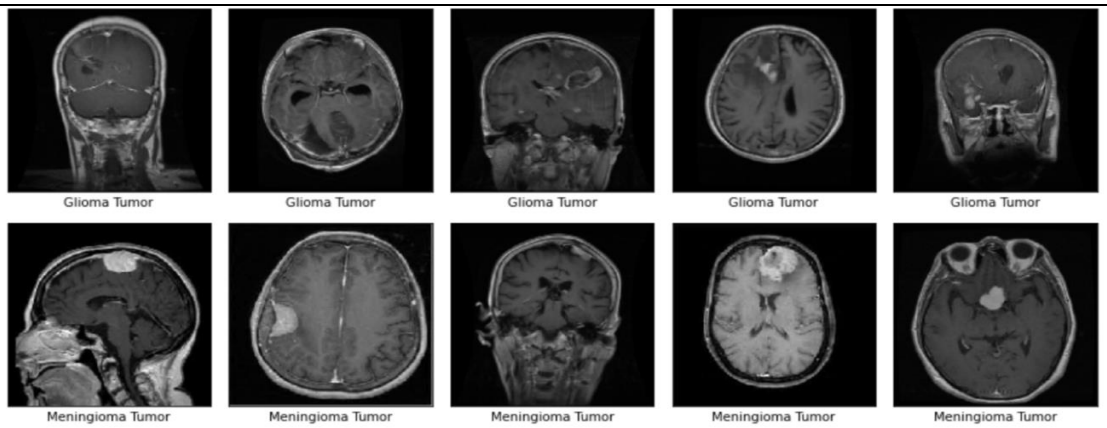


Fig1.

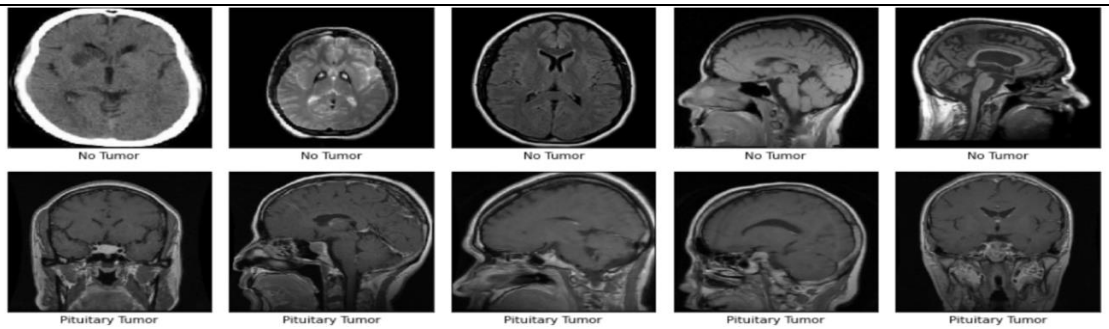


Fig.2

Fig 1 & fig 2 : Sample normal and abnormal brains from the Navoneel Chakrabarty ,Swati Kanchan, and clinical dataset

5.2 Code

In this project the Python language was used and I will show some of this work

```
        img = cv2.imread(path)
        img = cv2.resize(img,(224,224))
        X_test.append(img)
        y_test.append('Glioma')

for i in tqdm(os.listdir(test_meningioma)):
        path = os.path.join(test_meningioma,i)
        img = cv2.imread(path)
        img = cv2.resize(img,(224,224))
        X_test.append(img)
        y_test.append('Meningioma')


for i in tqdm(os.listdir(test_no_tumour)):
        path = os.path.join(test_no_tumour,i)
        img = cv2.imread(path)
        img = cv2.resize(img,(224,224))
        X_test.append(img)
        y_test.append('No_Tumour')

for i in tqdm(os.listdir(test_pituitary)):
        path = os.path.join(test_pituitary,i)
        img = cv2.imread(path)
        img = cv2.resize(img,(224,224))
        X_test.append(img)
        y_test.append('Pituitary')
```

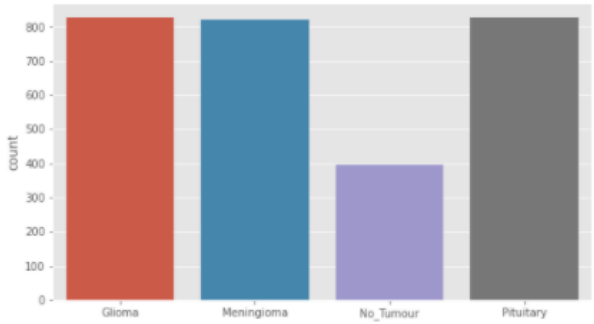
```
100%|██████████| 100/100 [00:00<00:00, 361.21it/s]
100%|██████████| 115/115 [00:00<00:00, 479.14it/s]
100%|██████████| 105/105 [00:00<00:00, 867.32it/s]
100%|██████████| 74/74 [00:00<00:00, 163.99it/s]
```

```
[ ]  # Converting to array form
```

```
[ ]  # train Reshaping
     X_train = np.array(X_train)
     y_train = np.array(y_train)
     # X_train.shape,y_train.shape
     print("Image shape:", X_train.shape[1:])
     print("Number of train samples:", X_train.shape[0])
```

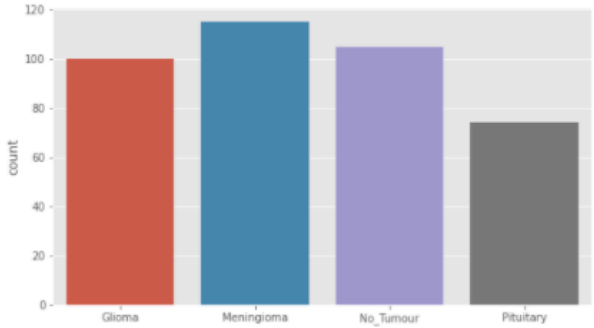
```
# Visualizing y_train and y_test count
import seaborn as sns
plt.style.use("ggplot")
plt.figure(figsize=(9,5))
sns.countplot(y_train)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12,
  FutureWarning
```




```
import seaborn as sns
plt.style.use("ggplot")
plt.figure(figsize=(9,5))
sns.countplot(y_test)
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12,
  FutureWarning
```



```
[ ]  1 num_classes = 4
     2
     3 model = tf.keras.Sequential([
     4   #data_augmentation,
     5   normalization_layer,
     6   #tf.keras.layers.Conv2D(32,3,activation='relu'),
     7   conv_layer_32,
     8   layers.MaxPooling2D(pool_size=(2,2)),
     9   conv_layer_32,
    10   layers.MaxPooling2D(pool_size=(2,2)),
    11   layers.Flatten(),
    12   layers.Dense(32, activation='relu'),
    13   layers.Dropout(0.25),
    14   layers.Dense(num_classes,activation='softmax')
    15 ])
```

model

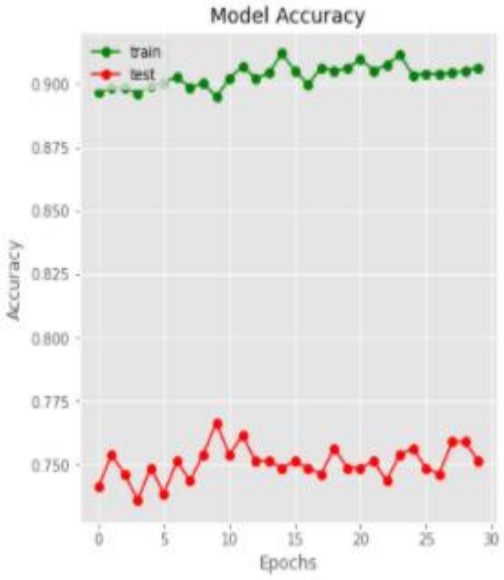
```
[ ]  1 model.summary()
```

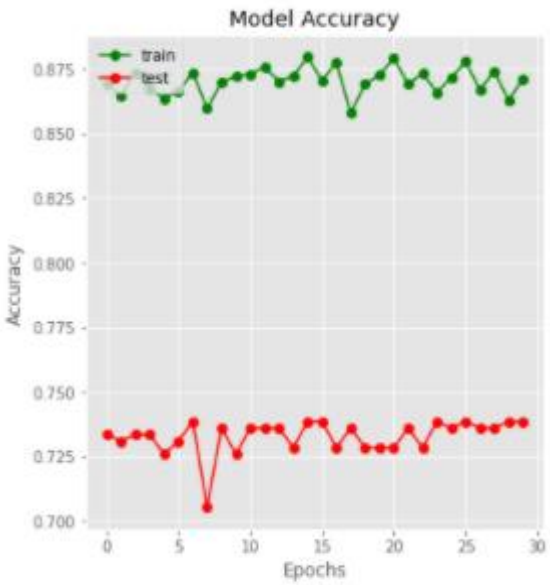
```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
rescaling (Rescaling)        (None, 224, 224, 3)       0
_________________________________________________________________
conv2d (Conv2D)              (None, 222, 222, 32)      896
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 111, 111, 32)      0
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 55, 55, 32)        0
_________________________________________________________________
flatten_2 (Flatten)          (None, 96800)             0
_________________________________________________________________
dense_4 (Dense)              (None, 32)                3097632
_________________________________________________________________
dropout_2 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_5 (Dense)              (None, 4)                 132
=================================================================
Total params: 3,098,660
Trainable params: 3,098,660
Non-trainable params: 0
```

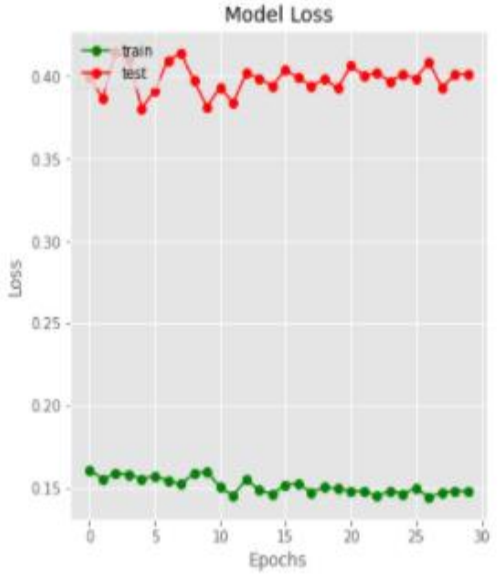
```
[ ]  1 effnet = applications.EfficientNetB7(weights = "imagenet",include_top=False,input_shape=(img_size,img_size,3))
     2
     3 for layer in effnet.layers:
     4     layer.trainable = False
     5
     6 model = models.Sequential([
     7     effnet,
     8     layers.GlobalAveragePooling2D(),
     9     layers.Dropout(0.3),
    10     layers.Dense(4, "softmax")
    11 ])
    12
    13 model.summary()
```

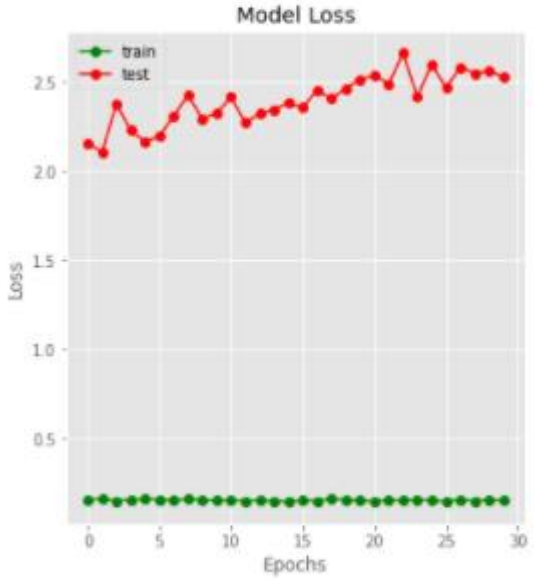
EfficientNetB7

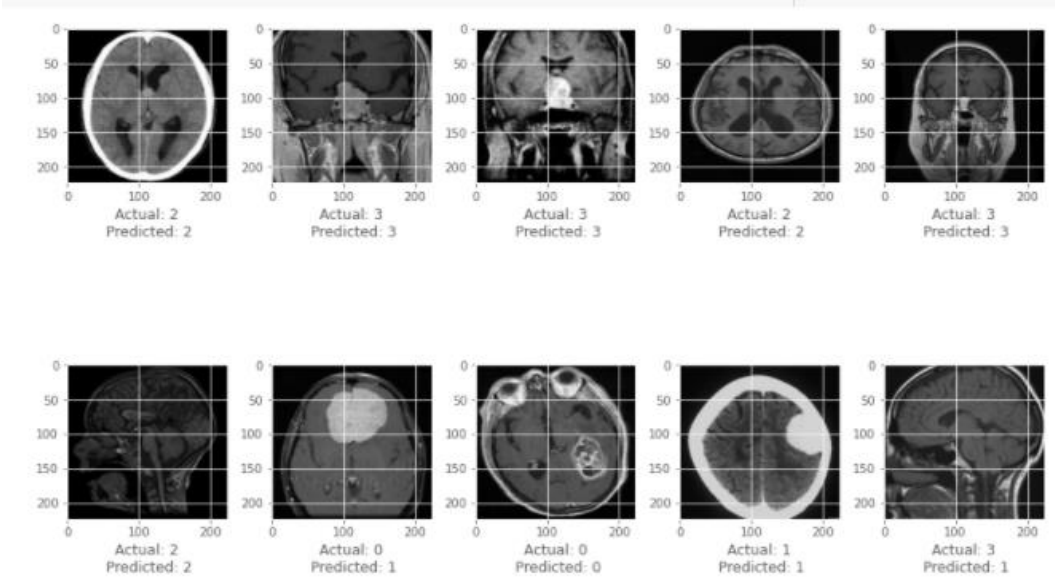
```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
efficientnetb7 (Functional)  (None, 7, 7, 2560)        64097687
_________________________________________________________________
global_average_pooling2d_14  (None, 2560)              0
_________________________________________________________________
dropout_14 (Dropout)         (None, 2560)              0
_________________________________________________________________
dense_14 (Dense)             (None, 4)                 10244
=================================================================
Total params: 64,107,931
Trainable params: 10,244
Non-trainable params: 64,097,687
_________________________________________________________________
```


Model Accuracy
train
test
Accuracy
Epochs


Model Accuracy
train
test
Accuracy
Epochs


Model Loss
train
test
Loss
Epochs


Model Loss
train
test
Loss
Epochs

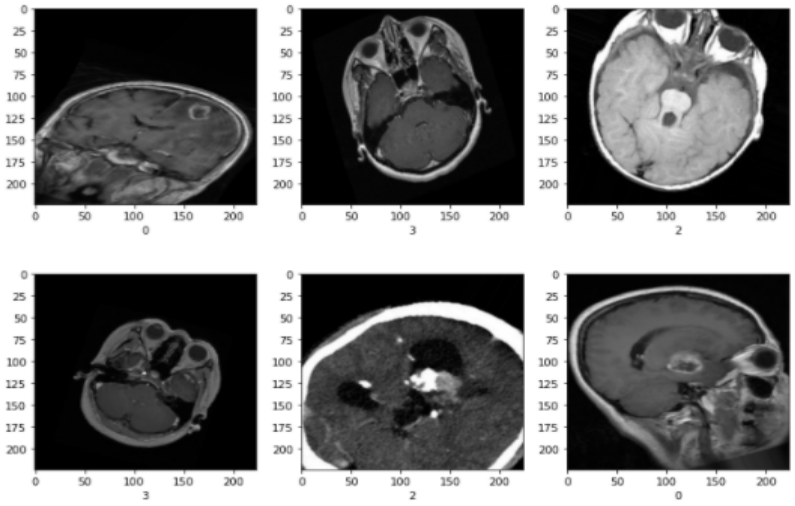

Actual: 2
Predicted: 2
Actual: 3
Predicted: 3
Actual: 3
Predicted: 3
Actual: 2
Predicted: 2
Actual: 3
Predicted: 3
Actual: 2
Predicted: 2
Actual: 0
Predicted: 1
Actual: 0
Predicted: 0
Actual: 1
Predicted: 1
Actual: 3
Predicted: 1

```
[ ]  1 # Data augmentation
```

```
 1 base_dir = '/content/brain_tumor_classification/dataset/train'
 2 img_size=224
 3 datagen= ImageDataGenerator(      rotation_range=30,height_shift_range=0.2,
 4                                   zoom_range = 0.3,horizontal_flip=True
 5                             )
 6
 7 test_dir = '/content/brain_tumor_classification/dataset/test'
 8 train_gen = datagen.flow_from_directory(base_dir,target_size=(img_size,img_size),shuffle=True,class_mode ="categorical",
 9                                         batch_size=32)
10 val_gen = datagen.flow_from_directory(test_dir,target_size=(img_size,img_size),shuffle=True, class_mode ="categorical",
11                                         batch_size=32)
12
13
```

```
Found 2870 images belonging to 4 classes.
Found 394 images belonging to 4 classes.
```

```
[ ]  1 from tensorflow.keras.preprocessing import image
     2 sample_x,sample_y = next(train_gen)
     3 plt.figure(figsize=(12,9))
     4 for i in range(6):
     5     plt.subplot(2,3,i+1)
     6     sample = image.array_to_img(sample_x[i])
     7     plt.xlabel(np.argmax(sample_y[i]))
     8     plt.imshow(sample)
     9 plt.show()
```

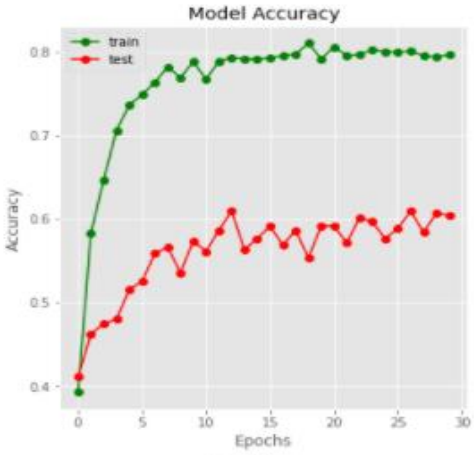


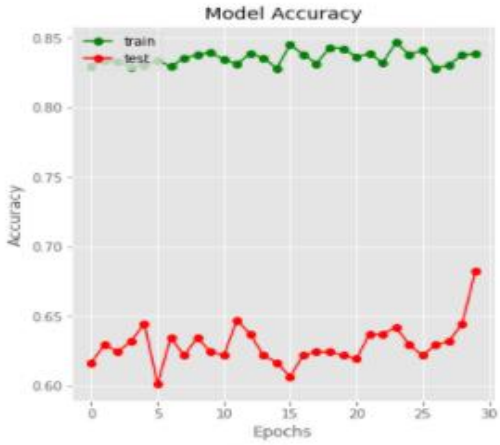

```
[ ]  1 effnet = applications.EfficientNetB6(weights = "imagenet",include_top=False,input_shape=(img_size,img_size,3))
     2
     3 for layer in effnet.layers:
     4     layer.trainable = False
     5
     6 model = models.Sequential([
     7     effnet,
     8     layers.GlobalAveragePooling2D(),
     9     layers.Dropout(0.3),
    10     layers.Dense(4, "softmax")
    11 ])
    12
    13 model.summary()
```

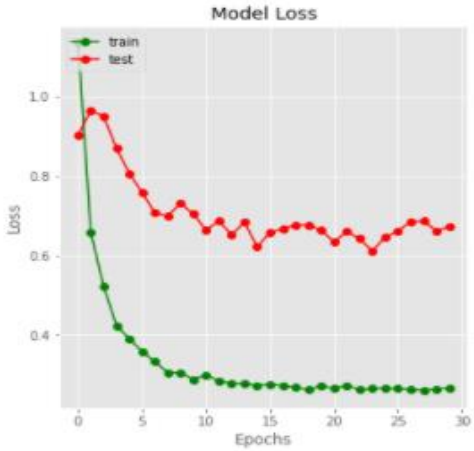
```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
efficientnetb6 (Functional)  (None, 7, 7, 2304)        40960143
_________________________________________________________________
global_average_pooling2d_11  (None, 2304)              0
_________________________________________________________________
dropout_11 (Dropout)         (None, 2304)              0
_________________________________________________________________
dense_11 (Dense)             (None, 4)                 9220
=================================================================
Total params: 40,969,363
Trainable params: 9,220
Non-trainable params: 40,960,143
_________________________________________________________________
```

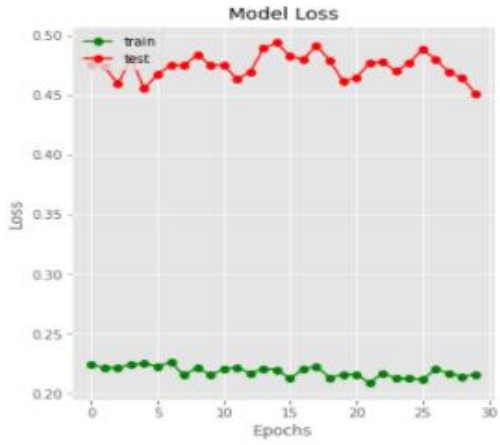
```
[ ]  1 vgg = applications.VGG16(weights = "imagenet",include_top=False,input_shape=(img_size,img_size,3))
     2 for layer in vgg.layers:
     3     layer.trainable = False
     4
     5 model = models.Sequential([
     6     vgg,
     7     layers.GlobalAveragePooling2D(),
     8     layers.Dropout(0.3),
     9     layers.Dense(4, "softmax")
    10 ])
    11
    12 model.summary()
```

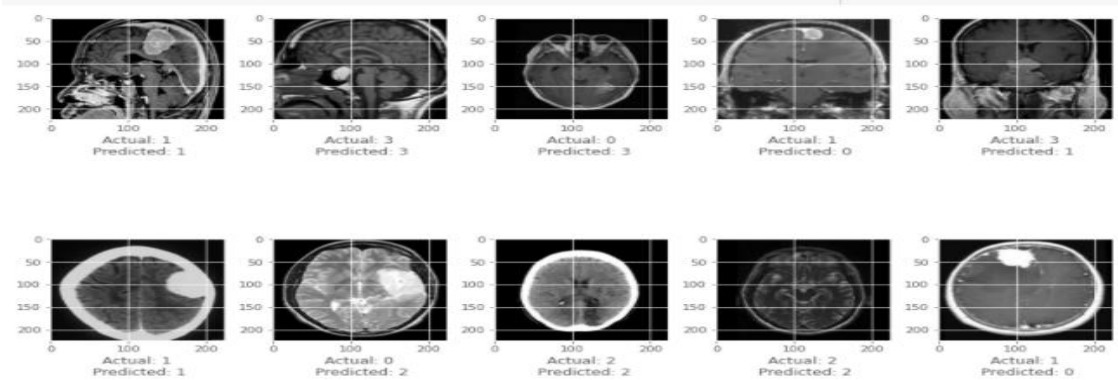
```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Functional)           (None, 7, 7, 512)         14714688
_________________________________________________________________
global_average_pooling2d_7 ( (None, 512)               0
_________________________________________________________________
dropout_7 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_7 (Dense)              (None, 4)                 2052
=================================================================
Total params: 14,716,740
Trainable params: 2,052
Non-trainable params: 14,714,688
```

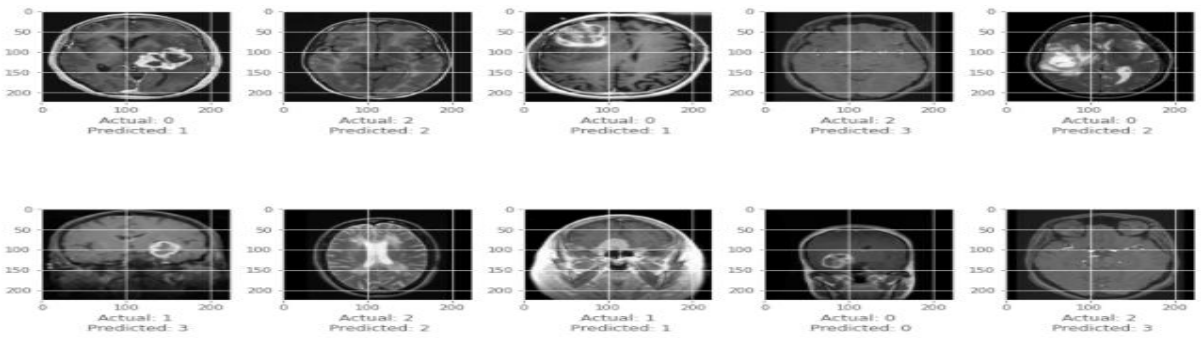












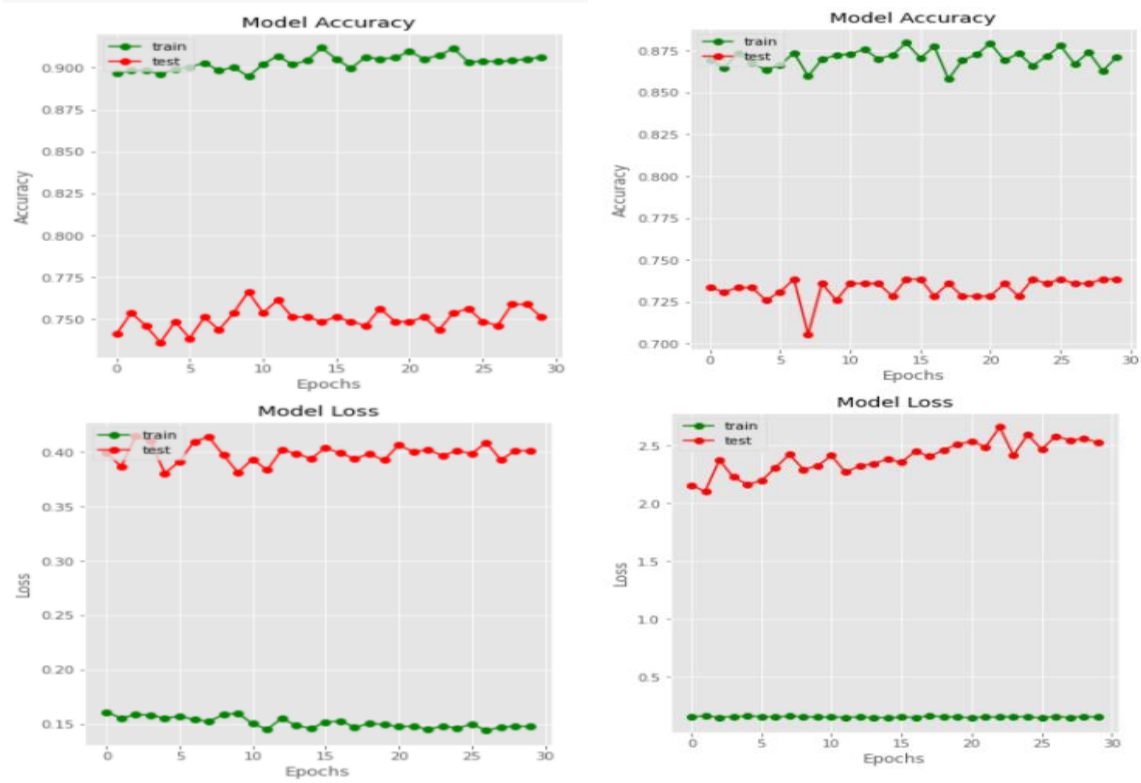
5.3 Results

The classification performance of the proposed transferred models is evaluated on different datasets, and the results are summarized in the form of tables and graphical figures. Table 2 illustrates the performance comparison for all the pre-trained models using all the chosen performance metric

Table 1. Brain tumor systems accuracy for dataset name without data augmentation

	Models	Training accuracy	Validation Accuracy
1	**EfficientNetB0**	0.8750	0.6701
2	**EfficientNetB1**	0.8860	0.7107
3	**EfficientNetB2**	0.8888	0.6701
4	**EfficientNetB6**	0.8796	0.6777
5	**EfficientNetB7**	0.9063	0.7513
6	**ResNet50V2**	0.7608	0.5228
7	**ResNet101V2**	0.7059	0.4391
8	**ResNet152V2**	0.6395	0.3934
9	**VGG16**	0.8878	0.6878
10	**VGG19**	0.8386	0.6904

	Models	precision	recall	f1-score
1	**EfficientNetB0**	0. 0.60 1. 0.61 2. 0.70 3. 0.76	0. 0.18 1. 0.81 2. 0.86 3. 0.85	0. 0.28 1. 0.69 2. 0.77 3. 0.80
2	**EfficientNetB1**	0. 0.81 1. 0.64 2. 0.71 3. 0.83	0. 0.21 1. 0.88 2. 0.90 3. 0.85	0. 0.33 1. 0.74 2. 0.80 3. 0.84
3	**EfficientNetB2**	0. 0.77 1. 0.57 2. 0.66 3. 0.90	0. 0.24 1. 0.81 2. 0.87 3. 0.76	0. 0.37 1. 0.67 2. 0.75 3. 0.82
4	**EfficientNetB6**	0. 0.72 1. 0.60 2. 0.70 3. 0.82	0. 0.23 1. 0.89 2. 0.82 3. 0.76	0. 0.35 1. 0.75 2. 0.75 3. 0.79
5	**EfficientNetB7**	0. 0.87 1. 0.64 2. 0.80 3. 0.88	0. 0.34 1. 0.91 2. 0.94 3. 0.78	0. 0.49 1. 0.75 2. 0.86 3. 0.83
6	**ResNet50V2**	0. 0.79 1. 0.59 2. 0.44 3. 0.70	0. 0.11 1. 0.47 2. 0.98 3. 0.51	0. 0.19 1. 0.52 2. 0.61 3. 0.59
7	**ResNet101V2**	0. 0.37 1. 0.68 2. 0.42 3. 0.47	0. 0.24 1. 0.11 2. 0. 77 3. 0.74	0. 0.29 1. 0.19 2. 0.55 3. 0.57
8	**ResNet152V2**	0. 0.64 1. 0.38 2. 0.38 3. 0.45	0. 0.07 1. 0.30 2. 0.87 3. 0.31	0. 0.13 1. 0.33 2. 0.52 3. 0.37
9	**VGG16**	0. 0.70 1. 0.61 2. 0.69 3. 0.86	0. 0.28 1. 0.82 2. 0.85 3. 0.81	0. 0.40 1. 0.70 2. 0.76 3. 0.83
10	**VGG19**	0. 0.78 1. 0.70 2. 0.60 3. 0.83	0. 0.28 1. 0.81 2. 0.88 3. 0.80	0. 0.41 1. 0.75 2. 0.71 3. 0.81



Whereas when using **data augmentation** to improve the results in a better way, it showed opposite results, so the results were low and the classification of tumors was poor due to the lack of familiarity with the model with tumors, and the results show the following.

Table 1. Brain tumor systems accuracy dataset name with data augmentation

	Models	Training accuracy	Validation Accuracy
1	**EfficientNetB0**	0.8330	0.6142
2	**EfficientNetB1**	0.8286	0.6371
3	**EfficientNetB2**	0.8514	0.6447
4	**EfficientNetB6**	0.8383	0.6827
5	**EfficientNetB7**	0.8402	0.6523
6	**ResNet50V2**	0.5613	0.4061
7	**ResNet101V2**	0.4735	0.3832
8	**ResNet152V2**	0.4427	0.3223
9	**VGG16**	0.7903	0.6041
10	**VGG19**	0.7786	0.6041

	Models	precision	recall	f1-score
1	EfficientNetB0	0. 0.58 1. 0.64 2. 0.62 3. 0.67	0. 0.22 1. 0.68 2. 0.83 3. 0.85	0. 0.32 1. 0.66 2. 0.71 3. 0.75
2	EfficientNetB1	0. 0.62 1. 0.59 2. 0.72 3. 0.68	0. 0.15 1. 0.75 2. 0.89 3. 0.88	0. 0.24 1. 0.66 2. 0.79 3. 0.76
3	EfficientNetB2	0. 0.67 1. 0.59 2. 0.69 3. 0.67	0. 0.26 1. 0.69 2. 0.81 3. 0.89	0. 0.37 1. 0.64 2. 0.74 3. 0.77
4	EfficientNetB6	0. 0.65 1. 0.68 2. 0.64 3. 0.61	0. 0.24 1. 0.65 2. 0.84 3. 0.89	0. 0.35 1. 0.66 2. 0.72 3. 0.73
5	EfficientNetB7	0. 0.60 1. 0.69 2. 0.71 3. 0.60	0. 0.38 1. 0.65 2. 0.85 3. 0.78	0. 0.47 1. 0.67 2. 0.77 3. 0.68
6	ResNet50V2	0. 1.00 1. 0.51 2. 0.50 3. 0.46	0. 0.06 1. 0.66 2. 0.55 3. 0.77	0. 0.11 1. 0.58 2. 0.52 3. 0.58
7	ResNet101V2	0. 0.67 1. 0.44 2. 0.51 3. 0.30	0. 0.02 1. 0.37 2. 0.37 3. 0.89	0. 0.04 1. 0.40 2. 0.43 3. 0.45
8	ResNet152V2	0. 0.00 1. 0.34 2. 0.30 3. 0.34	0. 0.00 1. 0.16 2. 0.70 3. 0.43	0. 0.00 1. 0.21 2. 0.42 3. 0.38
9	VGG16	0. 0.59 1. 0.55 2. 0.80 3. 0.72	0. 0.30 1. 0.73 2. 0.77 3. 0.86	0. 0.40 1. 0.63 2. 0.79 3. 0.79
10	VGG19	0. 0.45 1. 0.63 2. 0.61 3. 0.70	0. 0.38 1. 0.51 2. 0.75 3. 0.81	0. 0.41 1. 0.57 2. 0.67 3. 0.75

5.4 Discussion

The subsection of discussion illustrates the comparison of the best performing transferred CNN models with other existing state-of-the-art works on brain tumor categorization and the challenging clinical dataset. the comparison has been made in terms of the performance metrics reported in the literature, i.e., the no. of features, classification accuracy, computation time, and data partitioning scheme.

5.5 Advantages of the proposed method

Advantages of the proposed method The advantages of using pre-trained DCNN models with transfer learning for classification are manifold: firstly, the classification mechanism is fully automated, secondly it eliminates the conventional steps of noise filtering, ROI delineation, feature extraction, and selection, thirdly no interand intra-observer biases are there and the predictions made by the pre-trained DCNN models are reproducible, and finally, ceiling level of accuracy is achieved in contrast to similar works reported using DCNN.