People's Democratic Republic of Algeria
الجمهورية الجزائرية الديمقراطية الشعبية
Ministry of Higher Education and Scientific Research
وزارة التعليم العالي و البحث العلمي
Kasdi Merbah University of Ouargla

Dissertation Submitted to the Department of Computer Science and Information Technology in Candidacy for the Degree of Master LMD in Industry Artificial intelligence and data science branch
Master degree thesis

Students :

Zakaria Babelhadj

Safa Khemgani

Classification of Brain Tumor using some approaches of Machine Learning

Supervisor: Dr. Bilel Khaldi
President: Dr. Leila Amrane
Examiner: Dr. Kamel Benkaddour

Aknowledgments

We would like to begin by expressing our utmost gratitude to Allah, the Almighty, for His blessings and guidance throughout the completion of our dissertation. We would also like to extend our sincere appreciation to the following individuals and organizations who played a vital role in the accomplishment of our master's degree thesis:

First and foremost, we would like to acknowledge and express our deepest gratitude to our thesis advisor, Dr. Khalid Bilel. His unwavering support, expert guidance, and invaluable insights have been instrumental in shaping this dissertation. We are truly grateful for his dedication and constructive feedback throughout our research journey.

We would also like to extend our thanks to the faculty members at Kasdi Merbah University. Their provision of a conducive academic environment and access to necessary resources have been pivotal in facilitating our research and ensuring its effectiveness.

Our heartfelt appreciation goes out to our families, particularly our parents and siblings, for their unconditional love, patience, and understanding. Their unwavering support and belief in us have served as a constant source of motivation throughout this endeavor.

Additionally, we would like to express our gratitude to our friends and colleagues who have provided valuable assistance, engaging discussions, and constant encouragement during the process of writing this dissertation. Their camaraderie and intellectual exchanges have been truly invaluable.

Furthermore, we would like to acknowledge and thank the participants of our research, whose generous contribution of time, knowledge, and experiences have made this study possible. Without their willingness to participate, our work would not have been able to come to fruition.

In conclusion, the successful completion of this master's degree dissertation would not have been possible without the support, guidance, and

contributions of all the individuals and organizations mentioned above. We are deeply indebted to every one of them

Abstract

Detecting brain tumors in their early stages is crucial. Therefore, prompt detection enables doctors to make timely and informed decisions. This dissertation focuses on developing and evaluating artificial intelligence techniques to identify the tumor and type of tumor using medical imaging and one of these medical imaging techniques to observe it is MRI.
This work aims to improve the accuracy and efficiency of brain tumor detection .

We have used artificial intelligence to detect the tumor and classify what type it is whether glioma, meningioma, pituitary or healthy brain, and from the artificial intelligence method we have used three machine learning techniques logistic regression, SVM, and decision tree for deep learning we have used CNN and Pre-trained model ResNet50.

The machine learning methods have been trained after extracting features from brain tumor MRI with the SURF method and created BoF using two clustering techniques: K-means clustering and GMM clustering. The SVM that has been trained by features which have created using GMM clustering had the best performance in machine learning.

The deep learning CNN and pre-trained have been trained on brain tumor MRI, the evaluation showed that the pre-trained had the best performance.

Overall, the pre-trained model seemed to give the best accuracy by 0.976, and machine learning SVM had the best performance when the features were generated with GMM with an accuracy of 0.866.

Keywords: MRI: Magnetic resonance imaging, SVM: Support vector

machines, CNN: Convolution Neural Networks, SURF: Speeded Up Robust Features, GMM: Gaussian mixture model.

Résumé

La détection des tumeurs cérébrales à leurs débuts est cruciale. Par conséquent, une détection rapide permet aux médecins de prendre des décisions opportunes et éclairées. Les objectifs de ce mémoire portent en premier lieu sur le développement et l'évaluation des techniques d'intelligence artificielle pour identifier la tumeur et le type de cette dernière à l'aide de l'imagerie médicale qui est l'une de ces techniques d'imagerie médicale pour l'observer est l'IRM.
Ce travail vise à améliorer la précision et l'efficacité de la détection des tumeurs cérébrales.

Nous avons utilisé l'intelligence artificielle pour détecter la tumeur et déterminer de quel type il s'agit: gliome, méningiome, hypophyse ou cerveau sain, et à partir de la méthode d'intelligence artificielle, nous avons utilisé trois techniques d'apprentissage automatique régression logistique, SVM et arbre de décision et pour l'apprentissage en profondeur nous avons utilisé CNN et le modèle pré-formé ResNet50.

Les méthodes d'apprentissage automatique ont été formées après avoir extrait des caractéristiques de l'IRM de tumeur cérébrale avec la méthode SURF, et on créé BoF en utilisant deux techniques de clustering, d'abord avec le clustering K-means et ensuite avec le clustering GMM. Le SVM qui a été formé par des fonctionnalités qu'on a créé en utilisant le clustering GMM avait les meilleures performances en apprentissage automatique.

Le CNN d'apprentissage en profondeur et les pré-formés ont été formés sur l'IRM des tumeurs cérébrales, l'évaluation a montré que les pré-formés avaient les meilleures performances.

Dans l'ensemble, le modèle pré-entraîné semblait donner la meilleure précision de 0,976, et l'apprentissage automatique SVM a eu les meilleures performances lorsque les fonctionnalités générées avec GMM avec une précision de 0,866.

Mots clés: IRM: Imagerie par résonance magnétique, SVM: Soutenir les machines vectorielles, CNN: Réseaux de Neurones Convolutifs, SURF: Surface Rapide de Caractéristiques Robustes, GMM: Modèle de Mélange Gaussien.

الملخص

يعد اكتشاف أورام المخ في مراحلها المبكرة أمرًا بالغ الأهمية. لذلك ، فإن الاكتشاف السريع يمكّن الأطباء من اتخاذ قرارات مستنيرة في الوقت المناسب. تركز هذه المذكرة على تطوير وتقييم تقنيات الذكاء الاصطناعي لتحديد الورم ونوع الورم باستخدام التصوير الطبي وأحد تقنيات التصوير الطبي هذه لمراقبته هو التصوير بالرنين المغناطيسي.

الهدف من هذا العمل هو تحسين دقة وكفاءة الكشف عن أورام المخ.

لقد استخدمنا الذكاء الاصطناعي للكشف عن الورم وتصنيف نوع الورم الدبقي أو الورم السحائي أو الغدة النخامية أو الدماغ الصحي ، ومن طريقة الذكاء الاصطناعي استخدمنا ثلاث تقنيات للتعلم الآلي: الانحدار اللوجستي ، SVM وشجرة القرار وللتعلم العميق استخدمت المدربين مسبقًا ResNet50 ونموذج CNN .

تم تدريب أساليب التعلم الآلي بعد استخراج الميزات من التصوير بالرنين المغناطيسي لورم الدماغ باستخدام طريقة SURF وإنشاء BoF باستخدام طريقتين للتجميع أولاً K-Means وثانيًا GMM . حقق SVM الذي تم تدريبه بواسطة

الميزات التي تم إنشاؤها باستخدام تجميع GMM أفضل أداء في التعلم الآلي.

تم تدريب شبكة CNN للتعلم العميق والمتدربين مسبقًا على التصوير بالرنين المغناطيسي لورم الدماغ ، وأظهر التقييم أن المتدربين مسبقًا كان لديهم أفضل أداء.

بشكل عام ، يبدو أن النموذج الذي تم تدريبه مسبقًا يعطي أفضل دقة تبلغ 0.976 ، و SVM التعلم الآلي كان له أفضل أداء عندما تم إنشاء الميزات باستخدام GMM بدقة 0.866 .

الكلمات المفتاحية: SVM : آلة الدعم الناقلة، CNN : شبكات التعرف العصبي المتسلسلة المتصالة، GMM :نموذج الخلط الجاوسي

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Diseases are in continuous development, and some of these mutations can make viruses deadlier and hard for the immunity to identify and fight disease for example the flu disease in a certain period the virus changes its genetic code and that change makes it hard to be identified the next time it enters the body. This is where we try to help our immune system with medicines, some of these drugs improve our immune system against the invaders [1]. Others assist the immune to destroy viruses and bacteria [2] and [3]. Furthermore, another kind of drug prevents the invaders from growing and deploying their code in the body. One of these deadly diseases is cancer.

1.1 Cancer

Cancer is a unique type of disease the reason behind that is the way it infects the body. Cancer is caused by unexpected changes in the DNA of a cell and it spreads from this cell to all cells around it. Four tests can indicate if a body has cancer or not, first, there is a physical exam, this approach is taken by a doctor when he sees an abnormality view on the body such as a change in skin color or an increase in the size of an organ and that may indicate skin cancer. There are also laboratory tests such as blood or urine tests, for example, if the blood test named blood count shows an irregular number of white blood cells this case doctor feels the presence of blood cancer. Another way to find out about cancer is by imaging where a doctor takes an image or images of your internal organs, and these images are taken using bone scan, computerized tomography (CT), ultrasound, X-ray and magnetic resonance

imaging (MRI), and others. Lastly, there is the biopsy, in this procedure, the doctor collects a sample from cells depending on the type of cancer and where it is located and then they put it under a microscope and look in this sample for abnormal cells from this sample [4].

1.2 Spotting and treating cancer

In the case of cancer, doctors will use one or more cancer treatments to cure the disease, the common approaches are surgery, chemotherapy, and radiation. In surgery, the surgeon removes the mass of cancerous cells, tumors, and the surrounding tissue. Chemotherapy corresponds to medicines that target cancer cells, these drugs are given all at once or one after another either by swallowing or through a blood vessel (IV). Radiation therapy is either X-ray, particles, or radioactive seeds it depends on the type or the treatment, if it is an external beam doctor will use an X-ray or particles on the area where the tumor is located outside the body, or an internal beam in this case he will using a radioactive seeds that will be inserted inside the body through a vein or swallowed in shape or pill or liquid.

Sometimes cancer can be hard to spot or diagnose from the tests, that is why doctors intend to repeat the test several times or make a different test each time. Some of these testing is particularly interesting which is the MRI. Magnetic resonance imaging or MRI is a technique used by doctors to generate detailed images of the internal organs, this type of medical imaging uses a magnetic field and computer-generated radio waves to put together clarified readable scans. The patient is placed inside a large machine and he must stand still until the end of the process as shown in Figure 1.1.

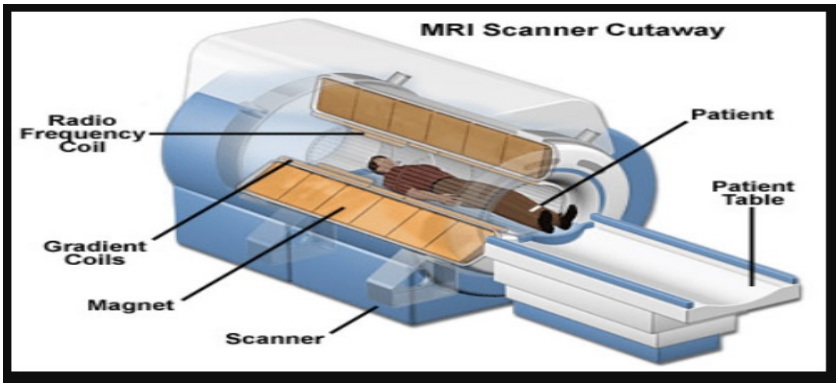


Figure 1.1: An Image showing how the patient is placed inside the MRI 1

After the patient gets inside the MRI, it deploys a strong magnetic field, this magnetic field forces the protons inside the patient to align with its magnetic field, after that it releases a radiofrequency current goes through the patient it disturbs the protons which makes them launch energy that MRI detects with its sensors. This energy that has been detected is what generates the image of the organ. The MRI is specially used on the brain because the brain is a soft tissue and any imaging testing like an X-ray would damage it unlike the MRI.

1.3 Brain tumor

Brain tumor is cancerous cells concentrated in part of the brain. There are a lot of types of brain tumors and each one has a set of symptoms depending on its size, location, and rate of growth, among them Meningioma, Pituitary, and Glioma.

1.3.1 Meningioma

Meningioma is a brain tumor that grows in the meninges of the brain as the name suggested, usually it grows very slowly and takes years for its symptoms to appear. The symptoms of a meningioma tumor are:

- Loss of smell.

- Seizures.
- Problems in vision like blurry or double vision.
- Headaches.
- Problems in-ears such as ringing or even loss of hearing.
- Memory loss.
- Speech difficulty.
- Fatigue in limbs.

Meningioma tumors manifest in specific regions of the brain. Figure 1.2 visually depicts the locations where meningioma tumors may be present.

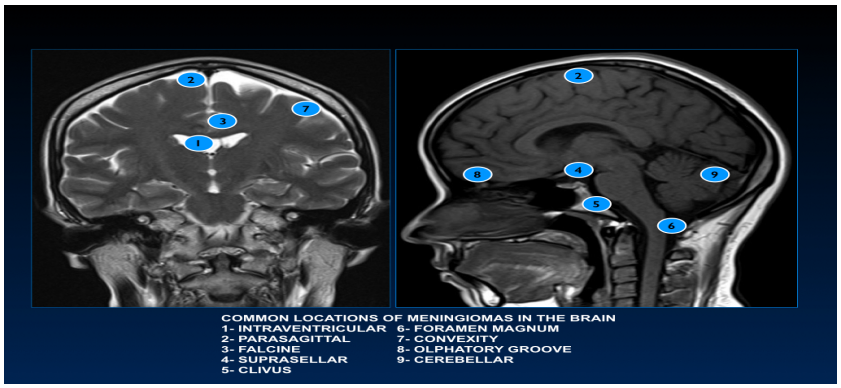


Figure 1.2: An Image showing The Location where Meningioma tumor can be found 2

1.3.2 Pituitary

Pituitary brain tumor is a tumor that appears in the pituitary gland of the brain, some pituitary tumors do not spread to other parts of the body. The symptoms of a pituitary tumor are:

- Headaches.
- Vision loss.

- Nausea and vomiting.
- Sexual dysfunction.
- Fatigue.
- Getting chilly.
- Reduce or stop the menstrual.
- Elevate the quantity of urine.
- Unexpected weight loss or gain.

Pituitary tumors are found in specific areas of the brain. Figure 1.3 illustrates the locations where pituitary tumors can be located.

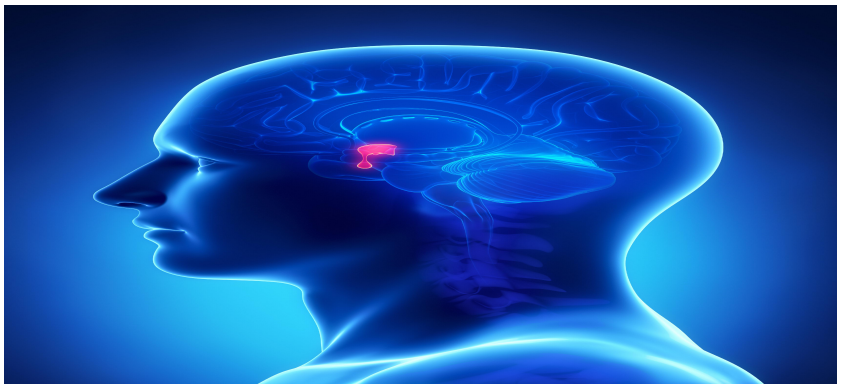


Figure 1.3: An Image showing The Location where Pituitary tumor can be found 3

1.3.3 Glioma

Glioma brain tumors can be seen in the brain or the spinal cord. Glioma grows in the glial cells these cells are non-neuronal do not make electrical impulses and they provide physical support to the neuron system.
There are four types of glioma tumors astrocytoma, ependymoma, glioblastoma, and Oligodendroglioma.

Astrocytoma

Astrocytoma tumors appear in cells called astrocytes these cells are located within the central nervous system. Astrocytoma tumors are localized in distinct areas within the brain. Figure 1.4 depicts the specific locations where astrocytoma tumors commonly occur. The symptoms depend on the location of the tumor. In the case Astrocytoma is located within the brain, the symptoms are:

- Seizures.
- Headaches.
- Nausea.

If the Astrocytoma is in the spinal cord:

- Fatigue.
- Disability in the area affected.

Figure 1.4 visually demonstrates the possible sites where Astrocytoma tumors may be found.

Figure 1.4: An Image showing The Location where Astrocytoma tumor can be found 4

Ependymoma

Ependymoma starts in the ependymal cells which are fluid passageways in the brain and spinal cord. These passageways are called cerebrospinal fluid.
Ependymoma most of the time occur in children's brain and it causes:

- Seizures.
- Headaches.

And in adults more common to be in the spinal core and it causes:

- Fatigue in the part affected by the tumor.

Ependymoma is highlighted in both Figure 1.5 and Figure 1.6 where it can be discovered in the brain and spine, respectively.

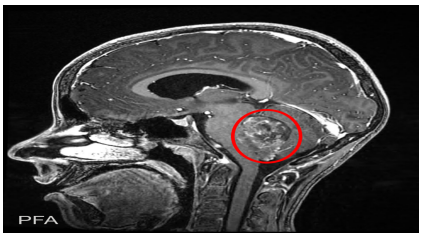


Figure 1.5: An Image showing The Location where Ependymoma tumor can be found in the brain 5

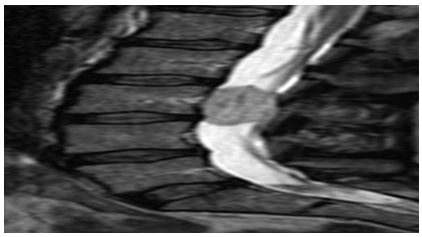


Figure 1.6: An Image showing The Location where Ependymoma tumor can be found in the spine 6

Oligodendroglioma

Oligodendroglioma occurs in oligodendrocytes cells these cells protect the nerve cells by producing matter on them. Oligodendroglioma affects more often adults. The locations of Oligodendroglioma tumors in both the brain and spine are displayed in Figure 1.7 and Figure 1.8.
The symptoms in case Oligodendroglioma affects the brain are:

- Headaches.
- Seizures.

In case it affects the spinal cord:

- weakness.
- disability of the part affected by the tumor.

Oligodendroglioma is indicated in Figure 1.7 and Figure 1.8, demonstrating its presence in the brain and spine, respectively.

Figure 1.7: An Image showing The Location where Oligodendroglioma tumor can be found in the brain 7

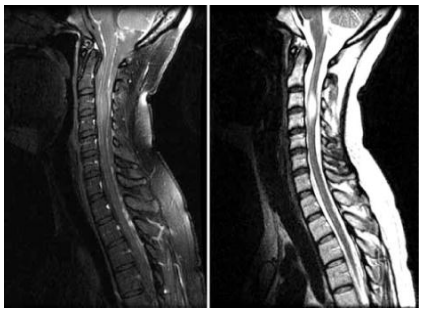


Figure 1.8: An Image showing The Location where Oligodendroglioma tumor can be found in the spine 8

Glioblastoma

Glioblastoma is an aggressive type of cancer that forms from astrocyte cells just like astrocytoma. Glioblastoma tends to occur in late adulthood, and its symptoms include:

- Headaches.
- Vomiting.
- Seizures.
- Nausea.

Figures 1.9 and 1.10 show the location of Glioblastoma in the brain and spine, accordingly.

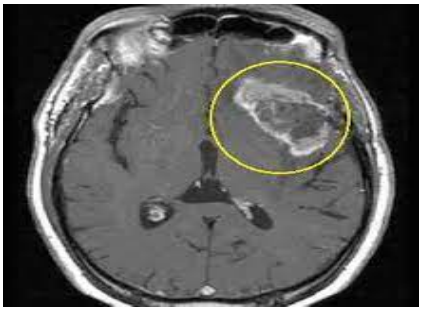


Figure 1.9: An Image showing The Location where Glioblastoma tumor can be found in the brain 9

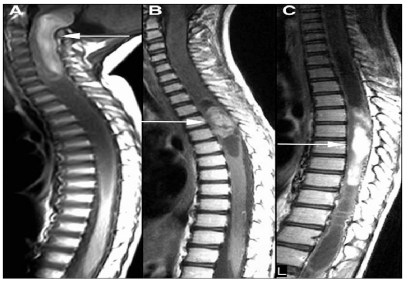


Figure 1.10: An Image showing The Location where Glioblastoma tumor can be found in the spine 10

Recognizing the challenges posed by misdiagnoses and undetected cases, scientists have been driven to find solutions that can significantly reduce or even eliminate these instances. With a strong motivation to enhance the accuracy and effectiveness of diagnostic processes, researchers have turned their attention to improving the hardware and software of medical machines. By leveraging the power of artificial intelligence and integrating it into these systems, they aim to revolutionize the field of medical diagnostics, ultimately leading to more reliable and precise detection of diseases. The prospect of saving lives and improving patient outcomes serves as a powerful driving force behind their efforts to develop innovative solutions in the fight against misdiagnoses and undetected cases.

1.4 Artificial intelligence

Artificial intelligence (AI) is the development of computer systems that are capable of doing tasks that usually necessitate human cognition. This multidisciplinary field integrates computer science, mathematics, cognitive science, and other fields to create intelligent robots that can perceive, reason, learn, and interact with their surroundings. This means that its systems strive to duplicate or simulate human intelligence, allowing machines to understand and process natural language, recognize images, make decisions, solve problems, and even exhibit creativity. These computers can quickly

and accurately do complex tasks thanks to their ability to analyze enormous volumes of data, detect patterns, and extract useful data. The use of AI is widespread and is constantly growing in various areas. AI is revolutionizing every aspect of our lives, from entertainment and transportation to healthcare and banking. It has the ability to better decision-making, automate repetitive operations, increase productivity, and open up fresh prospects for creativity. The two main categories of AI are general AI and narrow AI. Narrow AI is created to carry out particular tasks inside a constrained domain. like autonomous systems, recommendation algorithms, and speech recognition systems. On the other hand, general AI refers to highly autonomous machines that have human-level intelligence and are capable of performing a variety of activities. Overall, artificial intelligence is a potent instrument that can enhance human abilities, resolve challenging issues, and influence the future of many industries, fostering progress and opening up new opportunities. This is due to the multiplicity of the strength of his different techniques and approaches, including natural language processing, computer vision, robotics, expert systems, and machine learning which has played a crucial role in advancing AI.

Machine learning is a sub-field of artificial intelligence (AI) that focuses on the development of algorithms and models that enable computers to learn from data and make predictions or decisions without being explicitly programmed. It is a data-driven approach that allows machines to automatically analyze and interpret patterns in data and learn the relationships, and dependencies, and then use those patterns to make informed predictions or take actions. Machine learning can be classified as Supervised, Unsupervised, Semi-supervised, and Reinforcement Learning, among others. In supervised learning, input data and intended outputs are both provided, and the system is trained on labeled instances. The system learns how input and output are mapped, allowing it to anticipate or categorize previously unexplored data. While unsupervised learning uses unlabeled data to train algorithms with the intention of finding patterns or structures in the data. Reinforcement learning involves training an agent to interact with an environment and learn optimal actions to maximize a reward signal. The agent learns through trial and error, receiving feedback on its actions and adjusting its behavior accordingly, We find it in gaming and robots and others. Machine learning encompasses various tasks and algorithms that enable computers to learn from data and make predictions or decisions. There are classification and regression in Supervised Learning, and clustering and Dimensionality

Reduction tasks in unsupervised learning, and pre-trained Models in reinforcement learning systems.
With the multiplicity of fields and the expansion of machine learning, in order to solve problems and tasks, there are many machine learning algorithms. We mention some of them support vector machines, random forests, decision trees, and principal Component Analysis PCA .
And the matter did not stop at machine learning and its algorithms only, but in recent years and thanks to the development of artificial neural networks, deep learning, a sub-field of machine learning based on ANNs, has emerged as a dominant approach. Deep learning models have achieved remarkable success in various domains.
The integration of AI and machine learning with other technologies, such as natural language processing, computer vision, and reinforcement learning, has further expanded the capabilities of intelligent systems. AI-powered applications are now pervasive in our daily lives, from virtual assistants and autonomous vehicles to medical diagnosis and personalized recommendations.

With regard to medical diagnosis, machine learning plays an important role in it, as it provides valuable insights and provides useful aids and support for healthcare professionals and experts. The variety of machine learning algorithms and their ability to analyze large amounts of patient data, discover patterns, and make predictions helps diagnose various medical conditions. One important use of ML is in image recognition, where ML models excel at deciphering medical images like X-rays, MRIs, and CT scans. These models help radiologists find diseases like cancer or organ abnormalities by teaching them to recognize patterns and anomalies. They also utilize patient data and compare it with extensive databases to provide accurate diagnoses for conditions such as diabetes and cancer. It also provides customized risk evaluations and helps forecast the possibility of contracting diseases. ML supports treatment planning by identifying patterns, abnormalities, and medication errors by examining electronic health records. Among its applications are decision support systems as well as prognostic modeling that allows predicting patient outcomes with the probability of disease progression and assisting in treatment planning and long-term treatment.
So it can be said that machine learning enhances the accuracy, efficiency, and quality of medical diagnosis through its various applications such as medical image processing, electronic victim records, patient detection, risk prediction, decision support, predictive modeling, and other applications. With its

potential to improve patient outcomes, machine learning continues to be a vital tool for healthcare professionals, increasing their expertise and enabling them to make informed decisions for the benefit of patients worldwide.

Through this study, our primary objective is to deal with the challenges associated with brain tumor detection using machine learning and deep learning techniques. We will focus on analyzing brain tumor MRI images and develop machine learning and deep learning models that are efficient and accurate in detecting brain tumors. And to achieve our objectives, we address the following outline:

1. **Introduction**
    - What are cancer and brain tumor and their solutions.
    - What is artificial intelligence and its type.
    - How artificial intelligence can be integrated with medical images.
2. **Related work**
    - Review some existing works.
    - Explore their different approaches and methods.
    - Propose our method.
3. **Definitions**
    - Define some of image processing, machine learning, and deep learning methods.
    - State the reason for choosing them.
4. **Methods**
    - Explain the flow work with methods-based machine learning and how they have been used.
    - Describe the sequence of operations with methods based on deep learning.
5. **Experiment and results**
    - Describe the dataset and the materials that have been used and define some metrics.

- Explain the experiments of both machine learning and deep learning and the parameters that have been used.
- Comparison between deep learning and machine learning.

6. **Conclusion**

Chapter 2

Related works

This chapter represents a thorough analysis of the subject of research on brain tumor detection. This review seeks to summarize the key conclusions, identify research gaps, and discuss how this study might advance the area.

In [5] a hybrid CNN architecture was proposed on MRI scans of brain tumors. The approach was suggested to improve the accuracy of basic CNN architecture in brain tumor classification. Both of these networks are based on the pre-trained architecture of AlexNet the first hybrid network was AlexNet-SVM and the second was AlexNet-KNN on the dataset of 2880 T1-weighted contrast-enhanced MRI brain scans, this dataset consists of four classes: glioma, meningioma, pituitary and no tumor, where glioma has 829 images, meningioma has 825 images, pituitary has 830 images and lastly no tumor has 396 images. This dataset has been split with 70% for training and 30% for validation and for evaluation, they used a new dataset of the same four classes, each with 100 images. The best accuracy of this comparison was by AlexNet*KNN 98.6%.

In [6], a new method, namely Simulated Annealing-Genetic Algorithms (SA-GA), has been proposed which to determine pertinent features from brain MRI images for tumor classification and segmentation. The suggested methodology combines two optimization methods simulated annealing (SA) and the genetic algorithm (GA) to select the best subset of features for classification and segmentation tasks, by using GA to explore the search space and generate a diverse set of candidate solutions. Then it uses SA to refine the solutions locally and escape from local optima. The hybrid SAGA is val-

idated on two brain tumor datasets, the first combine 83 MRI images with 22 malignant tumors, 32 benign tumors, and 29 non-tumor images, while the second is obtained from the Brain Tumor Image Segmentation (BRATS) challenge dataset that consists of 3064 MRI images with 2134 malignant tumor and 930 normal brain tissue(benign tumor) images. The data were randomly split into 5 subsets of roughly equal size, five-fold cross-validation is used, one subset is sequentially used as the test set, whereas the remaining four subsets are used as training sets. The first was used to evaluate the feature selection methods and to select the most informative features for brain tumor classification and segmentation. The second was used to evaluate the performance of the proposed method for brain tumor classification and segmentation.
They also use different machine learning algorithms, including decision trees, k-nearest neighbors (KNN), support vector machines (SVM), and random forests, for classification and segmentation tasks.

in [7], the focus was on classifying various brain tumors: glioma, meningioma, and pituitary. Different types of tumors tend to show similar appearances, which makes categorization difficult. This problem makes it challenging to use traditional machine learning approaches. So it was described as a new method for using a hybrid deep learning model. The proposed method looked at two different hybridizations. The first model, which is known as SN-SVM, combines SqueezeNet and SVM. The second model is a combination of SqueezeNet and Fine-Tuning, which is known as SN-FT. To evaluate the effectiveness of the technique suggested, MRI scans of the brain were used to analyze a total of 1937 images of glioma tumors, 926 images of meningioma tumors, 926 images of pituitary tumors, and 396 images of a normal brain. The SN-FT model obtained an accuracy of 96.5% , while the SN-SVM model recognition accuracy increased to 98.7% .

In [8], the global average pooling layer was used as an output layer for a pre-trained model VGGNet16. This method was used on figshare brain tumor dataset containing 3064 T1-weighted brain MRI with three classes which are: glioma, meningioma, and pituitary and each class has 1426, 708, 930 training samples respectively and this dataset has been augmented and enhanced, for testing they used 104 samples for meningioma, 214 for glioma and 139 pituitary. They achieved an accuracy of 98.51%

In [9], different methods of machine and deep learning have been evaluated to identify the best method in both handcraft methods and deep learning architecture. In this experiment, the applied image-based dataset comprised 3264 T1-weighted contrast-enhanced MRI images. There are four types of images in this dataset: glioma 926 images, meningioma 937 images, pituitary 901 images, and healthy brain 500 images, this data has been reshaped and augmented. The outcome of this experiment was that KNN had the highest accuracy in the classical methods with 86% and for deep learning classic 2D CNN was better than CAE with accuracy 93%.

In [10], the data that is used consists of 3446 Brain tumor MRI images which are split into four types (classes) 936 MR images of glioma, 1052 MR images of meningioma, 975 pituitary and 493 of no tumor. In order to analyze the data and recognize the types of tumors, four systems with multiple technologies are proposed. These methods include machine learning, deep learning, and hybrid learning. The first system comprises artificial neural network (ANN) and feed-forward neural network (FFNN) algorithms that classify the hybrid features based on combined features of a local binary pattern (LBP), grey-level co-occurrence matrix (GLCM), and discrete wavelet transform (DWT) algorithms. Before the features can be extracted by the LBP, GLCM, and DWT, the data were enhanced by employing Laplacian and average filers to get a more contrasted image highlighting the tumor's edges emerging. To separate the tumor area from the healthy area the adopted region-growing method and Morphological Process were applied. The second system comprises pre-trained GoogleNet and ResNet-50 models for dataset classification. The third system is a hybrid technique between convolution neural network CNN and support vector machine SVM. The CNN is used to extract features and the SVM to classify. It used hybrid GoogleNet with SVM and ResNet-50 with SVM. The final system is a hybrid that uses ResNet50 and GoogleLeNet features together with custom features from the LBP, GLCM, and DWT algorithms to identify representative features and categorize them using ANN and FFNN.

In [11], a new hybrid CNN-based architecture was developed with two techniques for classifying different types of brain tumors from MRI data. The first approach combines an SVM for pattern classification with a pre-trained GoogleNet model of the CNN technique for feature extraction. While the second approach combines a softmax classifier and a highly tuned GoogleNet.

The proposed method was examined using MRI brain tumor images, which included 396 healthy brain images, 708 meningioma images, 1426 glioma images, and 930 images of pituitary tumors. The results showed the superiority of the first model over the second with an accuracy of 98.1% and 93.1% .

The objective of [12] was to recognize and classify MRI brain tumor images utilizing image segmentation, previously trained models, and transfer learning by proposing a new approach based on pre-trained convolution neural networks (CNN) and support vector machines (SVM). The suggested approach uses transfer learning to categorize brain tumors and a range of segmented pictures as augmentation. The segmentation is used by k-means and fuzzy-c-means algorithms. The two pre-trained convolution neural network models, Alexnet and ResNet are used to produce the features. A support vector machine (SVM) is used to classify tumor images after characteristics have been extracted. To assess classification accuracy and processing speed, they trained the designs using segmented pictures and little pre-processing over three epochs. With transfer learning, an experimental performance of 97.34%was achieved.

In [13], an artificial neural network (ANN) along with the Levenberg Marquardt optimizer (LMA) is proposed to classify MRIs and diagnose brain tumors and to reduce and optimize errors. As a pre-processing, the proposed technique converts the MRI image to greyscale, and threshold it. Next, the skull is removed and various morphological procedures are applied. Twelve features are taken from MRIs with an image size of (256,256) using statistical analysis and a grey-level co-occurrence matrix (GLCM), and these features are utilized as inputs for the ANN. To evaluate the proposed method, 670 normal brain, and 670 abnormal brain MRIs are used as input data. This dataset is utilized as input data to assess the suggested approach. The accuracy, sensitivity, specificity, precision, dice, recall, and MSE of the model are 98.7%, 97.61%, 99.7%, 97.61%, 98.6%, 97.61%, and 0.005, respectively.

In [14], the Alexnet model is used to classify MRI images of brain tumors into healthy brain or sick brain. With the aim of improving the original model and increasing its accuracy, the paper describes this in sequential phases where the network uses features generated by 16 extraction methods. After features selection, the features are concatenated and fed into the CNN model to train with. From 274 patients, 7620 images were collected, 6762 im-

ages for train data, and 858 images for test images. By assigning these weight factors to the methods, An accuracy above 99.76% was achieved, which is a 2% improvement using these feature extraction methods compared to the original method.

Overall, brain tumor detection has been an interesting task for many researchers, and they have been trying different methods and solutions to obtain the best performance with different datasets, and most of these methods have been deep learning methods. Therefore, we have tried a different approach by focusing mostly on training machine learning models and a few deep learning models with different dataset to see the performance of machine learning and deep learning in brain tumor detection.
Table 2.1 summarize this chapter.

Papers	Methods	Data-set	Accuracy
[1]	AlexNet-KNN	figshare: 2880 T1-weighted	98.6%
[2]	Simulated Annealing Genetic Algorithms (SA-GA)	combine between figshare & from the Harvard Medical School brain atlas	95.65%
[3]	Squeeze Net and SVM		98.7%
[4]	Fine-tuned VGGNet16 + Global average pooling	figshare: 3064 T1-weighted brain MRI	98.51%
[5]	CNN	Figshare: 3064 T1-weighted brain MRI	93%
[6]	ResNet-50 +GoogleLeNet + LBP, GLCM, DWT and ANN+FFNN	Collected from Tianjin Medical University General Hospital and Nanfang Hospital, China	99.9%
[7]	GoogleNet + CNN	T1-weighted contrast-enhanced MRI	98.1%
[8]	Alexnet and ResNet +SVM	from Kaggle	97.34%
[9]	ANN with Levenberg Marquardt optimizer (LMA)	figshare dataset	98.7%
[10]	Alexnet +CNN	Collected from imaging centers	99.76%

Table 2.1: Summary of each article

Chapter 3

Description of AI methods

In this chapter, we define some methods of image processing, machine learning, and deep learning which we have used in our work and the reason for choosing such methods.

3.1 Image Processing

Image processing is the technique to manipulate and analyze digital images, it includes many methods and operations such as enhancement, reconstruction, segmentation, compression, and recognition of images. The goal of image processing is to extract features from images and use this useful information in various areas such as medical imaging, robotics, computer vision, and surveillance.
There are many interesting methods in image processing, and among the common and powerful methods are Scale-Invariant Feature Transform (SIFT) and Speed-Up Robust Features (SURF).

3.1.1 Scale-Invariant Feature Transform (SIFT)

SIFT [15] is an algorithm for detecting and describing local image features. These descriptors are scale, rotation, and brightness invariant. The SIFT algorithm consists of several steps which are:

1. **Scale Space :** Refers to the representation of an image at different scales and this step involves multiple smoothing and resizing to versions

of the image. Applying the difference of Gaussian (DoG) to detect potential interest points in certain regions.

2. **Key-points localization :** In this step, we apply a threshold to eliminate the low-intensity pixel and locate the key-point in the coordination of (x, σ) where x is the location of the pixel and σ is the scale where it has been found.

3. **Orientation assignment :** Here we draw a circle with the radius of the scale of the interest point that has been found in step 2 and make a grid in that location and get the orientation of this grid and the highest orientation is the orientation of that key-point.

4. **Key-point descriptor :** Lastly, SIFT makes a descriptor to the key points based on the previous steps and this key-point is represented in a 128 dimension because the region of interest is divided into 16 sub-regions and each sub-region has 8 bins so $16x8 = 128$

128 dimensions require a high computation and time expensive which is why it has been developed to Speed-Up Robust Features (SURF) to speed up the computation of finding and matching key-points and it is also the cause that made us pick SURF over SIFT [16].

3.1.2 Speed-Up Robust Features (SURF)

Speed-up robust features (SURF) [17] is an image processing method used for detecting and describing key-points or interest points in digital images. The SURF algorithm performs a series of steps to localize and describe key-points and SURF uses the integral Image instead of the original image.

Integral of image

The integral of the image is the sum of all pixels on the left and above the current location of the pixel including it.

$$II(x, y) = \sum_{x' \leq x, y' \leq y} I(x', y')$$

Figure 3.1 below shows the transaction from the original image to the integral image.

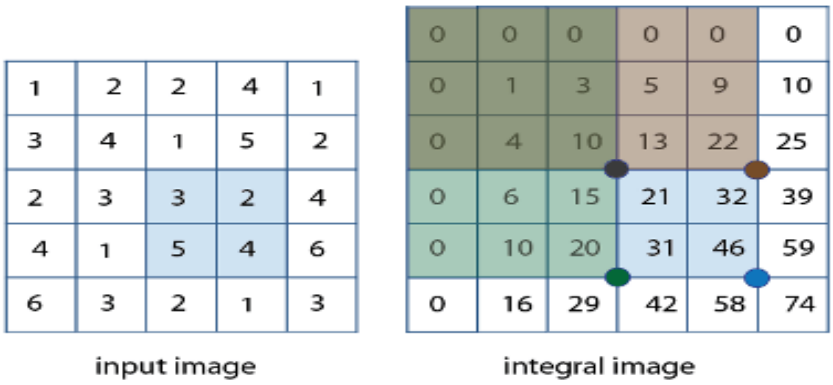


Figure 3.1: Image showing the output of integral image 11

Fast Hessian detector

The value of the determinant of the Hessian matrix is computed for every pixel using the integral image. It is used for both localization and scale detectors.

$$H(x,\sigma) = \begin{bmatrix} L_x x(x,\sigma) & L_y x(x,\sigma) \\ L_x y(x,\sigma) & L_y y(x,\sigma) \end{bmatrix}$$

x : Location of pixel.
L : Laplacian of gaussian.
σ : The scale.

The usage of the integral image is when computing Laplacian of Gaussian where in this case a box filter has been used and these last are an approximation of the Gaussian second-order derivatives. The initial scale of $\sigma = 1.2$ and the initial kernel size is 9x9.

The scale-space is the method of finding the same detector in different sizes, and since surf uses box filters and the integral image it can use any kernel of any size directly on the original image. The scale of σ and kernel size have positive relationship.

After this process, You check if the same region in the image has a high intensity With different of gaussian (DoF).

These steps are for the localization of a key-point. Now to give this key-point a description surf operates in two steps.

Orientation Assignment

To find the orientation of a key-point we have to:

1. Draw a circle of 6s radius around the key-point (s is the scale of which the key-point has been found).
2. Make a grid on the circle with points separated by a distance of s this makes approximately 100 points per grid.
3. Convolve this grid with the Haar wavelet filter, this filter has the size of 4s and it is computed using the integral image.
4. Centring the outcome of step 3 by Gaussian of $stf = 2.5s$.
5. Calculate the sum of all responses of the Gaussian centering in a window sliding of 60 degrees.
6. The dominant orientation is estimated with the highest sum.

Computing descriptor

We first create a window of 20s around the key-point with orientation along the dominant vector. After that, we split this region into 4x4 sub-regions each sub-region sampled with a 5x5 grid separated evenly, and then we apply a Haar wavelet with the size of 2s, and this response is centered with Gaussian $\sigma = 3.3s$ and resulting a descriptor of 64 feature vector.

3.2 Machine Learning

Machine learning is the use of mathematical and statistical algorithms to analyze data and learn patterns and relationships within it to make predictions or curtain types of actions. These algorithms are designed to automatically identify the features and choose their behavior based on the data

patterns and relationships. Machine learning is used in a wide range of applications we mention: image recognition, natural language processing, recommendation systems, and many others.
There are many algorithms and methods used in machine learning for example support vector machine (SVM), decision tree, and logistic regression.

3.2.1 K-Means

K-means [18] is a clustering method that is used to group or cluster a partition of data into one group, the number of groups or clusters is defined by K. The algorithm works iteratively assigning each data point to the nearest centroid or center of a cluster.
K-means algorithms work according to the following steps:

1. initializing the K centroids randomly from the datasets.
   In this step after initializing the K number of clusters, k-means define random points from datasets equal to the K number.

2. Assign each data point to the nearest centroid based on the Euclidean distance or any distance formula.
   The following step is to define each cluster with its nearest points using any distance formula and the most common one is Euclidean distance where this last one is defined by in general where $P_1 = (x_1, ..., y_1)$ and $P_2 = (x_2, ..., y_2)$:
   $$d(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + ... + (y_2 - y_1)^2}$$

3. Recalculate the centroid of each cluster by taking the mean of all data points in that cluster.
   After we define the points of the cluster we calculate the mean of each cluster to determine the centroid of the clusters and we calculate the centroids by the following equation:
   $$M = \frac{1}{n} \sum_{i=1}^{n} P_i$$

4. Repeat steps 2 and 3 until the centroids no longer change, or until the maximum number of iterations is reached.

In Figure 3.2 The output of K-means clustering.

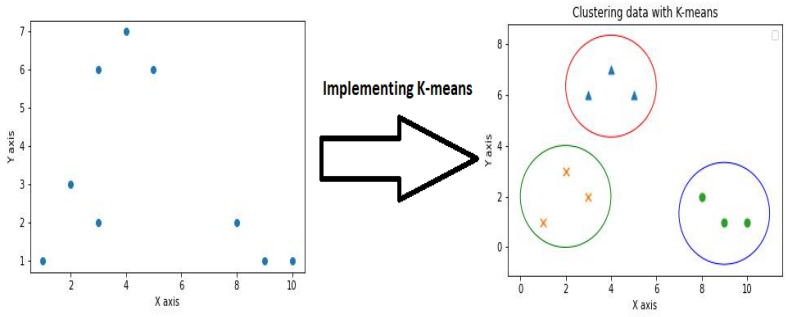


Figure 3.2: Image showing the output of K-means clustering. Different colors represent different classes K-means assigned the data samples two (three classes in this example)

3.2.2 Gaussian Mixture Model GMM

The Gaussian Mixture Model (GMM) [19] is a probabilistic model representing a probability distribution as a mixture of multiple Gaussian distributions, each with its own mean and covariance matrix. It is a powerful algorithm used for clustering and density estimation tasks.
The GMM algorithm allows for soft clustering, meaning that each data point can have a probability distribution over the clusters rather than being assigned to a single cluster. The main principle of GMM can be summarized in the following steps:

1. **Initialization:** Set the GMM's parameters, such as the number of Gaussian components(clusters) K, their means μ, covariances Σ, and mixing coefficientsπ, to their initial values. This can be carried out at random or using another initialization technique, like K-means clustering.

2. **Expectation-Maximization (EM):** The GMM algorithm uses EM algorithm to estimate the parameters of the model iteratively. The EM algorithm consists of two steps: the E-step and the M-step.
   a. E-step (Expectation): In this step, the algorithm computes the responsibility of each Gaussian component for generating each data

point. It is meant to estimate the probabilities that a data point belongs to each component. The responsibility is the posterior probability that a data point belongs to a specific Gaussian component, given the current parameter estimates. The responsibility got by:

$$\gamma(i,j) = \frac{\pi(j) \cdot \mathcal{N}(x(i)|\mu(j), \Sigma(j))}{\sum_{k=1}^{K} \pi(k) \cdot \mathcal{N}(x(i)|\mu(k), \Sigma(k))}$$

where $\mathcal{N}(x(i)|\mu(j), \Sigma(j))$ is the probability density function (PDF) of the Gaussian component j evaluated at data point x(i).

b. M-step (Maximization): Update the parameters of the Gaussian components based on the responsibilities computed in the E-step. It computes the new estimates of the means, covariances, and mixing coefficients by maximizing the expected log-likelihood.
Update the mixing coefficients:

$$\pi(j) = \frac{\sum_{i=1}^{N} \gamma(i,j)}{N}$$

Update the means:

$$\mu(j) = \frac{\sum_{i=1}^{N} \gamma(i,j) \cdot x(i)}{\sum_{i=1}^{N} \gamma(i,j)}$$

Update the covariances:

$$\Sigma(j) = \frac{\sum_{i=1}^{N} \gamma(i,j) \cdot (x(i) - \mu(j)) \cdot (x(i) - \mu(j))^T}{\sum_{i=1}^{N} \gamma(i,j)}$$

3. **Convergence:** Steps a and b are repeated until convergence. The convergence criterion can be based on the change in log-likelihood or the change in the parameter estimates between iterations.

4. **Prediction:** After the GMM parameters converge, the model can be used to estimate the density of the data at any given position or assign new data points to the Gaussian components.

Determining the appropriate number of Gaussian components can be challenging and may require additional techniques, such as model selection criteria (e.g., Bayesian Information Criterion or Akaike Information Criterion). Figure 3.3 illustrates the result of the GMM clustering method.

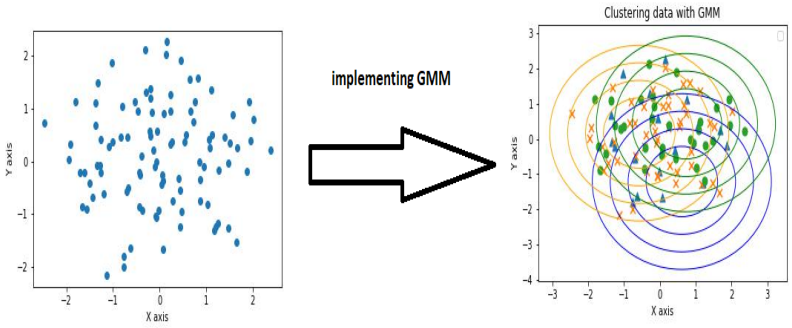


Figure 3.3: Image showing the output of GMM clustering

3.2.3 Support Vector Machine

Support vector machine (SVM) [20] is a supervised machine learning algorithm that is looking to determine the best hyperplane that separates the different classes or groups in the dataset. SVM tries to find the hyperplane that best separates classes while maximizing the margin at the same time minimizing the distance between support vectors. Support vector coordination at simply the closest two points to the margin from each class.
The best hyperplane fulfills the following equation :

$$w^T x - b = 0$$

w: the weights.
x: data points.
b: the bias.

The hyperplane and support vectors are pointed out in Figure 3.4 below.

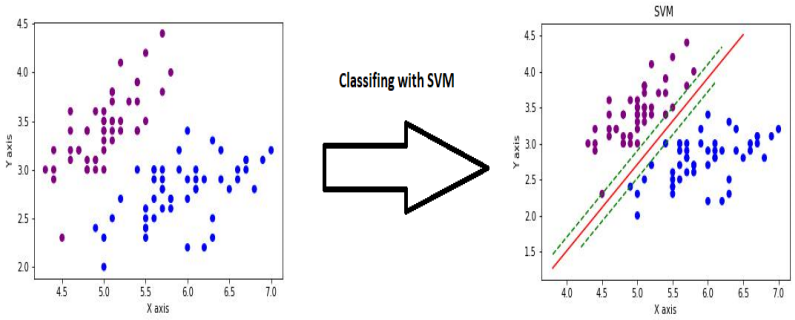


Figure 3.4: Image showing the output of SVM algorithm

3.2.4 Decision Trees

Decision trees [21] are machine learning algorithms used for classification and regression tasks. It works by partitioning recursively a data of smaller subset based on the values of the input features until a decision can be made about the target variable.
In a classification job, a decision tree is constructed by continually dividing the input data into branches based on the values of the input features. The method selects the feature at each split that gives the target variable the highest information gain, i.e., reduces impurity or entropy in the data the most.

$$IG(D, s) = Impurity(D) - \frac{N_{left}}{N} Impurity(D_{left}) - \frac{N_{right}}{N} Impurity(D_{right})$$

$IG(D, s)$: The information gain of a split.
$Impurity(D)$: The impurity of the dataset before the split.
N_{left}: The number of instances in the left node child.
N_{right}: The number of instances in right child nodes.
N: The total number of instances in the dataset.
$Impurity(D_{left})$: The impurities of the left child nodes.
$Impurity(D_{right})$: The impurities of right child nodes.

A simple example of a decision tree is shown below in Figure 3.5.

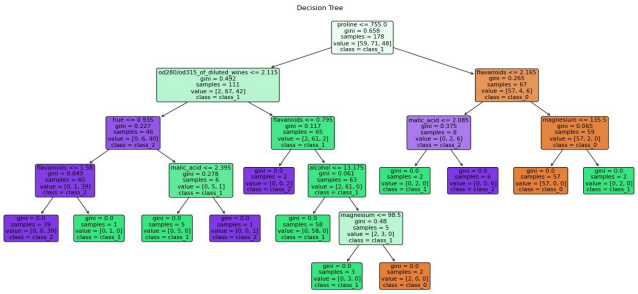


Figure 3.5: Image showing decision tree algorithm

3.2.5 Logistic Regression

In supervised machine learning algorithms, Logistic regression [22] is defined as a statistical model used for binary classification problems, where the goal is to predict the probability of an event occurring or not occurring. by fitting a line to separate two classes.
The logistic regression model can be represented mathematically as:

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n)}}$$

$P(Y = 1|X)$: The probability of the dependent variable Y being 1 given the values of the independent variables X.
$\beta_0, \beta_1, \beta_2, \ldots, \beta_n$: The coefficients or weights associated with the independent variables.
$X_1, X_2, \ldots, X_n$: The values of the independent variables.

The likelihood function in logistic regression is based on the assumption that the observations are independent and identically distributed. It can be represented as:

$$L(\beta) = \prod [P(Y = 1|X)^Y \cdot (1 - P(Y = 1|X))^{(1-Y)}]$$

$L(\beta)$: The likelihood function that depends on the coefficients β.
$\prod$: The product operator, which multiplies together all the terms.
$P(Y = 1|X)$: The predicted probability of Y being 1 given the value of X.

Logistic regression examines the link between the available data (referred to as independent variables) and the likelihood of the event (referred to as the dependent variable) in order to create predictions. Using coefficients, it calculates the influence of each independent variable on the likelihood.
In practice, it's easier to work with the log-likelihood function, which is the natural logarithm of the likelihood function:

$$LL(\beta) = \log L(\beta) = \sum [Y \cdot \log(P(Y=1|X)) + (1-Y) \cdot \log(1-P(Y=1|X))]$$

$LL(\beta)$: The log-likelihood function.
Σ: The summation operator, adds up all the terms.
Y: The dependent variable.
$P(Y=1|X)$: The predicted probability of Y being 1 given the value of X.

The Figure 3.6 demonstrate how logistic regression classifies data.

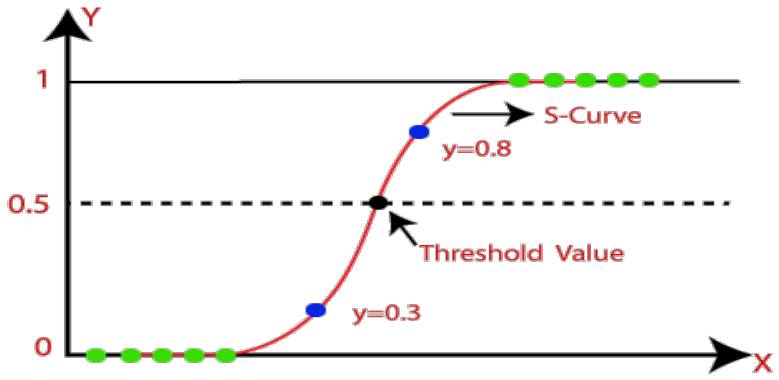


Figure 3.6: Image showing the output of Logistic Regression algorithm 15

3.3 Deep Learning

Deep learning is a black box that uses multiple layers of artificial neural networks connected with each other to analyze and learn from complex data. Deep learning is designed to automatically learn the structure and processing of large amounts of data. It can be used for tasks such as facial recognition, speech recognition, autonomous, and many other tasks.
There are various deep learning architectures for instance convolution neural networks (CNN).

3.3.1 Artificial Neural Network (ANN)

Artificial neural networks (ANNs) are computational models inspired by the structure and function of biological neural networks in the human brain. ANNs are composed of interconnected nodes, called artificial neurons or "units," organized into layers as illustrated in Figure 3.7. Information flows through the network, with each neuron receiving input signals, applying a transformation, and passing the result to the next layer. The connections between neurons have associated weights that adjust during the learning process to optimize the network's performance.
Over time, ANNs evolved with advancements in computing power and the availability of large-scale datasets. So new architectures have appeared, such as convolution neural networks (CNNs) and recurrent neural networks (RNNs).
Figure 3.7 shows a simple artificial neural network architecture.

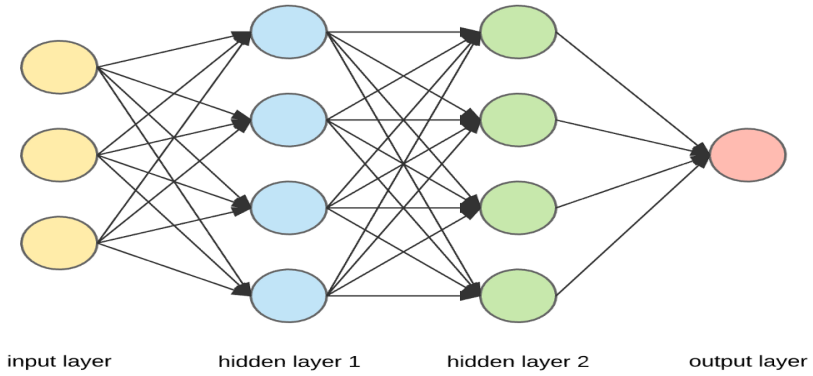


Figure 3.7: Image showing an artificial neural network 16

3.3.2 Convolution neural network (CNN)

Convolution neural network (CNN) is deep learning architecture used mainly for image and video. CNN consists of convolution, pooling, and fully connected layers as shown in Figure 3.8.

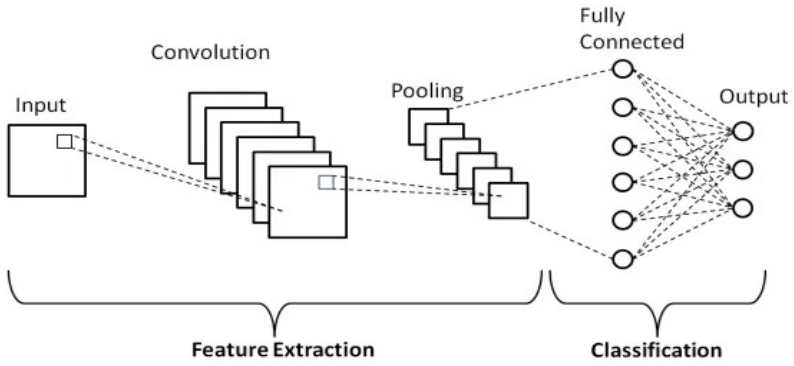


Figure 3.8: Image showing a convolution neural network 17

Convolution layer

Convolution layers are the core building blocks of a CNN. They consist of filters or kernels, which are small windows that slide over the input data and perform element-wise multiplications and summations. Convolution layers are used to take local features from the input data, such as edges, textures, and patterns.
Applying the convolution operation with stride S, padding P and size of filter F results in an output tensor of size $\left(\frac{W-F+2P}{S}+1, \frac{H-F+2P}{S}+1, N\right)$, where N is the number of filters in the layer. The output of the convolution layer gets by:

$$\Phi(i,j,k) = \sum_{x}\sum_{y}\sum_{c}(\chi(x,y,c)\cdot\Omega(i,j,c)) + B(k)$$

pooling layer

Following convolution layers, pooling layers seek to condense the input's spatial dimensions. Information is gathered within a neighborhood and reduced in size through the use of pooling methods like average or max pooling.

As a result, the computational burden is reduced and the learned characteristics are strengthened against subtle input fluctuations.

$$\Phi(i, j, k) = \max_{p} \max_{q} (\chi(i \cdot s + p, j \cdot s + q, k))$$

fully connected layers

The output of the convolution and pooling layers is typically passed through one or more fully connected layers. These layers connect every neuron in one layer to every neuron in the next layer, allowing for high-level representations and making predictions based on the learned features.

$$\mathbf{Y} = \text{Activation}(\mathbf{W} \cdot \mathbf{X} + \mathbf{B})$$

Training a CNN involves forward propagation, where data flows through the network, and back-propagation, where the error is propagated backward to update the network's parameters (weights and biases). The most common optimization algorithm used in CNN training is gradient descent, with variants like stochastic gradient descent (SGD) or adaptive optimization methods (e.g., Adam).
CNNs have shown exceptional performance in various computer vision tasks, including image classification, object detection, semantic segmentation, and image generation.

In conclusion, the methods that we have defined have been used in this work starting by processing the images with the surf method followed by making machine learning models logistic regression, SVM, and decision trees based on them. And in Deep learning convolution neural networks models have been made without the need for the SURF.

Chapter 4

Methods

In this chapter, we will explain the steps or the flow of our work in both machine learning where we used SVM, decision tree, and logistic regression as our classifier methods, and deep learning specifically convolution neural network, at the end will be comparing the results of the methods.

It is worth mentioning that the input of our models is MRI images. MRI consists of multiple channels stacked on top of each other and represented as an array of grayscale images, and has a different angle with different parameters and contrast.

4.1 Methods based on machine Learning

This section discusses in detail the steps our machine-learning-based method consists of, as shown in Figure 4.1.

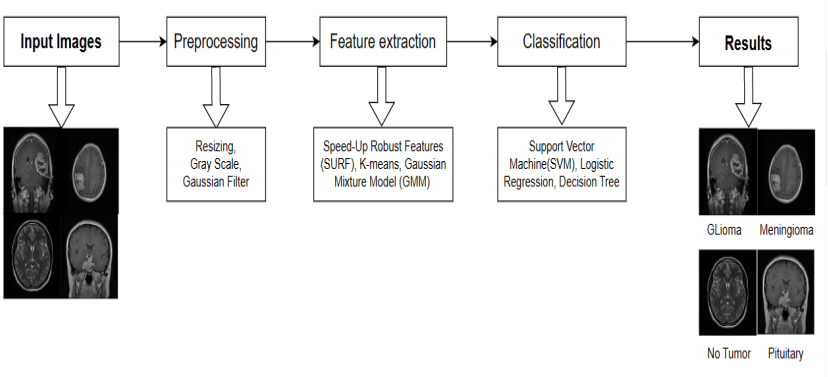


Figure 4.1: Image Illustrating the flow of the work with machine learning methods

4.1.1 Pre-processing

The image, before it is fed to a model, goes through some processes which are: RGB to gray conversion, and then resizing.

Gray scale

Gray scaling is the process of converting a color to one channel image. We have performed such a procedure to alleviate the computation. The following formula has been used :

$$G_I(x, y) = I_R(x, y) * 0.299 + I_G(x, y) * 0.587 + I_B(x, y) * 0.114$$

Figure 4.2 shows an example of converting RGB image to gray level.

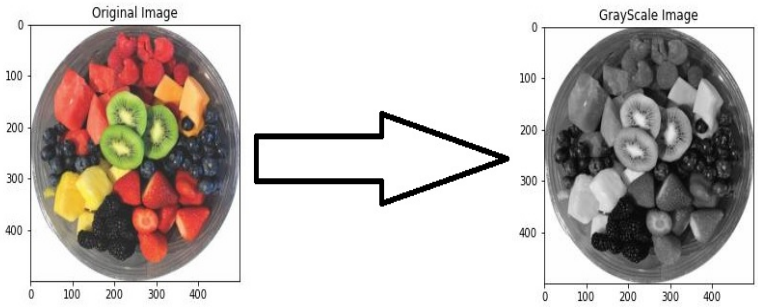


Figure 4.2: Image showing RGB to Gray Scale

Resize

This operation aims at reducing or augmenting the size of the original image for various purposes (reducing calculation, fitting the input size of some model, etc.). We consider resizing the images because the shape of the images was unbalanced. This causes extracting more features in one image compared to others. Among many methods, bilinear interpolation provides the best compromise between processing speed and image quality. The main idea behind this method is:

1. Detect the four closest pixels.
2. Calculate the distance between the target pixel and each of the four pixels. The distance is considered as weights and the sum of these weights is equal to 1.
3. Multiply each known pixel point's intensity value by its matching weight.
4. Add the weighted intensity values from the previous step.

Figure 4.3 below shows the process of bilinear interpolation.

We used resize so the descriptor matrices have values close to each other and models will not have any bias.
Before resizing an image of the class pituitary has the shape of (900,920), SURF extracted 3876 key-points.
Therefore from an image of the class no tumor with the shape of (192,192), SURF extracted 387 interest points as an output.

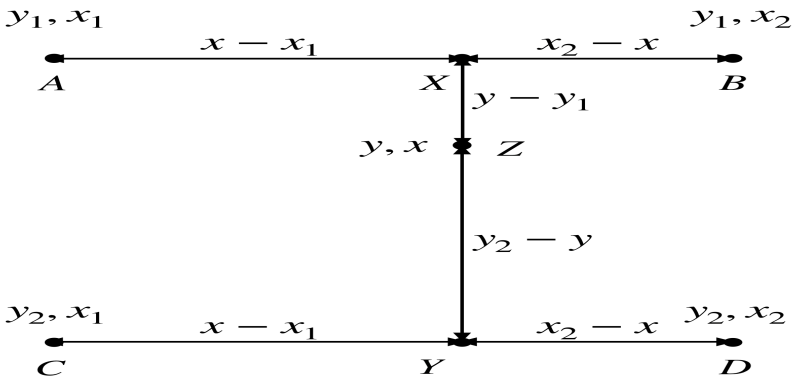


Figure 4.3: Image showing bilinear interpolation 19

Noise removal

Removing noise from images is a crucial step in image processing to improve their quality and facilitate accurate analysis. The Gaussian filter, generated using the Gaussian distribution function, is a widely employed technique for this purpose. It is represented by the following formula:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

By convolving the image with the Gaussian filter, high-frequency noise is attenuated while preserving the essential structures and edges. This smoothing operation significantly enhances image clarity, reduces noise interference, and prepares the image for subsequent feature extraction and analysis. The application of the Gaussian filter plays a vital role in ensuring reliable and robust image processing outcomes.

4.1.2 Feature extraction

As mentioned earlier, the extraction of meaningful features from data is crucial for training machine learning models. In this work, the SURF method has been employed as the feature extraction technique. The decision to use the SURF method is based on the following reasons:

1. Invariant for different conditions such as size, orientation, brightness ... etc.

2. The efficiency in both time and computations.

3. The accurate descriptor representation.

After extracting the features from the images, we construct a bag of features (BoF). Bag of features is a method based on a bag of visual words (BoVW). Its aim is to convert the matrix of descriptors into a feature vector so all images will be represented as a feature vector of the same dimension. To construct BoF, we go through the following steps :

1. **Stacking descriptors :** Load all descriptors and stack them on top of each other and that resulted in a large matrix of the size (1839255,64).

2. **Clustering :** We used two clustering methods: 1) hard clustering method which is K-means 2) soft clustering method which is GMM.

3. **Stacking Frequencies :** We collected the frequencies and with that, we produced our BoF.

The reason behind using such a method is to convert each of the descriptor matrices of the image into a feature vector with the same length and reduce the number of features.

4.1.3 Classification

There are many machine learning classifications methods we list them:

1. **K-Nearest neighbor (KNN):** KNN is a method that takes no training time but takes a lot of time to test that is why called a lazy method which is we did not test it.

2. **Naive Bayes :** Naive Bayes is a method based on the theory that features of the data are not dependent and since our data features are dependent naive Bayes will not give good results.

3. **Decision Tree :** The decision tree is usually used to describe the data and can be used to handle a wide range of data types which made it a powerful tool.

4. **Random forest :** Random forest is a multi-decision tree so the training time will be large and also the testing time since it takes the most voted the prediction of as an outcome.

5. **Support Vector Machine (SVM) :** SVM is the most commonly used supervised machine learning algorithm for classification for its utility and accurate predictions that is what it considered an effective method.

6. **Logistic Regression :** Logistic regression is a machine learning method that is used for simplicity, low computation time for training, and probability modeling.

And from this list, we have used a support vector machine (SVM), decision tree, and logistic regression.

4.2 Methods based on deep learning

In this section, we introduce the DL methods which are CNN and transfer learning, and their models. In all of these models, we opted for the Sparse Categorical Cross-entropy loss function optimized and minimized by the Adam optimizer.

4.2.1 The loss and the optimizer functions that used

As we said above, we will now see two concepts or functions that complement each other. We benefited from them in our training of our models. The first is for loss or calculating the extent of the model's error, and the second is for improving and optimizing it. That are Sparse Categorical Cross-entropy and Adam functions.

Categorical Cross-entropy

To know the Sparse Categorical Cross-entropy function, we must know Categorical Cross-entropy first. So, categorical cross-entropy is a commonly used loss function in machine learning, particularly in classification problems where the output variable is categorical. It calculates the difference between the target variable's actual probability distribution and the predicted probability distribution. In the context of multi-class classification, categorical cross-entropy calculates the loss by considering each class independently.
The formula for categorical cross-entropy is as follows:

$$\text{CCE} = -\sum_{i}(y_i \cdot \log(p_i))$$

CCE: The categorical cross-entropy,
y_i: The true probability (or binary indicator) of class i.
p_i: The predicted probability of class i.
the sum is taken over all classes.

In the categorical cross-entropy function when the true labels y are given as integers rather than as one-hot encoded vectors, we are in the Sparse categorical cross-entropy. In this case, we only need to compute the loss for the correct class rather than summing over all classes as in the categorical cross-entropy. The formula for sparse categorical cross-entropy is:

$$\text{SCCE} = -\log(p_y)$$

$SCCE$: The sparse categorical cross-entropy loss and p_y is the predicted probability for the correct class y.
Sparse categorical cross-entropy is particularly helpful when having several classes in a classification task and it would be impossible to record the real labels as one-hot encoded vectors because it is memory-intensive. It allows training the model well while directly providing the integer labels as the real class values.

Adam optimizer

The Adaptive Moment Estimation (Adam) is a common optimization algorithm used in deep learning for updating the parameters of a model during the training process. It refers to the adaptive updates of the learning rate and the momentum terms.
The Adam optimizer combines the benefits of both AdaGrad (which adapts the learning rates per parameter) and RMSProp (which uses the moving average of squared gradients).

a. Learning rates : Adaptive Learning Rates are the process of adjusting and changing the learning rate, which determines the step size of parameter updates, based on the observed behavior of the gradients during training. In other words, it seeks to determine a suitable learning rate for each parameter that enables effective convergence and prevents becoming stuck in local optima.

b. Momentum : Momentum is a concept that denotes the acceleration of the optimization process and it represents a moving average of past gradients. It accumulates the gradients over previous iterations and adds a fraction of the accumulated gradient to the current gradient for updating the parameters. The momentum term effectively determines how much influence the previous gradients have on the current update. In situations when the gradients are noisy or the loss landscape is complex, optimization methods can move more steadily toward the optimum and speed up convergence.

The Adam optimizer tries to adjust the learning rate for each parameter based on the historical gradients and the historical squared gradients of that parameter.
Each parameter updating procedure in Adam can be summed up as follows:

$$
\begin{aligned}
m_t &= \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \\
v_t &= \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2 \\
\hat{m}_t &= \frac{m_t}{1 - \beta_1^t} \\
\hat{v}_t &= \frac{v_t}{1 - \beta_2^t} \\
w_{t+1} &= w_t - \frac{\text{learning\_rate} \cdot \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}
\end{aligned}
$$

Where:
m_t :Represent the first moment(mean)
v_t :The second moment (variance)
g_t :is the gradient at iteration t.
$\hat{m}_t$,$\hat{v}_t$:are the bias-corrected estimates of the moments.
w_t, w_{t+1} :are the parameters at iterations t,$t+1$, respectively.
β_1, β_2, ϵ, `learning_rate`: are hyperparameters specific to the Adam optimizer.

The activation function

An activation function is a mathematical function used in neural networks to add non-linearity and allow the network to learn complex patterns and make predictions based on the input data. Whereas in CNNs, the activation functions are applied to the output of convolutional layers and fully connected layers for the same reason.

1. **Rectified linear unit ReLU:** Function outputs for positive inputs may range from 0 to infinity. However, when the input is 0 or a negative value, the function output is 0 and it interferes with back-propagation. This problem is known as the dying ReLU problem.

Its formula is given as:

$$f(x) = \max(0, x)$$

2. **Softmax function:** The softmax function, which determines the relative probability of each class, is defined as a mixture of several sigmoids. In the case of multi-class classification, it is most frequently utilized as an activation function for the output layer of the neural network. Mathematically it can be represented as:

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}}$$

In Dl methods, we used the softmax activation function in output layers (last dense layer), while we used the ReLU in other dense layers.

4.2.2 Deep learning using CNN architectures

The following image represents the procedure flowchart of our method for classifying brain tumor MRI images.

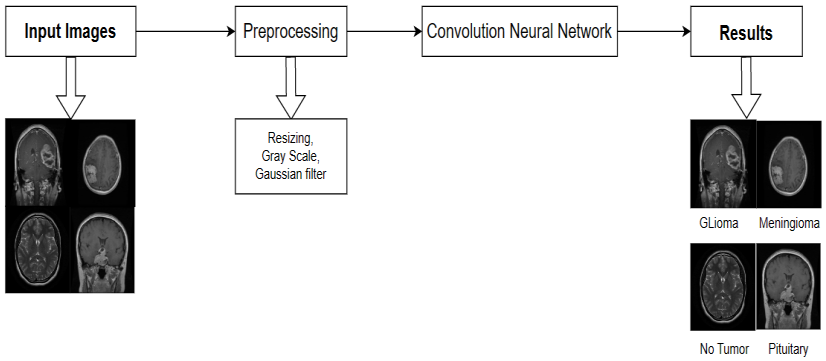


Figure 4.4: Image Illustrating the flow of the work with deep learning methods

After collecting the data, we pre-process it, which is the resizing and gray-scaling. We feed the images obtained from the pre-processing to different CNN models to train them.
In the next step, we classified the different types of brain tumors using a

number of models containing 3 to 5 convolution layers to identify and obtain the most important features and accurate classification. Our assumption is that the greater the depth of the CNN (i,e the more convolution layers), the better to correctly identify more portion of the image. So the reason we did not use one layer or two layers, and according to experience, was that it was not sufficient in number to determine the features, and therefore the classification would be inaccurate.
Although our CNN structures vary, many of their steps or parameters do not. For instance, all models have the same number and size of convolution filters with the same strides parameter in the pooling layer. Additionally, they all share an output layer that activates four neurons to depict the different kinds of brain tumors using softmax. Here are some specifications of our CNN models:

CNN Models	Conv_pool layers	Flattning size	1st danse layer	2nd danse layer	Paramétrés of training
1st CNN model	3	(23,23,64)=33856	300 nuerons		10,232,800
2nd CNN model	3	(23,23,64)=33856	200 nuerons		6,846,700
3rd CNN model	3	(23,23,64)=33856	100 nuerons		3,460,600
4th CNN model	4	(10,10,64)=6400	100 nuerons		751,928
5th CNN model	4	(10,10,64)=6400	100 nuerons	12 nuerons	752,788
6th CNN model	5	(4,4,64)=1024	200 nuerons		354,156
7th CNN model	5	(4,4,64)=1024	100 nuerons		251,256
8th CNN model	5	(4,4,64)=1024	300 nuerons		457,056
9th CNN model	5	(4,4,64)=1024	300 nuerons	30 nuerons	456,006
10th CNN model	6	(1,1,64)=64	200 nuerons		199,084
11th CNN model	6	(1,1,64)=64	40 nuerons		188,044
11th CNN model	6	(1,1,64)=64	20 nuerons		186,664

Table 4.1: Summaries of CNN models

The first model has three convolution aind pooling layers which have given

(23,23,64) as data size, with one dense layer having 300 neurons based on the RELU activator. Due to the fact that the majority of the dense layer's neurons had a value of 0, we reduced its size in the second and third CNN models to 200 and 100 neurons.
In order to lower the error, we decided to delve deeper by adding convolution layers in the next models.
Let us recall that each CNN model ends with a pooling layer, then we convert the multidimensional output into a one-dimensional vector with flatten layer.

4.2.3 Using transfer learning by ResNEt50 pre-trained model

Transference learning is a machine learning approach that takes advantage of pre-trained models to solve new tasks by transferring the knowledge of the pre-trained model that has been learned through previous experiences and giving its weights to the new models. These weights are used as initial weights to a new model, by adding new layers, the parameters of these new layers are updated while others are not to adapt to the new task. In this way, with the aim of classifying brain tumors, that is why we use the ResNet50 pre-trained model.
ResNet50 is a variant of the ResNet model architecture that consists of 50 layers. It was introduced by Microsoft Research in 2015 and has become one of the most widely used and influential deep learning models in computer vision.
It was chosen because it is thought to be somewhat deep and because its depth enables the capture and representation of complex patterns and features in the data, which can provide models with higher accuracy than shallow models. This is due to the fact that residual connections are introduced to address the vanishing scaling problem. These connections make it possible for the model to be aware of the remaining mappings, which facilitates very deep network training and optimization. Because of this, it can gain more depth without experiencing performance deterioration. In addition, ResNet50 is trained on big datasets like ImageNet, which has millions of labeled images categorized into thousands of classes.

The images that we used in this method are in RGB format and have

been resized to (200,200,3). Regarding the models, we used the outputs of the ResNet50 as an input to them with its layer frozen, followed by a flattening layer, then dense layers according to each model, and finally an output layer with 4 neurons representing the types of brain tumors.

To summarize, we converted the MRI images from images of 3 channels to 1 channel with the grayscale formula and resized them, therefore removing the noise with the Gaussian kernel. After, we extracted the features with SURF, clustered them with K-means and GMM, and encoded each feature matrix of an image to the feature vector, collecting all these feature vectors into one large matrix called codebook, to give this last to a machine learning model. For deep learning, we converted to grayscale and resized and gave them to the CNN model and pre-trained model.

Chapter 5

Experiment and results

In this chapter, we will be describing the dataset that has been used in this work, the materials that have been used, the used parameters, and analyze the results.

5.1 Experiment set-up

5.1.1 Dataset

This data set was taken from Kaggle [23] and consists of 7023 MRI brain images and it is a combination of three data sets named: figshare, SARTAJ, and Br35H. It comprises four classes which are: glioma, meningioma, pituitary, and no tumor. These classes have 1321, 1339, 1457, and 1595 training samples and 300, 306, 300, and 405 testing samples, respectively. No tumor samples were taken from the Br35H data set, whereas, Glioma samples were taken from the figshare data set.

5.1.2 Materials

This experiment has been done in both an online editor (Google Colab) and a local machine. The machine that has been used for this work has the following specifications :

1. 3070 Ti GPU.
2. AMD Ryzen 5 3600X 6-Core Processor 3.79 GHz.

3. 16 Gb RAM.

5.1.3 Metric

In the context of machine learning for medical diagnosis, the performance metric might change based on the specific task and issue being addressed.

1. **Confusion matrix:** The confusion matrix is one of the most common metrics, which is a square matrix that is used to evaluate the performance of a classification model and summarizes its performance by counting the number of correct and incorrect predictions for each class. The confusion matrix is typically organized as shown in Table 5.1:

		Predicted Class	
		Positive	Negative
Actual Class	Positive	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

Table 5.1: Confusion Matrix

The confusion matrix's components are described in the following order:
True Positives (TP): The number of situations where a positive outcome was accurately predicted.
False Positives (FP): The number of instances that are falsely identified as positive.
True Negatives (TN): The number of instances that are correctly predicted as negative.
False Negatives (FN): The number of situations that were mistakenly projected as negative.

2. **Accuracy:** By comparing the proportion of accurate forecasts to all predictions, accuracy assesses how accurate the model's predictions are overall. Accuracy might not be enough, though, when working with datasets that are unbalanced. Calculated as:

$$\frac{(TP + TN)}{(TP + FP + FN + TN)}$$

3. **Sensitivity/Recall:** It measures the model's ability to correctly identify positive cases. It is calculated as:

$$\frac{TP}{(TP + FN)}$$

4. **Precision:** It assesses how well the model can recognize positive cases among the anticipated positive cases. It is determined as:

$$\frac{TP}{(TP + FP)}$$

5. **F1 Score:** F1 Score is the harmonic mean of precision and recall, providing a fair assessment of both measurements. It is advantageous when classes are unbalanced since it combines precision and recalls into a single value. Calculated as 2:

$$\frac{2 * (Precision * Recall)}{(Precision + Recall)}$$

6. **Area Under the Receiver Operating Characteristic Curve (AUC-ROC):**It evaluates the model's ability to distinguish between positive and negative cases by plotting the true positive rate against the false positive rate.

5.2 Experiment results

Our experiment as mentioned above has been split into two categories. The first one evaluates methods that are based on handcrafted features, and the second one deal with methods based on CNN models.

We note that our classes are labeled as 0, 1, 2, and 3 for no tumor, glioma, meningioma, and pituitary respectively

5.2.1 Machine Learning experiment

In this part, we evaluate and analyze results yielded by methods not based on artificial networks, which are discussed in the chapter Methods above.

Firstly, in the pre-process section we converted the images from RGB to gray level and then resized them to (200,200) and used Gaussian kernel on it with $\sigma = 0.3$ to reduce the noise. Then, we create the feature vectors once with k-means clustering and another with GMM, and collect them into two codebooks one for features of K-means and the other with GMM. Then we use these codebooks to train and test the selected machine learning methods (decision tree, support vector machine, and logistic regression). This process happened in the local machine. The execution time of each method is displayed in Table 5.2.

Methods	Input	N$_c$*lustering*	Time Execution
surf	7023 images		5 min
k-means	7023 images	10	1 h
k-means	7023 images	100	2 h and 38 min
GMM	7023 images	10	1 h
GMM	7023 images	100	3 h and 49 min
Feature encoding	7023 images		12 h

Table 5.2: Table shows time execution of methods

Decision tree

The decision tree parameters that have been used are the default Scikit-learn parameters for the decision tree classifier which are:

1. **Max depth:** Refers to the maximum level that the tree can reach. In this case, it holds the value of None which leads to purifying the leaves or reaching the minimum sample in each leaf.

2. **Min sample split:** The minimum sample that each leaf can hold to perform the split. In this case, it is 2.

3. **Criterion :** In this case, it is gini which refers to the probability of misclassification of each node.

4. **Splitter :** The value that this parameter holds is best which means that the decision tree will evaluate every possibility according to the criterion and choose the best one to spilt from it.

5. **Min sample leaf :** Refers to the minimum of samples to generate a new node after the split which in this instance the value is 1 sample.

6. **Min weight fraction leaf :** This parameter is the sum of the weights of samples to create a leaf, this is used when the samples are weighted for example $min_sample_leaf = 1$ in this case you will need a sample weighted 0.3 and another 0.7 sums that up to 1 which is the min value of make the leaf and will be made base on this two sample. In this work this parameter is None.

7. **Max features :** Represents the maximum number of features considered to make a split. It has the value of None.

8. **Random state :** It has None as value.

9. **Max leaf nodes :** It controls the number of nodes the tree will generate. Here it has None.

10. **Min impurity decrease :** This parameter works like a threshold to the impurity to make the split. In this case, it is 0.

11. **Class weight :**This parameter is used in case of an unbalanced distribution of samples which makes the model has a bias toward the class with higher samples.The value is None .

12. **Cost Complexity Pruning (CCP) alpha :** This parameter is used to reduce the size of the tree after it has been constructed. The value is set to 0

After building this model we fit the codebooks that have been made by both clustering methods.

1. **K-means :** The codebooks have been created with a number of clusters from 10 up to 100 clusters. We fed the feature vectors to the decision tree and each one of them took about 0.2 secs executing time. The results are resumed in Table 5.3.

Metrics of decision tree with K-means clustering

	Accuracy	Precision	Recall	F1-Score
10 n_clusters	0.766	0.758	0.751	0.752
20 n_clusters	0.768	0.76	0.752	0.753
30 n_clusters	0.767	0.754	0.748	0.749
40 n_clusters	0.759	0.747	0.742	0.743
50 n_clusters	0.723	0.712	0.706	0.706
60 n_clusters	0.737	0.722	0.717	0.716
70 n_clusters	0.731	0.718	0.713	0.713
80 n_clusters	0.744	0.73	0.726	0.726
90 n_clusters	0.739	0.724	0.721	0.721
100 n_clusters	0.739	0.724	0.721	0.721

Table 5.3: Table shows the results of decision tree methods

As we can see the metrics of the decision trees were unsteady and the metrics had a positive relationship between them as they went up when the number of clusters was 20 where the accuracy was maximum with $acc \approx 0.768$ and then decreased steadily reaching minimum at number of clusters 50 with accuracy $acc \simeq 0.723$ after that it increased and decreased until number of clusters was 100 it had accuracy $acc \approx 0.739$. The reason why the accuracy was at its peak was cause there were few features for the decision tree to split therefore the more they increase the more the performance of the model decreased and this is what it is called "curse dimensionality" one of the downsides of decision tree.
Here we present the confusion matrix of the best model and the sum of TN, FN, and FP together are represented in Figure 5.1.

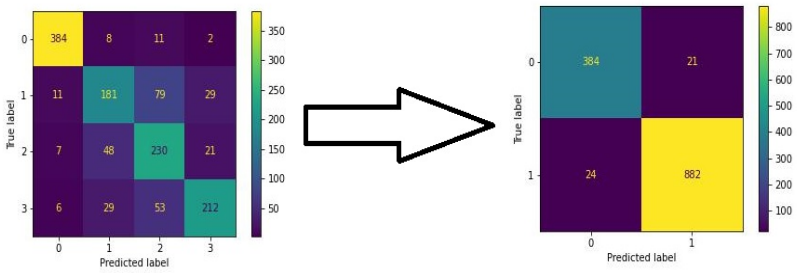


Figure 5.1: Image shows the confusion matrix of decision tree methods

Figure 5.2 below demonstrates the ROC and AUC of this model.

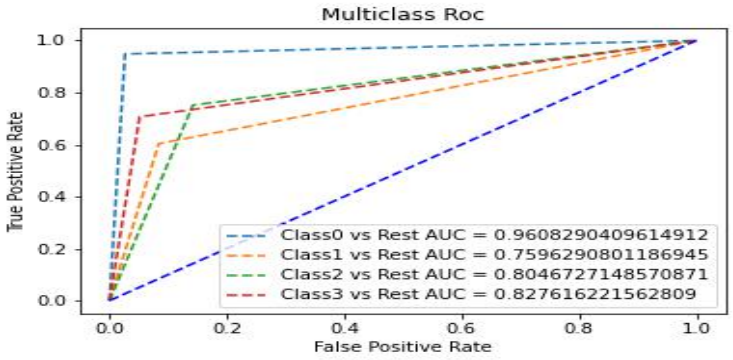


Figure 5.2: Graph shows the Receiver Operating Characteristic of decision tree methods multi-classes

2. **Gaussian Mixture model (GMM):** The codebooks were constructed 10 times using a number of clusters 10 to 100 and used to train the decision tree model. The results are shown in Table 5.4.

Metrics of decision tree with GMM clustering

	Accuracy	Precision	Recall	F1-Score
10 n_clusters	0.793	0.781	0.779	0.78
20 n_clusters	0.81	0.801	0.796	0.797
30 n_clusters	0.793	0.782	0.777	0.776
40 n_clusters	0.785	0.777	0.769	0.771
50 n_clusters	0.756	0.74	0.737	0.737
60 n_clusters	0.789	0.778	0.773	0.775
70 n_clusters	0.785	0.773	0.77	0.771
80 n_clusters	0.769	0.754	0.752	0.752
90 n_clusters	0.783	0.773	0.768	0.769
100 n_clusters	0.773	0.76	0.756	0.756

Table 5.4: Table shows the metrics of decision tree methods

The results of the decision tree were unstable, the metrics were changing in the same way as they increased when the number of clusters was 20 the accuracy was $acc \simeq 0.81$ and then it started wobbling in between to get an accuracy of $acc \simeq 0.756$ when the cluster number was 50 and $acc \simeq 0.773$ at the end.

The peak accuracy of the model can be attributed to the limited number of features available for the decision tree to divide. As the number of features increases, the performance of the model tends to decrease, which is commonly referred to as the "curse of dimensionality" a drawback specific to decision trees. The confusion matrix for the best model, and we convert the confusion matrix into a binary matrix, where 0 represents a healthy and 1 represents a sick brain. This conversion is done by combining the true negatives (TN), false negatives (FN), and false positives (FP) values. By doing so, we gain a more detailed understanding of the data, and the resulting binary matrix and they are depicted in Figure 5.3.

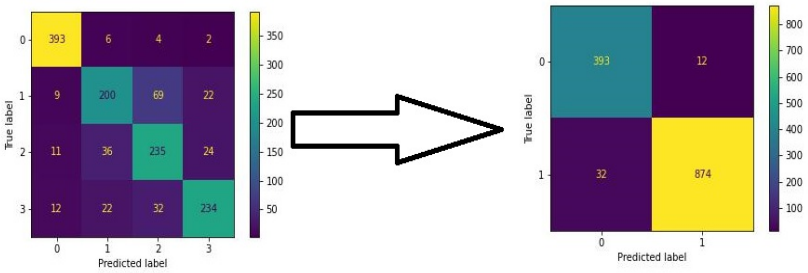


Figure 5.3: Figure shows the metrics of decision tree methods

Figure 5.4 below shows the ROC and AUC of this model.

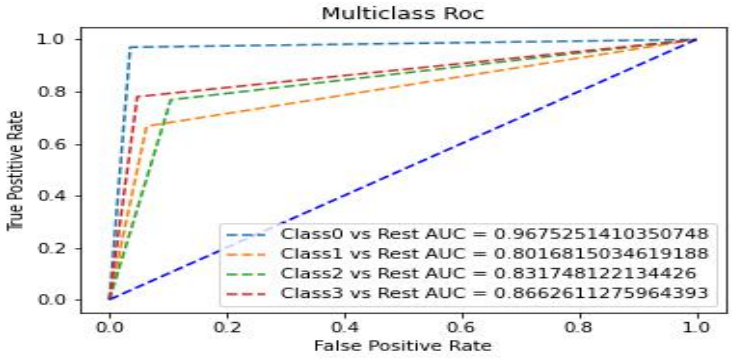


Figure 5.4: Graph shows the Receiver Operating Characteristic of decision tree methods multi-classes

With those models, we can conclude that the model based on feature vectors created with GMM with 20 clusters had better accuracy than the one based on feature vectors made with k-means by the same number of clusters by a small difference estimated by 0.042, as for the other metrics We try to increase FN of this model and to do this we have to decrease FP and the reason for that is that $FNR = 1 - FPR$, so the smaller is FP the bigger FN so the model train with K-means feature vectors was better in this department therefor the model with K-means feature vectors, is better than the based on GMM feature vectors.

Logistic regression

The logistic regression parameters that have been used are the default Scikit-learn default parameters for logistic regression with a few changes which are :

1. **Penalty :** Sklearn uses the ridge penalty for logistic regression L2 penalty.

2. **Dual :** This refers to the algorithm that optimizes logistic regression in this case it has the value of False so the algorithm is used in the primal formulation.

3. **Tol :** This parameter control the stopping condition if the optimizer value was below 10^{-4} the process will stop par default.

4. **Max iteration :** controls how many iterations should happen before the process stops. In this case, the default number was 100 but it was too small the model did not converge in 100 iterations. Therefore we increased it to 1000.

5. **C :** It has the value of 1 par default and it controls the amount of regularization used in the model, it works the same way as a soft classification.

6. **Fit intercept :** It is set to True in order to fit the intercept as well (bias).

7. **Intercept scaling :** This parameter is used to scale the intercept calculated from the fit intercept parameter, it helps when the features are not scaled. The default value is 1 which means there is no scaling.

8. **Class weight :** This parameter is used in case of imbalanced samples which makes the model have a bias to the class with higher samples. The value is None.

9. **Random state :** It has None as value.

10. **Multi class :** This parameter manages how the algorithm should treat the data in case of multi-class classification. We have selected the parameter to one vs rest (OVR).

After building this model we train it based on the codebooks that have been generated with both clustering method as follow:

1. **K-Means :** The codebooks were generated using various numbers of clusters ranging from 10 to 100. These codebooks were then used as inputs to logistic regression. Each logistic regression model took approximately 0.2 seconds to execute. The outcomes of these experiments are summarized in Table 5.5.

Metrics of logistic regression with K-means clustering

	Accuracy	Precision	Recall	F1-Score
10 n_clusters	0.624	0.599	0.613	0.601
20 n_clusters	0.691	0.662	0.679	0.665
30 n_clusters	0.71	0.685	0.699	0.686
40 n_clusters	0.727	0.703	0.715	0.703
50 n_clusters	0.734	0.713	0.722	0.714
60 n_clusters	0.745	0.728	0.734	0.728
70 n_clusters	0.751	0.733	0.74	0.732
80 n_clusters	0.767	0.75	0.755	0.748
90 n_clusters	0.778	0.763	0.768	0.76
100 n_clusters	0.776	0.759	0.764	0.759

Table 5.5: Table shows the results of Logistic regression methods

The outcomes of the Logistic regression model were found to be consistent. The evaluation metrics displayed stability as the number of clusters increased. For instance, when there were 90 clusters, the accuracy was approximately 0.778, and then it dropped down to 0.776 in the number of clusters 100.
Logistic regression seemed to do better as the number of features increased, thus it curved when the number of features increased more than it can handle and the reason for that was that the complexity of the model was not enough for that many features, Figure below 5.5 displays the confusion matrix of this model and The total sum of true negatives (TN), false negatives (FN), and false positives (FP).

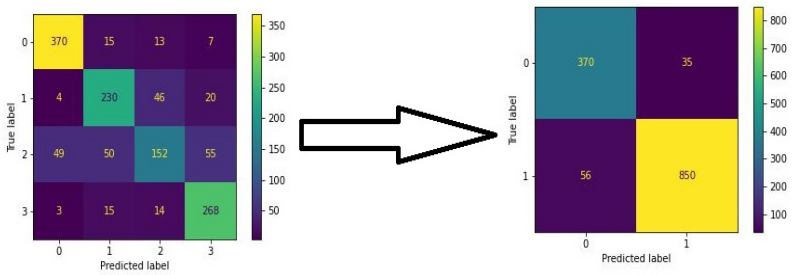


Figure 5.5: Figure shows the Confusion matrix of Logistic regression methods

and Figure 5.6 display the ROC and AUC of this model.

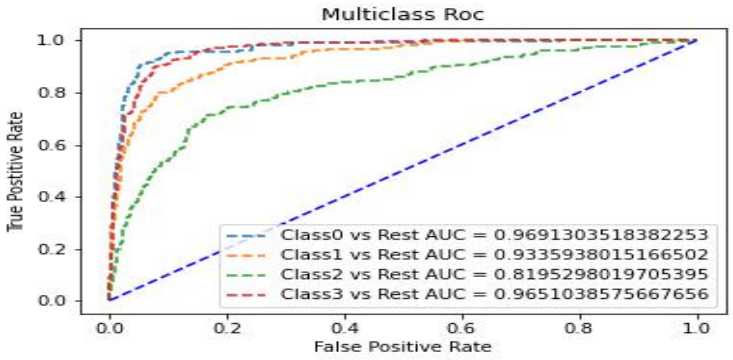


Figure 5.6: Graph shows the Receiver Operating Characteristic of Logistic regression methods multi-class

2. **GMM :** The codebooks were generated using several numbers of clusters ranging from 10 to 100. These codebooks were then used as inputs to logistic regression. Each logistic regression model took approximately 0.1 seconds to execute. The outcomes of these experiments are summarized in Table 5.6.

Metrics of logistic regression with GMM clustering

	Accuracy	Precision	Recall	F1-Score
10 n_clusters	0.674	0.655	0.663	0.655
20 n_clusters	0.704	0.691	0.694	0.69
30 n_clusters	0.734	0.717	0.723	0.718
40 n_clusters	0.735	0.717	0.725	0.719
50 n_clusters	0.76	0.745	0.75	0.746
60 n_clusters	0.775	0.759	0.765	0.76
70 n_clusters	0.793	0.779	0.783	0.78
80 n_clusters	0.783	0.769	0.772	0.769
90 n_clusters	0.795	0.782	0.785	0.783
100 n_clusters	0.798	0.783	0.788	0.784

Table 5.6: Table shows the results of Logistic regression methods

The results of the Logistic regression model demonstrated stability and consistency. As the number of clusters increased, the evaluation metrics remained relatively stable. It peaked at the final cluster number which is 100 with an accuracy of 0.797.
As the number of features increased, logistic regression initially demonstrated improved performance, and as we can see the effectiveness of the model did not decline at any point. Figure 5.7 presents the confusion matrix of this logistic regression model, providing a visual representation of its performance. To analyze the combined data of the sick class labeled as 1 and the healthy class labeled as 0.

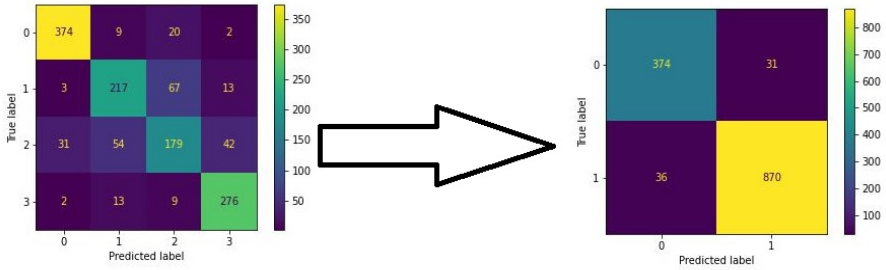


Figure 5.7: Figure shows the Confusion matrix of Logistic regression methods

And Figure 5.8 to visualize the ROC and AUC of this model.

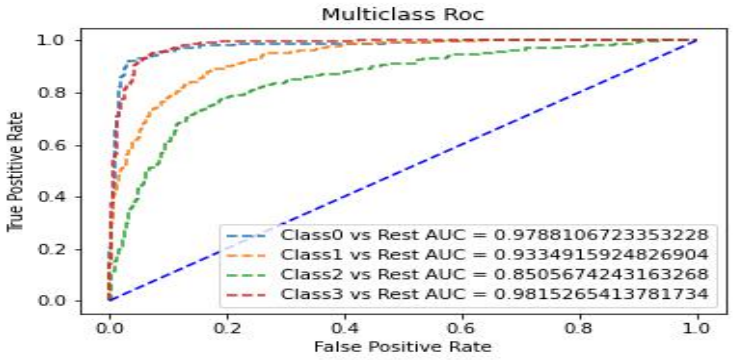


Figure 5.8: Graph shows the Receiver Operating Characteristic of Logistic regression methods multi-class

Based on the analysis of different models, we can conclude that the best model using feature vectors created with GMM with 100 clusters shows a slightly high accuracy compared to the best model using feature vectors generated with k-means clustering with the 90 number of clusters, with a different estimated to be approximately 0.019.
And even when focusing on the other metrics, specifically trying to increase the false negative (FN) rate, it was observed that decreasing the false positive (FP) rate is necessary. This is because the false negative rate (FNR) is the complement of the false positive rate (FPR), and decreasing FP would increase FN we found that the GMM model was better on that side as well. So the model based on GMM feature vectors was better overall.

Support vector machine

The subsequent model employed in our study is the Support Vector Machine (SVM). For SVM, we utilized the default parameters provided by the Scikit-learn library with the change of the kernel:

1. **Kernel :** We have selected the linear kernel.
2. **C :** Control The misclassification of the model. The default is 1 with the penalty of L2 (ridge).

3. **Shrinking :** This parameter is set to True par default so the model improves its computation efficiency.

4. **Probability :** Par default is set to False but we changed it to True so we can generate the confusion matrix.

5. **Tol :** By default, this parameter controls the termination criterion of the optimizer in the logistic regression model. If the optimizer's value falls below 10^{-3}, the optimization process will stop.

6. **Class weight :** This parameter is utilized when dealing with imbalanced datasets, where the model may show a bias towards the class with a larger number of samples. The default value for this parameter is None, indicating that no specific weighting is applied to address the class imbalance.

7. **Max iter:** There is no max iteration.

8. **Decision function shape :** We have selected one vs rest (OVR).

9. **Random state :** Is set To None.

The next step is to feed the model with feature vectors made by both the clustering methods GMM and K-means.

1. **K-means :** A range of codebooks was created by generating clusters with varying numbers, ranging from 10 to 100. These codebooks were subsequently applied as inputs for SVM. Each SVM model took approximately 13s seconds to execute. The results obtained from these experiments are simply summarized in Table 5.7.

Metrics of SVM with K-means clustering

	Accuracy	Precision	Recall	F1-Score
10 n_clusters	0.638	0.619	0.627	0.622
20 n_clusters	0.719	0.703	0.706	0.704
30 n_clusters	0.739	0.722	0.727	0.723
40 n_clusters	0.748	0.731	0.735	0.731
50 n_clusters	0.757	0.744	0.745	0.744
60 n_clusters	0.761	0.749	0.75	0.749
70 n_clusters	0.767	0.753	0.755	0.753
80 n_clusters	0.788	0.775	0.774	0.774
90 n_clusters	0.791	0.777	0.779	0.778
100 n_clusters	0.8	0.787	0.787	0.785

Table 5.7: Table shows the results of SVM methods

The SVM model displayed consistent and stable results. As the number of clusters increased, the evaluation metrics remained relatively steady. The accuracy reached its highest point at the final cluster number, which was 100, with a value of 0.8.
With an increase in the number of features, the SVM model consistently showed improved performance without any decline. This indicates that the effectiveness of the model remained stable throughout. The performance of the SVM model is visually depicted in Figure 5.9 through a confusion matrix, providing a clear representation of its classification performance. And to further examine FN, TP, TN, and FP we merged them to sick class labeled as 1 and the healthy class labeled as 0.

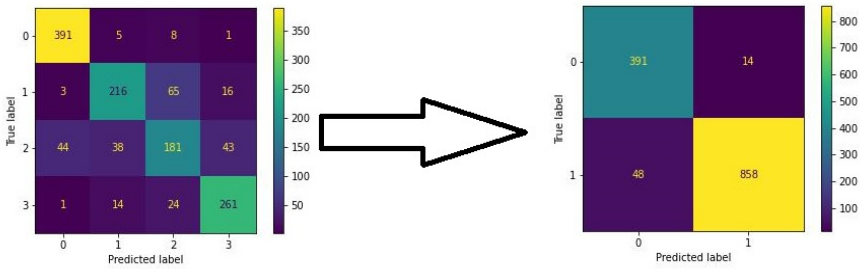


Figure 5.9: Figure shows the Confusion matrix of SVM methods

And Figure 5.10 is the ROC and AUC curve of this model.

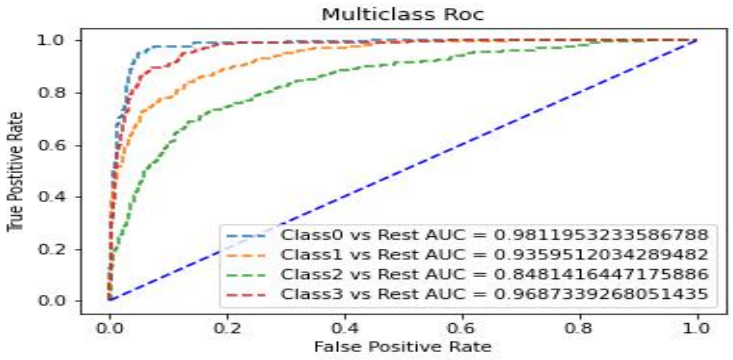


Figure 5.10: Graph shows the Receiver Operating Characteristic of SVM methods multi-class

2. **GMM :** Codebooks were created with varying numbers of clusters between 10 and 100. These codebooks were subsequently utilized as inputs for Support Vector Machines (SVM). Each SVM model required an average execution time of approximately 15 seconds. The findings from these experiments are consolidated and presented in Table 5.8.

Metrics of SVM with GMM clustering

	Accuracy	Precision	Recall	F1-Score
10 n_clusters	0.665	0.651	0.658	0.652
20 n_clusters	0.724	0.717	0.715	0.715
30 n_clusters	0.741	0.729	0.729	0.728
40 n_clusters	0.759	0.746	0.749	0.746
50 n_clusters	0.775	0.763	0.763	0.762
60 n_clusters	0.787	0.778	0.776	0.777
70 n_clusters	0.808	0.797	0.798	0.797
80 n_clusters	0.804	0.795	0.792	0.792
90 n_clusters	0.804	0.796	0.794	0.794
100 n_clusters	0.809	0.8	0.797	0.797

Table 5.8: Table shows the results of SVM methods

The performance of the SVM model exhibited stability and consistency. Regardless of the number of clusters, the evaluation metrics remained relatively constant.

The accuracy consistently improved as the number of clusters increased, reaching its peak of $acc \simeq 0.81$ when there were 100 clusters.
As the number of features increased, the SVM model consistently demonstrated improved performance without any decrease. This suggests that the model's effectiveness remained stable across different feature dimensions. The performance of the SVM model is visually illustrated in Figure 5.11 using a confusion matrix, which provides a clear illustration of its classification performance. This merged dataset allows for the examination and analysis of the data encompassing both classes And Figure 5.12 demonstrate the ROC and AUC of this model

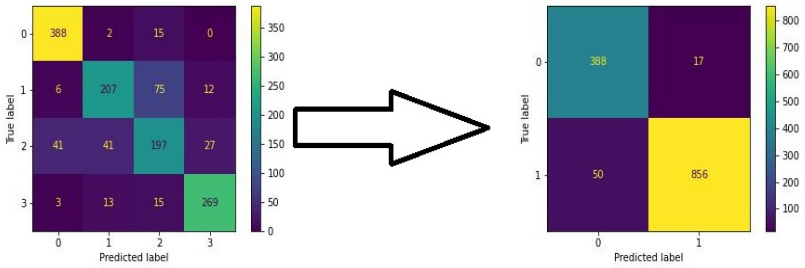


Figure 5.11: Figure shows the Confusion matrix of SVM methods

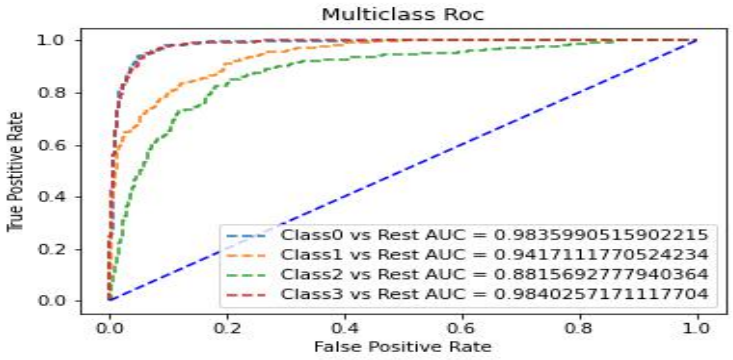


Figure 5.12: Graph shows the Receiver Operating Characteristic of SVM methods multi-class

Based on our analysis of different models, we have determined that the model using feature vectors created with Gaussian Mixture Model (GMM) using 100 clusters performs slightly better in terms of accuracy compared to the model using feature vectors generated with k-means clustering using 90 clusters. The difference in accuracy between the two models is estimated to be around 0.009.

When examining other metrics, specifically focusing on increasing the false negative (FN) rate, we observed that it is necessary to decrease the false positive (FP) rate. This is because the false negative rate (FNR) is the complement of the false positive rate (FPR), and reducing FP would lead to an increase in FN. In this regard, we found that the GMM model performed better, as it achieved a lower FP rate.

Considering all aspects, including accuracy and the trade-off between FN and FP rates, the model based on GMM feature vectors emerged as the superior choice.

After seeing these results we want to enhance our model performance so we took the codebooks of both GMM and K-means that was created by 10 number of cluster and fit them to PCA with the number of component of 3 this way we can find out the distribution of the data and visualize it. Figures 5.13, 5.14, and 5.15 are the distribution of points in codebooks of GMM, and the contribution of these three components in the data is 72.7%.

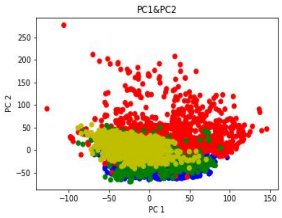


Figure 5.13: A Figure shows the distribution of points in PC1 and PC2

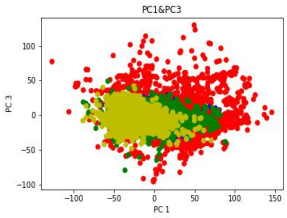


Figure 5.14: A Figure shows the distribution of points in PC1 and PC3

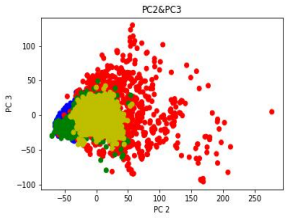


Figure 5.15: A Figure shows the distribution of points in PC2 and PC3

Figures 5.16, 5.17, and 5.18 are for the K-mean distribution with a variance ratio of 71.8%.

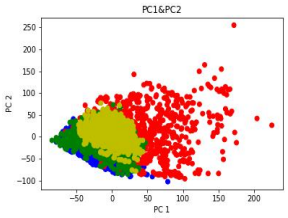


Figure 5.16: A Figure shows the distribution of points in PC1 and PC2

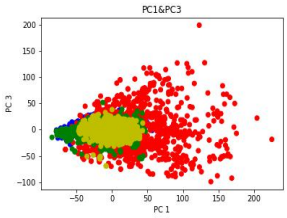


Figure 5.17: A Figure shows the distribution of points in PC1 and PC3

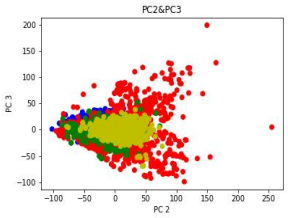


Figure 5.18: A Figure shows the distribution of points in PC2 and PC3

As we can see the data is separable with Gaussian kernel which is why we fit it again to SVM with Radial Basis Function (RBF) kernel with $\gamma = \frac{1}{nF * Var}$ where nF is the number of features, and this called the inverse of std and it scale with the data as default from sklearn library.

1. **K-means :**A series of codebooks were generated by clustering with different numbers, varying from 10 to 100. These codebooks were then used as inputs for Support Vector Machine (SVM) models. Each SVM model took around 21 seconds to execute. The outcomes of these experiments are summarized in Table 5.9.

Metrics of SVM with K-means clustering

	Accuracy	Precision	Recall	F1-Score
10 n_clusters	0.726	0.708	0.712	0.708
20 n_clusters	0.809	0.795	0.796	0.794
30 n_clusters	0.822	0.808	0.809	0.807
40 n_clusters	0.83	0.818	0.818	0.817
50 n_clusters	0.845	0.835	0.834	0.833
60 n_clusters	0.844	0.835	0.833	0.833
70 n_clusters	0.846	0.836	0.835	0.835
80 n_clusters	0.85	0.843	0.839	0.84
90 n_clusters	0.86	0.851	0.85	0.85
100 n_clusters	0.855	0.847	0.845	0.844

Table 5.9: Table shows the results of SVM methods

According to the results of the SVM model using the RBF kernel were consistently reliable. The assessment measures showed consistent patterns as the number of clusters increased. Specifically, with 90 clusters, the accuracy was around 0.86, and it slightly decreased to 0.855 when there were 100 clusters.
The performance of Support Vector Machines (SVM) improved as the number of features increased. However, there was a point where the model's performance started to decline, creating a curved pattern. This decline occurred because the complexity of the model was insufficient to handle such a large number of features. We present Figure 5.19, which illustrates the confusion matrix of this particular SVM model and an illustration of the merged dataset, which comprises data from both the sick class (labeled as 1) and the healthy class (labeled as 0).

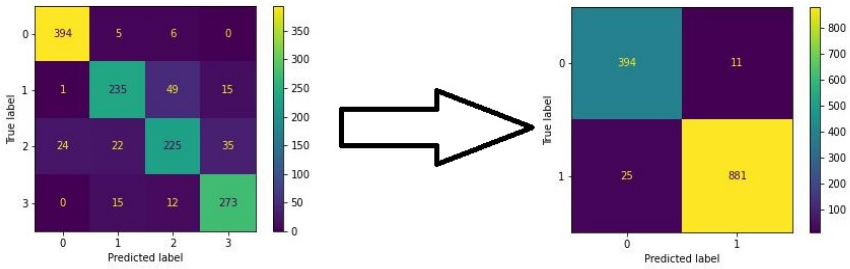


Figure 5.19: Figure shows the Confusion matrix of SVM methods

And Graph 5.20 illustrate the ROC and AUC of this model.

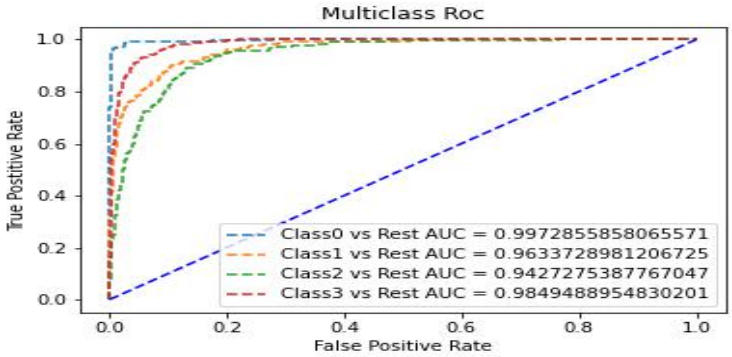


Figure 5.20: Graph shows the Receiver Operating Characteristic of SVM methods multi-class

2. **GMM :** Codebooks have been generated with different cluster numbers ranging from 10 to 100. These codebooks were then used as inputs for Support Vector Machines (SVM). On average, each SVM model took around 27 seconds to execute. The results of these experiments are summarized and shown in Table 5.10.

Metrics of SVM with GMM clustering

	Accuracy	Precision	Recall	F1-Score
10 n_clusters	0.756	0.743	0.743	0.74
20 n_clusters	0.818	0.814	0.808	0.808
30 n_clusters	0.823	0.814	0.813	0.812
40 n_clusters	0.842	0.832	0.831	0.83
50 n_clusters	0.842	0.832	0.833	0.831
60 n_clusters	0.861	0.855	0.851	0.851
70 n_clusters	0.858	0.849	0.848	0.848
80 n_clusters	0.867	0.864	0.858	0.858
90 n_clusters	0.854	0.846	0.844	0.843
100 n_clusters	0.863	0.857	0.854	0.853

Table 5.10: Table shows the results of SVM methods

The results obtained from the SVM model using the RBF kernel consistently demonstrated reliability. The evaluation metrics consistently displayed certain trends as the number of clusters increased. Specifically, when there were 80 clusters, the accuracy was approximately

0.867. The accuracy slightly decreased to 0.853 when the number of clusters increased to 90, but then rose again to 0.863 with 100 clusters. As the number of features increased, the performance of Support Vector Machines (SVM) showed improvement. However, at a certain point, the model's performance began to exhibit fluctuations, forming a curved pattern. This decline can be attributed to the model's limited capacity to handle a large number of features effectively. In Figure 5.21, we provide a visualization of the confusion matrix for this specific SVM model. and data from both the sick class (labeled as 1) and the healthy class (labeled as 0) have been merged in the other matrix.

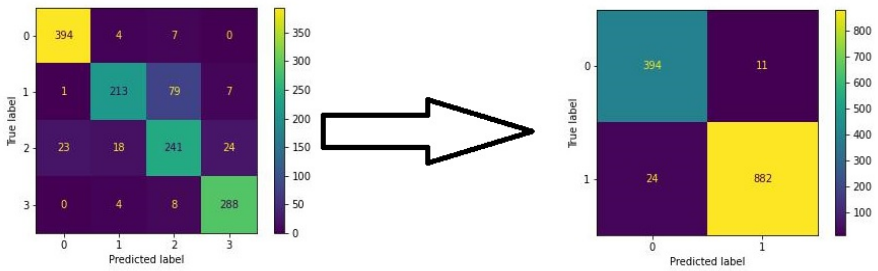


Figure 5.21: Figure shows the Confusion matrix of SVM methods

And The Figure 5.22 presents the ROC graph with AUC.

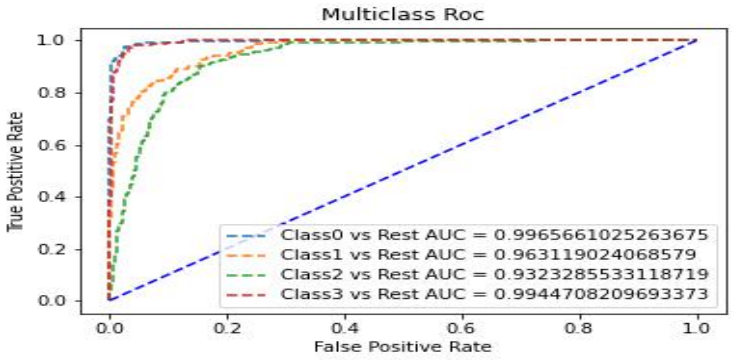


Figure 5.22: Graph shows the Receiver Operating Characteristic of SVM methods multi-class

After analyzing the new SVM models, it has been concluded that the model utilizing feature vectors created with Gaussian Mixture Model (GMM) and 80 clusters exhibits slightly better accuracy compared to the model using feature vectors generated with k-means clustering and 90 clusters. The difference in accuracy between the two models is estimated to be approximately 0.007. Further examination of other metrics, specifically with a focus on increasing the false negative (FN) rate, reveals the need to decrease the false positive (FP) rate. This is due to the fact that the false negative rate (FNR) is the complement of the false positive rate (FPR), and reducing FP would result in an increase in FN. In this regard, both models performed similarly in this area. Taking into consideration all factors, including accuracy and the trade-off between FN and FP rates, the model based on GMM feature vectors stands out as the superior choice. Depending on all these experiments and models we concluded that when the features has be created with the GMM cluster and fed to SVM using the RBF kernel we had the best results.

5.2.2 Deep learning discussions and results

Here, we will cover the findings and analyses of the models and deep learning techniques that we choose to deploy.
The pre-processing section for DL methods is the same as for ML methods only without the Gaussian filter. Next comes the training of 12 CNN models. As we said earlier, the models are trained in 100 epochs, and each epoch takes 7-8 seconds, that is, it takes about 13 minutes to train a single model. For transfer learning methods, each epoch takes at least 44 seconds, which is about 1:15 hours. This means that the training in the DL methods took 8 hours.

Convolution neural network (CNN)

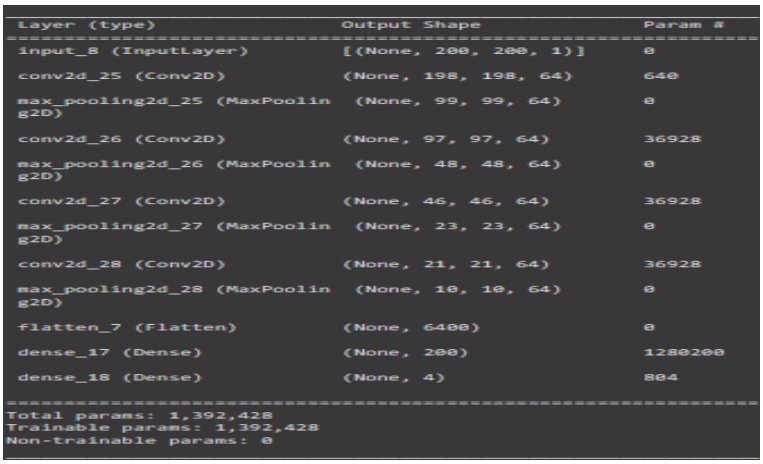
As we explained above, we used 12 CNN models. Let's now present the best model we have, which is the one with the highest accuracy validation or the lowest loss, so we will both present it. Table 5.11 illustrate the accuracy and loss of CNN models

CNN models	1st model	2nd model	3rd model	4th model	5th model	6th model	7th model	8th model	9th model	10th model	11th model	12th model
Loss Val	0.644	0.5937	0.592	0.4856	0.4358	0.480	0.379	0.3858	0.448	0.371	0.269	0.4436
Accuracy	94.89	94.28	94.97	95.73	94.13	96.19	95.96	95.35	95.58	95.58	95.58	95.35

Table 5.11: The result of CNN models

The sixth model achieved the highest accuracy among the others that estimated at 96% with loss validation 0.42, indicating that it accurately classified a larger proportion of the samples. This suggests that the model performs well in terms of overall prediction correctness. On the other hand, the eleventh model achieved lower loss estimates of 0.2 with acc 94.6%, which implies that it has a better ability to minimize errors and discrepancies between the predicted and true values. We previously explained the strictures of these models, and now we will provide the Summaries of the two models:

Figure 2.23 the architecture of the sixth model.

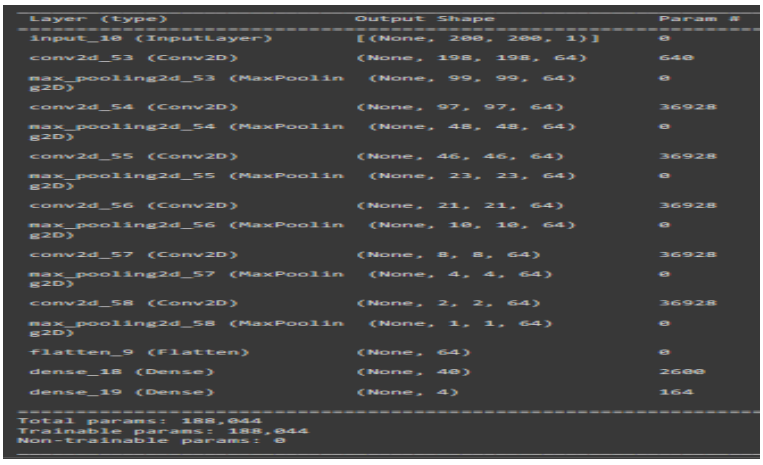


Layer (type)	Output Shape	Param #
input_8 (InputLayer)	[(None, 200, 200, 1)]	0
conv2d_25 (Conv2D)	(None, 198, 198, 64)	640
max_pooling2d_25 (MaxPooling2D)	(None, 99, 99, 64)	0
conv2d_26 (Conv2D)	(None, 97, 97, 64)	36928
max_pooling2d_26 (MaxPooling2D)	(None, 48, 48, 64)	0
conv2d_27 (Conv2D)	(None, 46, 46, 64)	36928
max_pooling2d_27 (MaxPooling2D)	(None, 23, 23, 64)	0
conv2d_28 (Conv2D)	(None, 21, 21, 64)	36928
max_pooling2d_28 (MaxPooling2D)	(None, 10, 10, 64)	0
flatten_7 (Flatten)	(None, 6400)	0
dense_17 (Dense)	(None, 200)	1280200
dense_18 (Dense)	(None, 4)	804

Total params: 1,392,428
Trainable params: 1,392,428
Non-trainable params: 0

Figure 5.23: the summary of the CNN architecture of the sixth model

As we can see, we used 4 successive convolution and pooling layers, one after one, and after flattening it, we noticed that most of the data carried 0, so we added a dense layer with 200 neural networks to reduce that, then an output layer. As a result, our model was trained on 1,392,428 parameters.
Figure 2.24 the architecture of the eleventh model.



Layer (type)	Output Shape	Param #
input_10 (InputLayer)	[(None, 200, 200, 1)]	0
conv2d_53 (Conv2D)	(None, 198, 198, 64)	640
max_pooling2d_53 (MaxPooling2D)	(None, 99, 99, 64)	0
conv2d_54 (Conv2D)	(None, 97, 97, 64)	36928
max_pooling2d_54 (MaxPooling2D)	(None, 48, 48, 64)	0
conv2d_55 (Conv2D)	(None, 46, 46, 64)	36928
max_pooling2d_55 (MaxPooling2D)	(None, 23, 23, 64)	0
conv2d_56 (Conv2D)	(None, 21, 21, 64)	36928
max_pooling2d_56 (MaxPooling2D)	(None, 10, 10, 64)	0
conv2d_57 (Conv2D)	(None, 8, 8, 64)	36928
max_pooling2d_57 (MaxPooling2D)	(None, 4, 4, 64)	0
conv2d_58 (Conv2D)	(None, 2, 2, 64)	36928
max_pooling2d_58 (MaxPooling2D)	(None, 1, 1, 64)	0
flatten_9 (Flatten)	(None, 64)	0
dense_18 (Dense)	(None, 40)	2600
dense_19 (Dense)	(None, 4)	164

Total params: 188,044
Trainable params: 188,044
Non-trainable params: 0

Figure 5.24: the summary of the CNN architecture of the eleventh model

In this model, we used 6 convolution and pooling layers to delve deeper and try to reduce the error. After flattening it, we added a dense layer with 40 neural networks, and then the output layer. Thus, our model was trained on 188,044 parameters
To gain a deeper understanding of the models' performance, additional evaluation metrics were considered. Precision, recall, and F1-score were examined to assess the models' predictive accuracy, ability to capture positive instances, and balance between precision and recall. Table 5.12 shows the metrics of the metrics of the 6Th and 11Th models.

CNN models	Accuracy	Precision	Recall	F1-score
6Th Model	0.961	0.960	0.958	0.959
11Th Model	0.955	0.953	0.952	0.952

Table 5.12: Table shows the metric of the best CNN models

To gain further insight into the models' performance, the confusion matrix was examined. The confusion matrix reveals the distribution of true positive, true negative, false positive, and false negative predictions. Figures 5.25 and 5.26 are the confusion matrix of the 6Th CNN model and 11Th CNN model respectively.

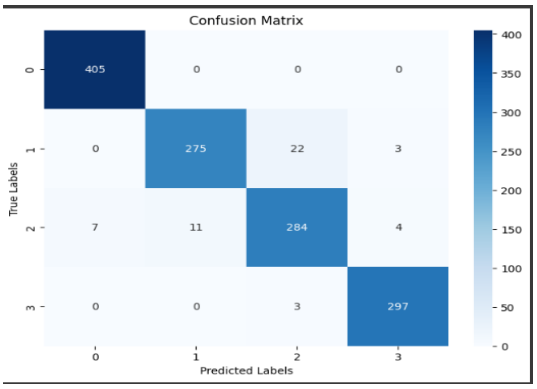


Figure 5.25: the Confusion matrix of the model with higher accuracy

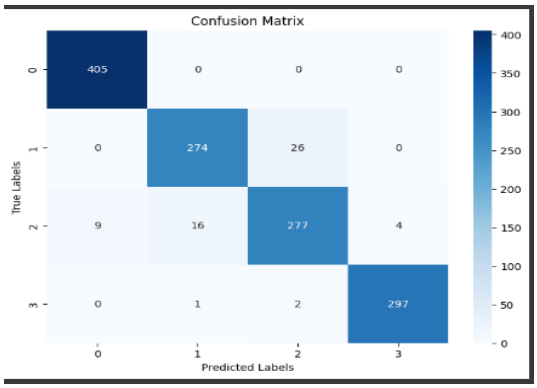


Figure 5.26: the Confusion matrix of the model with lower loss

The two models achieved high accuracy and correctly classified the majority

of instances across all classes, indicating strong overall performance. Where, Class 0 (representing the absence of a tumor) was accurately identified in all instances, demonstrating the ability of the two to correctly detect non-tumor cases. However, misclassifications were observed in distinguishing between different types of tumors (classes 1, 2, and 3 that represent glioma, meningioma, and pituitary tumors). Notably, class 2 had instances misclassified as class 0 and class 1, which is a significant issue as it may result in false negatives and misdiagnosis. Despite the model's success in identifying non-tumor cases, it still has to be improved in order to distinguish between glioma and meningioma classes. To increase the model's diagnostic precision and guarantee reliable tumor detection, the misclassification problem for class 2 must be resolved. In conclusion, while the model with higher accuracy often showed fewer misclassifications, both models had similar patterns in terms of confusion matrix outcomes.

To evaluate the models' discriminative power across various decision thresholds, the ROC curves were also examined. Figures 5.27 and 5.26 demonstrate the ROC curve for 6Th model and 11Th model respectively.

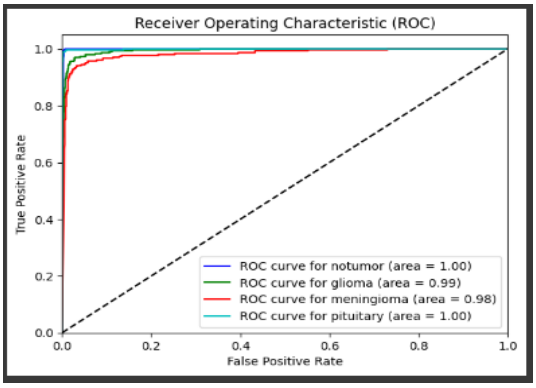


Figure 5.27: the Roc curve of the model with higher accuracy

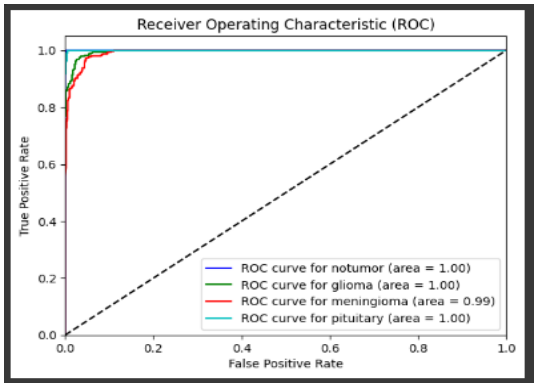


Figure 5.28: the Roc curve of the model with lower loss

Transfer learning

In this section, we will introduce the best model we have obtained in the transfer learning method.
Since we used ResNet50 as an input to the model, we will remove the decision layer that contains 1,000 categories, so the input will contain 24,877,616 features, followed by a flattening layer, then a dense layer with 500 neurons, then a final dense layer as an output layer with 4 neural networks representing our classes Notumor, glioma, meningioma, and pituitary tumor. We will provide a summary of the model, but let's first give the ResNet50 parameters that we used :

1. **Include-top:** That chooses whether or not to incorporate the network's fully linked upper levels. our choice is $include_top = False$ because we need to leverage the pre-trained ResNet-50 model as a feature extractor, not as a classifier.

2. **Weights:** The weights parameter specifies whether pre-trained weights or random weights should be used to establish the model. like we said earlier, we initialize with pre-trained weights i.e. put weights='imagenet'.

3. **Input-shape:** Only needed if *include_top* is set to False. it refers to the shape or dimensions of the input data that the model expects.

4. **Pooling:** It specifies the type of pooling operation to be used in the last pooling layer of the network. there are 'avr', 'max', and None. we used the max pooling.

Figure 5.29 shows the summary of the pre-trained model.

```
Model: "model_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_11 (InputLayer)       [(None, 224, 224, 3)]     0

 resnet50 (Functional)       (None, 7, 7, 2048)        23587712

 flatten_5 (Flatten)         (None, 100352)            0

 dense_9 (Dense)             (None, 400)               40141200

 dense_10 (Dense)            (None, 4)                 1604

=================================================================
Total params: 63,730,516
Trainable params: 40,142,804
Non-trainable params: 23,587,712
_________________________________________________________________
```

Figure 5.29: the summary of the fine-tuned model

Our model was trained on 40,142,804 parameters and gave an accuracy of 97.45 and an error of 0.2. Figure 5.30 shows a graph of the change in loss and accuracy.

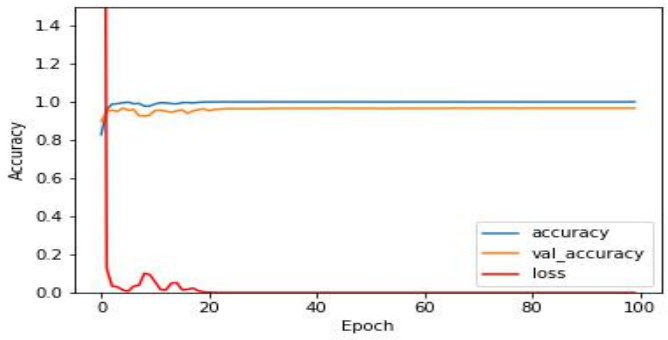


Figure 5.30: Curve showing the change in accuracy, validation acc, and loss per epoch in the training model

We see a rapid decline in the loss in the seventh epoch, then it begins to change and oscillate slightly until it stabilizes at 20. Until we see the same

thing for the two minutes, you notice stability at the 20Th epoch as well. So, we think that 20 epochs were enough to train the model.
Figure 5.31 shows the confusion matrix of this model.

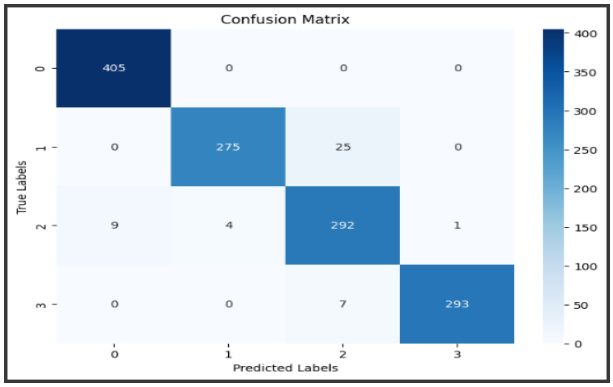


Figure 5.31: Confusion matrix of model

As we said earlier, there is a problem with the classification of classes 1 and 2, which means that the model was not sufficient to separate the two types of brain tumors, glioma, and meningioma.
Figure 3.32 illustrate the ROC curve of the model.

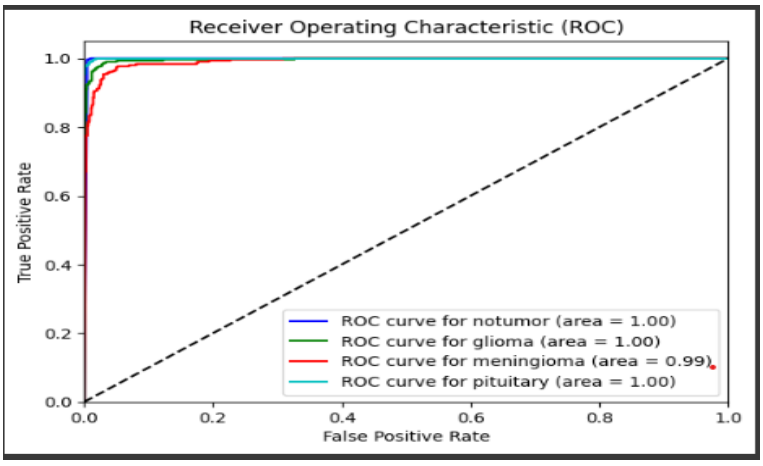


Figure 5.32: Graphe showing the Roc curve of fine-tuned ResNet50 model

In conclusion, MRI brain tumor images were classified into four classes notumor, glioma, meningioma, and pituitary tumors using the DL techniques of CNN and transfer learning: tumors, and no tumor. The ResNet50 model used in the transfer learning strategy achieved a higher accuracy of 97.6% with a loss of 0.2 loss while the CNN model only managed to achieve 96.2% accuracy with the same loss. This shows that the accuracy of transfer learning with ResNet50 exceeded the standalone CNN model by a factor of 1.4%. These findings show how DL techniques can successfully categorize images of brain tumors, with the transfer learning method outperforming other approaches by using characteristics from the trained ResNet50 model.

Overall, after many experiments we can conclude that deep learning had the upper hand in classifying brain tumors over machine learning techniques in accuracy and in time consumption. Table 5.13 illustrates the accuracy and time consumed to train the best models.

Model	Accuracy	Time
SVM	86.6%	17h
ResNet50	97.6%	1h

Table 5.13: Accuracy and Time consumption to train of each best model

Chapter 6

Conclusion

For ML methods, feature extraction was performed using the Surf algorithm, followed by clustering the features using k-means and Gaussian Mixture Models (GMM). A bag of feature representation was then used, and the data were classified using Support Vector Machines (SVM), Decision Trees (DT), and Linear Regression (LR). Among the ML methods, SVM, and DT showed better performance than LR, achieving an accuracy of 81% with GMM clustering. To further improve the performance, an RBF kernel was incorporated into SVM, which proved suitable for capturing the complex distribution of features exhibiting loop-like patterns, resulting in the highest accuracy of 86.6% among the ML methods.

Few options could improve the accuracy furthermore. Using PCA after extracting the features with SURF. Although it extracts features of the tumor but also the brain features which might be the same in all classes, removing them will decrease the time to train also. Also, change the σ of the RBF kernel and use algorithms to find the best K for clustering methods.

DL methods were also employed, beginning with CNN architectures for classification. Additionally, transfer learning using the ResNet50 model as input was utilized. Transfer learning achieved the best performance, with an accuracy of 97.6% and a loss of 0.2, while the CNN architecture achieved 96.2% accuracy with the same loss.

Several suggestions might be taken into consideration to enhance DL models. Investigating data augmentation methods is one recommendation. By using different transformations, such as rotation, scaling, or flipping, on the given data, data augmentation includes creating extra training examples.

This may assist increase and improving the training dataset, thus enhancing the performance of the DL model.
Another approach to consider is leveraging autoencoders. Autoencoders are unsupervised learning models that can learn a compressed representation of the input data. By training an autoencoder on the brain tumor MRI images, meaningful features can be extracted, which can then be used as input to the DL models. This additional step of feature extraction with autoencoders may help capture more relevant information and improve the classification performance.
Furthermore, a hybrid approach combining the features extracted from the pre-trained ResNet50 model and the SURF features could be explored. This could involve concatenating or combining the features and feeding them into a CNN or SVM classifier. By combining both high-level learned features from ResNet50 and handcrafted SURF features, the model may be able to capture a wider range of discriminative information and potentially enhance classification accuracy.

In summary, the study successfully employed both ML and DL methods for brain tumor MRI image classification. Transfer learning showed superior performance among the DL methods, while SVM with an RBF kernel demonstrated the best accuracy among the ML methods. The proposed modifications, such as data augmentation, autoencoder, or hybrid DL methods, provide potential avenues for future research and improvement of the existing model.

References

[1] Lu Wang et al. "Advances in research on the effects of natural drugs with immune-promoting effects on immune function". In: *European Journal of Inflammation* 18 (April 23, 2020), p. 15. DOI: https://journals.sagepub.com/doi/full/10.1177/2058739220926878.

[2] Shamaila Kausar et al. "A review: Mechanism of action of antiviral drugs". In: *Immunopathology and Pharmacology* 35 (March 16, 2021), p. 12. DOI: https://doi.org/10.1177/20587384211002621.

[3] Christine Rohde, Johannes Wittmann, and Elizabeth Kutter. "Bacteriophages: A Therapy Concept against Multi-Drug–Resistant Bacteria". In: *SURGICAL INFECTIONS* 19.8 (Dec 5, 2018), p. 8. DOI: 10.1089/sur.2018.184.

[4] Seyed hossein hassanpour and Mohammadamin dehghani. "Review of cancer from perspective of molecular". In: *Cancer Research and Practice* 4 (December 2017), p. 9. DOI: 10.1016/j.jcrpr.2017.07.001.

[5] Fatma E. AlTahhan et al. "Refined Automatic Brain Tumor Classification Using Hybrid Convolutional Neural Networks for MRI Scans". In: *MDPI* 13 (23 February 2023), p. 16. DOI: https://doi.org/10.3390/diagnostics13050864.

[6] Ahmed KHARRAT and Mahmoud NEJI. "Feature selection based on hybrid optimization for magnetic resonance imaging brain tumor classification and segmentation". In: *Applied Medical Informatics* 40 (March 31, 2019), p. 15. DOI: https://www.researchgate.net/publication/332245380_Feature_selection_based_on_hybrid_optimization_for_magnetic_resonance_imaging_brain_tumor_classification_and_segmentation.

[7] Mohammed Rasool et al. “A Novel Approach for Classifying Brain Tumours Combining a SqueezeNet Model with SVM and Fine-Tuning”. In: *MDPI* 12 (29 December 2022), p. 18. DOI: https://doi.org/10.3390/electronics12010149.

[8] Prince Priya Malla, Sudhakar Sahu, and Ahmed I. Alutaibi. “Classification of Tumor in Brain MR Images Using Deep Convolutional Neural Network and Global Average Pooling”. In: *MDPI* 11 (23 February 2023), p. 17. DOI: https://doi.org/10.3390/pr11030679.

[9] Soheila Saeedi et al. “MRI-based brain tumor detection using convolutional deep learning methods and chosen machine learning techniques”. In: *BMC Medical Informatics and Decision Making* 23 (23 January 2023), p. 17. DOI: https://doi.org/10.1186/s12911-023-02114-6.

[10] Badiea Abdulkarem Mohammed et al. “Hybrid Techniques of Analyzing MRI Images for Early Diagnosis of Brain Tumours Based on Hybrid Features”. In: *MDPI* 11 (9 January 2023), p. 27. DOI: https://doi.org/10.3390/pr11010212.

[11] Mohammed Rasool et al. “A Hybrid Deep Learning Model for Brain Tumour Classification”. In: *MDPI* 24 (8 June 2022), p. 16. DOI: https://doi.org/10.3390/e24060799.

[12] Hanumantha Rao Kondamur and Venkata Nageswara Rao Padmanabhuni. “Brain Tumour Images Classification Using Support Vector Machine and Pre-Trained Convolutional Neural Networks”. In: (), p. 23. DOI: https://doi.org/10.21203/rs.3.rs-2668214/v1.

[13] Marzieh Ghahramani and Nabiollah Shiri. “Brain tumour detection in magnetic resonance imaging using Levenberg–Marquardt backpropagation neural network”. In: *IET Image Processing* (3September 2022), p. 16. DOI: https://doi.org/10.1049/ipr2.12619.

[14] Masoumeh Siar and Mohammad Teshnehlab. “A combination of feature extraction methods and deep learning for brain tumour classification”. In: *IET Image Processing* (20 September 20212), p. 26. DOI: DOI:10.1049/ipr2.12358.

[15] David G. Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *International Journal of Computer Vision* 90 (November 2004). DOI: https://doi.org/10.1023/B:VISI.0000029664.99615.94.

[16] Dr Darshana Mistry and Asim Banerjee. "Comparison of Feature Detection and Matching Approaches: SIFT and SURF". In: *Research Gate* (March 2017). DOI: https://www.researchgate.net/publication/314285930_Comparison_of_Feature_Detection_and_Matching_Approaches_SIFT_and_SURF/link/58bfda4292851c7b7275f20c/download.

[17] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. "Speeded-Up Robust Features (SURF)". In: *European Conference on Computer Vision (ECCV)* (15 December 2007). DOI: https://doi.org/10.1016/j.cviu.2007.09.014.

[18] Youguo Li and Haiyan Wu. "A Clustering Method Based on K-Means Algorithm". In: *ResearchGate* (December 2012). DOI: 10.1016/j.phpro.2012.03.206.

[19] Jiehao Zhang et al. "Maximum Gaussian Mixture Model for Classification". In: *ResearchGate* (December 2016). DOI: 10.1109/ITME.2016.0139.

[20] Theodoros Evgeniou and Massimiliano Pontil. "Support Vector Machines: Theory and Applications". In: *ResearchGate* (September 2001). DOI: 10.1007/3-540-44673-7_12.

[21] Lior Rokach and Oded Maimon. "Decision Trees". In: *ResearchGate* (January 2005). DOI: 10.1007/0-387-25465-X_9.

[22] Joanne Peng. "An Introduction to Logistic Regression Analysis and Reporting". In: *ResearchGate* 96 (September 2002). DOI: 10.1080/00220670209598786.

[23] "Brain Tumor MRI Dataset". In: (). DOI: https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset?select=Training.