BRAIN TUMOR RADIOGENOMIC CLASSIFICATION - AN ADAPTED SWIN TRANSFORMER APPROACH

BY

M. Satya Sai Teja (Enrolment No. 20/11/EC/011)

under the guidance of

Dr. Anupama Namburu, School of Engineering, JNU, Delhi

in the fulfillment of the requirements
for the award of the degree of

Bachelor of Technology

School of Engineering

Jawaharlal Nehru University, New Delhi

May 2024

JAWAHARLAL NEHRU UNIVERSITY
SCHOOL OF ENGINEERING

DECLARATION

I declare that the project entitled **"BRAIN TUMOR RADIOGENOMIC CLASSIFICATION - AN ADAPTED SWIN TRANSFORMER APPROACH"**, submitted by me in fulfillment of the requirements for the award of the **Bachelor of Technology** in **Computer Science Engineering** to the School of Engineering, Jawaharlal Nehru University, Delhi comprises only my original work, and due acknowledgement has been made in the text to all other materials used.

M. Satya Sai Teja

JAWAHARLAL NEHRU UNIVERSITY
SCHOOL OF ENGINEERING

CERTIFICATE

This is to certify that the project report entitled **“BRAIN TUMOR RADIOGENOMIC CLASSIFICATION - AN ADAPTED SWIN TRANSFORMER APPROACH”,** being submitted by **Mr. M. Satya Sai Teja** (Enrolment No. 20/11/EC/011), in fulfillment of the requirements for the award of the **Bachelor of Technology** in **Computer Science Engineering**, was carried out by him under my supervision.

In my opinion, this work fulfills all the requirements of an Engineering Degree in respective streams as per the regulations of the School of Engineering, Jawaharlal Nehru University, Delhi. This thesis does not contain any work, which has been previously submitted for the award of any other degree.

Dr. Anupama Namburu
(Supervisor)
Associate Professor
School of Engineering
Jawaharlal Nehru University, Delhi

ACKNOWLEDGMENT

I express my sincere gratitude to **Dr. Anupama Namburu**, Associate Professor at the School of Engineering, Jawaharlal Nehru University, Delhi, for her invaluable guidance and supervision during the completion of my project, **“BRAIN TUMOR RADIOGENOMIC CLASSIFICATION - AN ADAPTED SWIN TRANSFORMER APPROACH”,** her expertise and encouragement have significantly contributed to the quality of my work. I am thankful for her unwavering support and mentorship.

M. Satya Sai Teja

ABSTRACT

Among brain tumors, glioblastomas are the most common and aggressive, resulting in a very short life expectancy, particularly in their highest grade. A critical predictor of chemotherapy response and patient outcomes in glioblastoma is the presence of *O6-Methylguanine-DNA Methyltransferase (MGMT) promoter methylation,* a *DNA* repair enzyme, within the tumor tissue. However, current genetic analysis methods require invasive surgery and involve prolonged waiting periods for results, often necessitating further surgical interventions based on findings.

In response to this challenge, our study proposes an innovative non-invasive approach that harnesses the power of radiogenomics, machine learning, and computer vision techniques. We primarily utilize the *Vision Transformer (ViT)* and its variant, *Swin Transformer (Swin-T)* model, to predict the genetic subtypes of glioblastoma from *Magnetic resonance imaging (MRI) scans*. These models were chosen for their demonstrated efficacy in image classification tasks, particularly in the field of medical imaging. Their ability to process visual information in a hierarchical manner, similar to how the human visual system operates, makes them promising choices for analyzing *MRI* scans to predict the genetic subtypes of glioblastoma.

Initially, we chose a basic *Vision Transformer (ViT)* architecture for the classification task. However, as our research progressed, we transitioned to the *Swin Transformer (Shifted Window Transformer/ Swin-T)* model due to its enhanced performance in capturing long-range dependencies within images. Given the complex and intricate nature of *MRI* scans, which contain vital information across various spatial scales, the *Swin Transformer's* capability to efficiently capture such dependencies proved advantageous for our predictive tasks. Furthermore, we made slight modifications to some of the layers of the *Swin-T* architecture to tailor it specifically for our classification task, resulting in an *Adapted Swin-T* model.

This transition and modification underscore our commitment to leveraging cutting edge technologies to achieve the highest levels of accuracy and reliability in predicting genetic subtypes of *glioblastoma*, thereby advancing the field of radiogenomics and improving patient care. The overarching goal of our approach is to revolutionize treatment decision-making processes by providing timely and accurate predictions. By obviating the need for invasive procedures and reducing the waiting time for results, our methodology aims to minimize the necessity for additional surgeries and, ultimately, enhance patient outcomes. Through this interdisciplinary fusion of medical imaging, artificial intelligence, and genetic analysis, we aspire to pave the way for more efficient and effective management strategies for glioblastoma patients.

LIST OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 BACKGROUND AND MOTIVATION

Radiogenomics, an emerging field at the intersection of *radiology* and *genomics*, holds immense promise in transforming tumor analysis and treatment planning [1]. It aims to decode the relationship between medical imaging and genetic changes in tumors, offering a less invasive pathway to unraveling tumor biology.

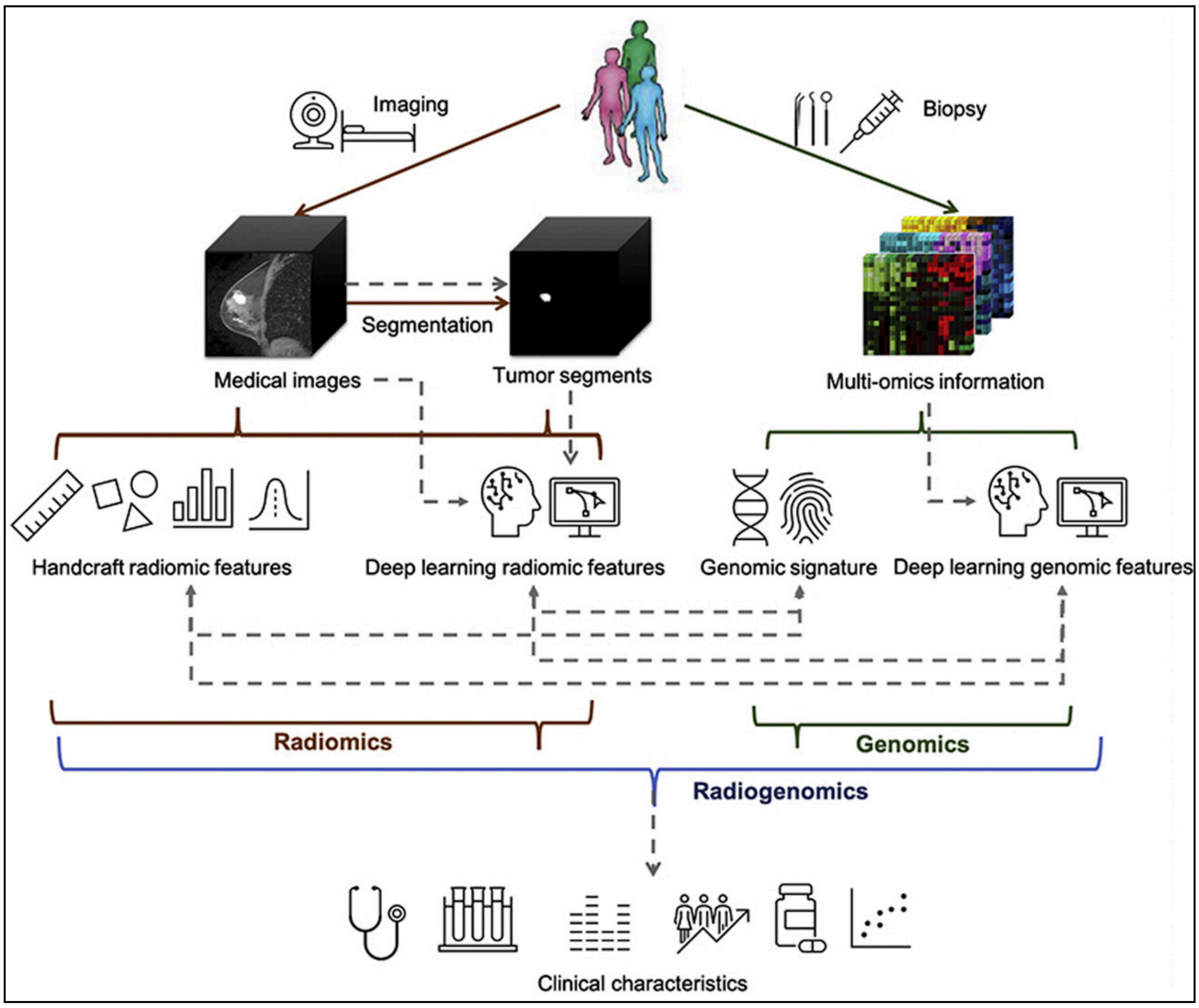


Fig 1.1.1 Graphical abstract of *Radiogenomics* [2]

By analyzing imaging data alongside genetic information, *radiogenomics* enables us to glean valuable insights into tumor behavior and treatment response [2], potentially sparing patients from invasive procedures like biopsies or surgeries. *Figure 1.1.1* illustrates a *radiogenomics* workflow, contrasting two feature extraction methods: handcrafted *radiomic* features, which are defined by researchers based on image properties, and *Deep Learning (DL) radiomic features*, which are automatically extracted by *Machine Learning (ML)* algorithms. The process begins with acquiring medical images *(X-ray, CT, MRI)* and identifying relevant regions within those images through segmentation techniques [3]. These tumor segments can then be analyzed to extract quantitative features. Concurrently, *multi-omics* information, including *genomic* signatures, is obtained, either through biopsy samples or non-invasive methods like liquid biopsies. Both the *radiomic* features and *genomic* features are then integrated and analyzed together to uncover insights into tumor biology, behavior, and potential treatment responses.

The *radiogenomics* approach leverages the complementary nature of imaging and *genomic* data, allowing for a comprehensive understanding of the tumor's characteristics. By combining *radiomics* and *genomics*, this field aims to develop more accurate predictive models and personalized treatment strategies, ultimately improving patient outcomes and reducing the need for invasive procedures.

Central to the advancements in *radiogenomics* are the fields of *computer vision, machine learning, and deep learning.* These techniques play a crucial role in extracting relevant features and patterns from medical images and *genomic* data, facilitating the discovery of meaningful correlations and predictive models.

Computer vision, a subfield of *artificial intelligence (AI)*, focuses on enabling computers to interpret and process visual information. In medical imaging, computer vision techniques are crucial for analyzing images, detecting abnormalities, segmenting anatomical structures, and classifying different types of tissues. This automated image analysis can significantly enhance diagnostic precision and treatment planning. *Figure 1.1.1* illustrates the radiogenomic association analysis process, which is central to the radiogenomics approach. It depicts the flow from medical imaging techniques like computed tomography, magnetic resonance imaging (MRI), and spectroscopy to the extraction of quantitative (radiomic) and qualitative imaging features. These imaging features are then associated with genetic information, such as gene methylation, mutation, and differential gene expression, through radiogenomic association analysis. The resulting imaging biomarkers can provide valuable insights into genomics, enabling a more comprehensive understanding of tumor behavior and treatment response.

Machine learning (ML), another subset of *AI*, entails the development of algorithms enabling computers to learn from data and make predictions. In medical imaging, *ML* algorithms are trained on extensive datasets of annotated images to recognize patterns and predict outcomes. This ability to learn from data is particularly useful in *radiogenomics*, where complex patterns in imaging data can be correlated with genetic mutations to support decision-making in clinical practice.

Deep learning, a specialized branch of *Machine Learning*, employs *neural networks* with numerous layers to model complex data patterns. *Convolutional Neural Networks (CNNs)*, a type of *deep learning* model, excel in image analysis due to their ability to automatically learn hierarchical features from raw pixel data. In *radiogenomics,* the application of *deep learning* has transformed medical imaging, achieving remarkable accuracy in tasks such as image classification, object detection, and segmentation.

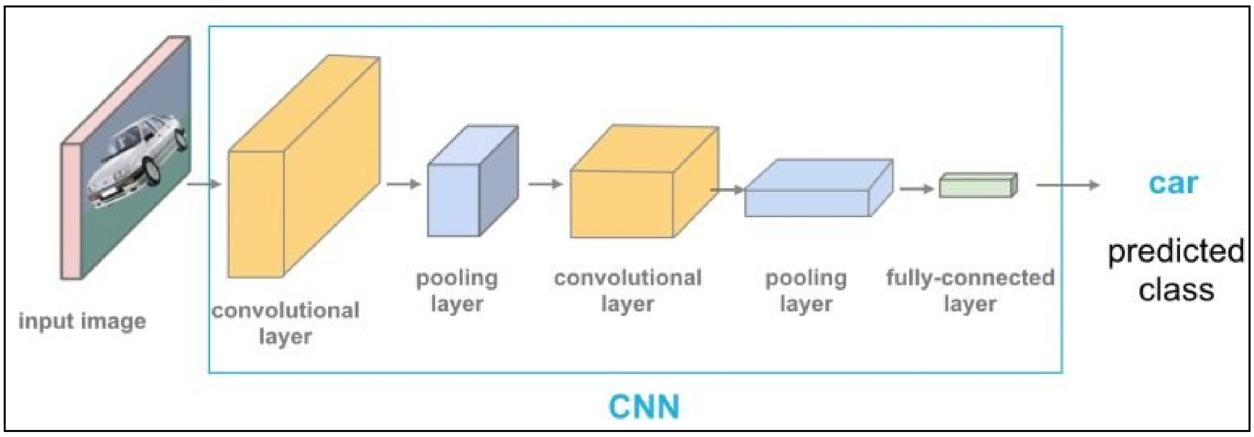


Fig 1.1.2 ***Deep Learning Convolutional Neural Network*** **[4]**

Figure 1.1.2 visually illustrates a *Convolutional Neural Network (CNN)* for image analysis tasks, delineating the essential steps involved. It depicts the input image traversing through convolutional layers, pooling layers, and fully connected layers to yield the predicted class output. The hierarchical feature extraction and processing of images by *CNNs* are pivotal in leveraging deep learning for radiogenomics and medical image analysis.

Image classification, a fundamental task in *computer vision*, entails distinguishing images based on their content. While *deep CNNs* have traditionally been utilized for image classification, recent advancements in ***Transformers***, initially designed for *natural language processing (NLP)*, have demonstrated competency in this domain. Particularly, *Vision Transformers (ViTs)* and *Multilayer Perceptrons (MLPs)* have garnered considerable attention in *computer vision* research.

The *Transformer* architecture, solely based on *attention* mechanisms, was proposed in 2017 and showed great performance on *NLP* tasks. This paved the way for the *GPT-3*, a huge transformer model released in 2020, marking a big step towards generative AI models. In the same year, *pure transformer* architectures like *ViT* were introduced, which worked well for visual recognition tasks.

Towards the end of 2020, *transformer* models found applications in low level vision tasks like image segmentation and multimodal tasks like image captioning. Unlike traditional *CNNs*, *transformers* use *self-attention* mechanisms to process data, capturing long range dependencies and relationships within the data more effectively. The *Vision Transformer (ViT)* and its advanced version, the *Swin Transformer (Shifted Window Transformer),* exemplify this adaptation. By dividing images into patches and processing them similarly to words in a sentence, these models can capture both local and global information efficiently, making them particularly powerful for analyzing complex medical images. In 2022, diffusion models like *DALLE2* demonstrated the ability to generate high quality images from *natural language* descriptions, further expanding the capabilities of *transformers*.

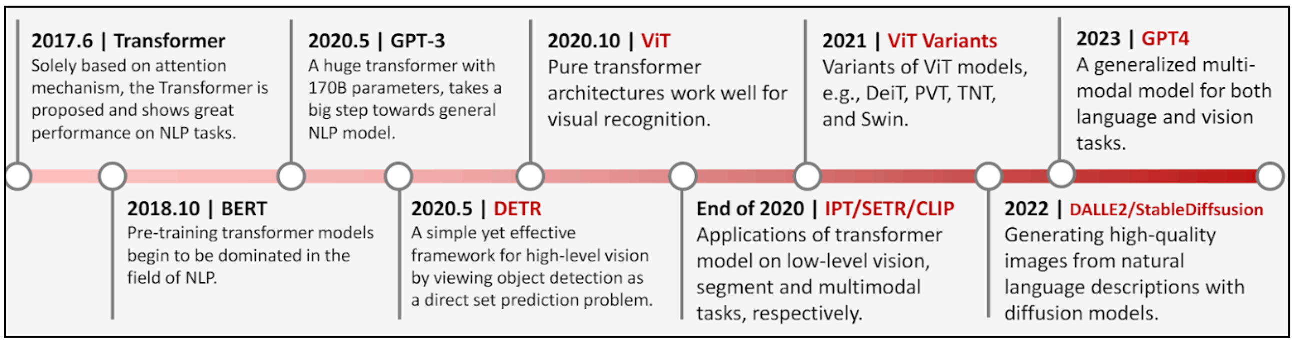

2017.6 | Transformer
Solely based on attention mechanism, the Transformer is proposed and shows great performance on NLP tasks.
2020.5 | GPT-3
A huge transformer with 170B parameters, takes a big step towards general NLP model.
2020.10 | ViT
Pure transformer architectures work well for visual recognition.
2021 | ViT Variants
Variants of ViT models, e.g., DeiT, PVT, TNT, and Swin.
2023 | GPT4
A generalized multi-modal model for both language and vision tasks.
2018.10 | BERT
Pre-training transformer models begin to be dominated in the field of NLP.
2020.5 | DETR
A simple yet effective framework for high-level vision by viewing object detection as a direct set prediction problem.
End of 2020 | IPT/SETR/CLIP
Applications of transformer model on low-level vision, segment and multimodal tasks, respectively.
2022 | DALLE2/StableDiffsusion
Generating high-quality images from natural language descriptions with diffusion models.

Fig 1.1.3 Key milestones in the development of *Transformers* [5]

The integration of advanced computational techniques holds significant promise for advancing *radiogenomics*. By leveraging *deep learning* models such as *Convolutional Neural Networks (CNNs)* and *transformers,* we can extract detailed *radiomic* features from medical images. These features, when analyzed alongside *genomic* data, can provide predictions about tumor behavior and treatment response. This interdisciplinary fusion of *radiomics, genomics, computer vision, machine learning,* and *deep learning* establishes the foundation for more precise and non-invasive diagnostic tools. Our project delves into this approach, with a specific focus on brain tumor classification. We aim to develop a robust, non-invasive method for predicting the genetic subtypes of *glioblastoma* from *MRI* scans.

1.2 PROBLEM STATEMENT

Glioblastoma, commonly referred to as glioma, represents the most lethal and prevalent type of brain tumor, resulting in a very short life expectancy in its advanced stage [3]. In recent years, researchers have identified a significant predictor of chemotherapy response in glioblastoma patients: the presence of the DNA repair enzyme *O6-Methylguanine-DNA Methyltransferase (MGMT) promoter methylation* [6] in tumor tissues. This finding holds promise for targeted therapy, offering the potential for more effective treatment strategies. However, we face a major challenge.

Current methods used to detect the *MGMT promoter methylation* require *invasive procedures*, such as surgery [7]. This reliance on invasive techniques poses several problems such as exposing patients to unnecessary risks and complications associated with surgery and prolonging the time needed to determine the most suitable treatment approach, potentially delaying crucial interventions. Our project aims to address this challenge by providing a non-invasive means of detecting *MGMT promoter methylation*. By utilizing advanced *radiogenomic* techniques and *machine learning models*, we seek to streamline treatment decisions for *glioblastoma* patients. This approach enhances patient outcomes by enabling timely and personalized interventions, reducing the burden of unnecessary surgeries and associated risks.

In essence, our project endeavors to bridge the gap between advanced computational techniques and clinical practice. We aim to develop a robust, non-invasive method for predicting genetic subtypes of *glioblastoma* from *MRI* scans, offering a pathway to more tailored and effective treatment interventions for *glioblastoma* patients. This not only improves the efficiency of clinical workflows but also significantly enhances the quality of patient care.

1.3 OBJECTIVE OF THE PROJECT

The project aims to achieve the following objectives:

- **Development of a Predictive Model:**
  Utilize the *Vision Transformer (ViT)* [8] and *Swin Transformer (Swin-T)* [9] architectures to construct a robust predictive model capable of discerning the presence of *MGMT promoter methylation* directly from *MRI* scans. This genetic marker is pivotal as it is associated with a favorable response to chemotherapy.
- **Improvement of Treatment Decision Making:**
  By enabling clinicians to quickly and accurately characterize the genetic type of *glioblastoma* tumors, we aim to enhance treatment decision-making processes. Through this, we aspire to reduce the reliance on invasive procedures and mitigate delays in initiating appropriate treatments.
- **Enhancement of Patient Outcomes:**
  The overarching goal of this project is to elevate patient outcomes. By facilitating the implementation of personalized treatment strategies tailored to the genetic profile of each *glioblastoma* patient, we seek to improve survival rates and enhance quality of life.

Through the development of innovative predictive models and the integration of advanced computational techniques, we aim to empower clinicians with the tools they need to provide more effective and personalized care to *glioblastoma* patients, ultimately leading to improved patient outcomes.

1.4 OVERVIEW OF THE REPORT

To provide a structured exploration of the predictive modeling for *glioblastoma* genetic subtypes using *MRI* scans, this report is organized as follows:

- **Chapter 2 Literature Review:** Provides a comprehensive review of related literature on predictive modeling for *glioblastoma, radiogenomics, machine learning, and computer vision* techniques. It covers existing work on the integration of imaging data with genetic information, the application of vision transformer models in medical imaging analysis, and an overview of relevant datasets used in similar studies.
- **Chapter 3 Methodology:** This chapter outlines the methodology adopted for data collection, and model development. It details the process of acquiring and preprocessing *MRI* scans for analysis, and introduces the architecture of the *Original Transformer, Vision Transformer*, *Swin Transformer* and our *Adapted Swin Transformer* model utilized for genetic subtype prediction, describes the training procedure and hyperparameters tuning, and discusses the evaluation metrics employed to assess model performance.
- **Chapter 4 Result Analysis:** Presents the findings of data visualization and the application of the *Swin Transformer* to detect the presence of *MGMT promoter methylation*. The chapter includes a performance analysis of the model, comparison with baseline models, and interpretation of model predictions.
- **Chapter 5 Conclusion:** In this final chapter, the main findings of the study are summarized, along with their clinical implications for clinical practice and future research directions. Recommendations for improving the predictive modeling approach and its application in clinical settings are provided, concluding with a reflection on the overall contributions and limitations of the study.

By following this structured framework, the report aims to provide a comprehensive exploration of the development and evaluation of predictive modeling techniques for *glioblastoma* genetic subtypes prediction from *MRI* scans.

CHAPTER 2
LITERATURE SURVEY

2.1 INTRODUCTION

The intersection of *radiology* and *genomics*, known as *radiogenomics*, offers promising avenues for enhancing the diagnosis and treatment of *glioblastoma*, an aggressive form of brain cancer. Recognizing the potential of this interdisciplinary approach, the *Radiological Society of North America (RSNA)* and the *Medical Image Computing and Computer Assisted Intervention (MICCAI)* society launched a collaborative initiative in 2021. This initiative included a *Kaggle competition* [10] focused on predicting the genetic subtype of *glioblastoma*, providing participants with the ***Brain Tumor Segmentation (BraTS) dataset***. *Figure 2.1.1* depicts all the sponsors of the competition, reflecting their broad interest and collaborative effort in this field.

BraTS
THANK YOU TO OUR SPONSORS!
intel
RSNA
Radiological Society of North America
neosoma
NIH NATIONAL CANCER INSTITUTE
SageBionetworks
kaggle

Fig: 2.1.1 Sponsors of the competition [10]

Inspired by this initiative, our project aims to build upon the foundation laid by the *RSNA* and *MICCAI* competition. We propose a novel approach utilizing the *Swin Transformer* model, a state-of-the-art architecture in *computer vision,* to analyze *MRI scans* for the detection of *MGMT promoter methylation*. This genetic marker is associated with better responses to chemotherapy, thus making its accurate identification essential for personalized treatment planning.

The *BraTS* dataset, made available through the *Kaggle* platform, provides an extensive and well-curated resource for training and validating our models. This dataset includes multimodal *MRI* scans, which are invaluable for developing and testing advanced computational methods for brain tumor segmentation and classification. By leveraging this dataset, we aim to create a predictive model that can non-invasively determine the presence of *MGMT promoter methylation*, thereby aiding in the accurate and timely diagnosis of glioblastoma.

This literature review will delve into the various methodologies and technologies that underpin our approach, including the application of *deep learning* models such as *Vision*

Transformers (ViTs) and *Swin Transformers* in medical imaging. We will explore the advancements in *radiogenomics* and how these innovations contribute to improved diagnostic accuracy and treatment efficacy. Additionally, we will examine the significance of the *BraTS* dataset in advancing research and development in this domain. Through this review, we aim to provide a comprehensive understanding of the current state-of-the-art techniques and their application in the field of brain tumor radiogenomic classification.

2.2 REVIEW OF RELATED WORK

The *BraTS* challenge has been running annually since 2012 [11], attracting researchers and practitioners from various fields to tackle the problem of brain tumor segmentation and analysis. Over the years, a wide range of techniques has been proposed and evaluated, including traditional machine learning, deep learning, and hybrid models.

2.2.1 Early Approaches: Traditional Machine Learning

In the early editions of the challenge, traditional machine learning methods such as *random forests* [12], *support vector machines* [13] and *decision trees* were commonly employed for brain tumor segmentation and classification tasks. These methods relied on handcrafted features extracted from the MRI scans, such as texture features, intensity-based features, and shape-based features. While these methods provided a solid foundation, their performance was often limited by the quality and relevance of the manually extracted features.

2.2.2 Deep Learning: Convolutional Neural Networks (CNNs)

With the advent of deep learning and the success of *convolutional neural networks (CNNs)* in various computer vision tasks, researchers began exploring *CNN-based* approaches for brain tumor segmentation and analysis. Notable work includes the use of *U-Net*, a popular *CNN* architecture for medical image segmentation, and its variants. *U-Net* and its derivatives leverage the encoder-decoder structure to capture both local and global features, significantly improving segmentation accuracy. *CNNs,* with their ability to learn hierarchical features directly from the data, have surpassed traditional methods, leading to state-of-the-art performance in many tasks.

2.2.3 Ensemble and Hybrid Models

In recent years, *ensemble models* and *hybrid approaches* that combine traditional machine learning methods with deep learning techniques have gained popularity. These methods aim to leverage the strengths of different techniques and improve overall performance. For example, some models combine the robustness of *CNNs* in feature extraction with the decision-making capabilities of traditional *machine learning* classifiers. This hybrid approach has shown promise in enhancing model accuracy and robustness.

2.2.4 Transformer Architecture in Vision Tasks

The *transformer architecture*, originally introduced for natural language processing tasks, has also been adopted for computer vision problems. ***Vision Transformer (ViT)*** has recently emerged as a competitive alternative to *convolutional neural networks (CNNs)* that are currently state-of-art in different image recognition computer vision tasks. *ViT* models outperform the current *CNNs* by almost x4 in terms of computational efficiency. They have shown promising results in various image classification and segmentation tasks by treating images as sequences of patches and applying the transformer architecture to model long-range dependencies.

2.2.5 Swin-Transformer Approach

Building upon the success of *ViT*, the ***Swin Transformer (Swin-T)*** was proposed as a hierarchical vision transformer that incorporates a shifted windowing scheme to capture local and global information more effectively. The *Swin Transformer* has demonstrated superior performance compared to traditional *CNNs* and *ViT* on various computer vision benchmarks. Its hierarchical design and shifted windowing mechanism allow it to efficiently process high-resolution images and model complex spatial dependencies.

2.2.6 RSNA-MICCAI BraTS Challenge 2021

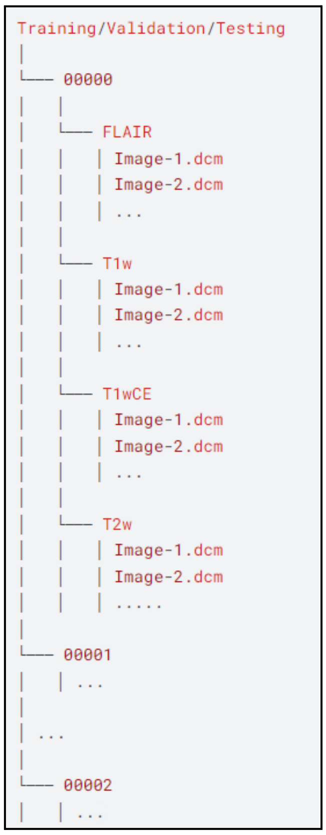
The 2021 edition of the *RSNA-MICCAI BraTS* challenge, which this work focuses on, attracted around 1958 members proposing solutions for the problem statement, with 1556 submissions being ranked. The top nine submission teams have published their solutions, which employed architectures such as ***EfficientNet-B0, B3, B1, ResNet-10, Bidirectional LSTM, U-Net*** [10]. These models represent a range of advanced techniques, showcasing the diversity and innovation in the field. Notably, the challenge took place in July 2021, coinciding with the publication of the *Swin Transformer* research paper. Consequently, no solution using the *Swin Transformer* architecture was reported for this challenge.

The pre-existing work related to the *BraTS* challenge has provided valuable insights and served as a foundation for further exploration. The evolution from traditional *machine learning* methods to sophisticated *deep learning* architectures highlights the continuous advancement in the field. The integration of transformer architectures, such as *ViT* and *Swin-T*, marks a significant shift towards more efficient and effective models for brain tumor segmentation and analysis.

By building on these advancements, our project aims to leverage the *Swin Transformer* architecture to develop a robust predictive model for glioblastoma classification. This approach seeks to enhance the accuracy and efficiency of non-invasive diagnostic tools, ultimately improving treatment decision-making and patient outcomes.

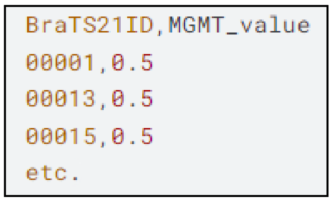
2.3 DATASET

The dataset utilized for this project is the ***BraTS*** 2021 dataset **[14]**, which is approximately 140 GB in size and comprises *MRI* scans of the brain in the *DICOM* format. Each case folder is identified by a unique five-digit number, known as the *BraTS21_ ID*. This dataset is pivotal for the tasks of brain tumor segmentation and classification, providing a comprehensive set of *multi-parametric MRI (mpMRI)* scans essential for developing robust predictive models.



```
Training/Validation/Testing
|
└── 00000
|   |
|   └── FLAIR
|   |   | Image-1.dcm
|   |   | Image-2.dcm
|   |   | ...
|   |
|   └── T1w
|   |   | Image-1.dcm
|   |   | Image-2.dcm
|   |   | ...
|   |
|   └── T1wCE
|   |   | Image-1.dcm
|   |   | Image-2.dcm
|   |   | ...
|   |
|   └── T2w
|   |   | Image-1.dcm
|   |   | Image-2.dcm
|   |   | .....
|
└── 00001
|   | ...
|
| ...
|
└── 00002
|   | ...
```

Fig 2.3.1 ***folder structure***



```
BraTS21ID,MGMT_value
00001,0.5
00013,0.5
00015,0.5
etc.
```

Fig 2.3.2 ***sample_submission.csv***

Each case folder contains four sub-folders, each corresponding to one of the four structural multi-parametric *MRI (mpMRI)* scans i.e *Fluid Attenuated Inversion Recovery (FLAIR), T1-weighted pre-contrast (T1w), T1-weighted contrast enhanced (T1wCE), T2-weighted* images.

FLAIR images are instrumental in highlighting brain lesions by suppressing the *cerebrospinal fluid (CSF)* signal. This suppression makes abnormalities such as edema and tumors more visible, aiding in the precise localization and characterization of brain tumors.

T1w images provide high-resolution detail of the brain's anatomy, making them useful for distinguishing between different types of tissue, particularly gray and white matter. This anatomical detail is crucial for understanding the spatial relationships and structural integrity of brain tissues.

T1-wCE (contrast enhanced) images are acquired after administering a contrast agent, which highlights blood vessels and regions with a disrupted blood-brain barrier, such as tumors. The contrast enhancement allows for better visualization of tumor boundaries and vascular structures, essential for accurate tumor delineation.

T2w images are sensitive to differences in the water content of tissues, making them particularly useful for detecting edema, inflammation, and other fluid-related changes in the brain. These images help in identifying areas of abnormal fluid accumulation associated with tumors.

These different imaging modalities offer unique insights into the brain's structure and pathology, enhancing the accuracy of tumor analysis and classification.

Additionally, the root directory of the dataset includes two critical files: *train_labels.csv* and *sample_submission.csv*. The *train_labels.csv* file contains the target *MGMT_value* for each *BraTS21_ID* in the training data, which indicates the methylation status of the *MGMT promoter* - a crucial genetic marker associated with response to chemotherapy. The *sample_submission.csv* file provides a template for formatting the final submission of predictions for the competition.

The *BraTS* 2021 dataset is well-curated and comprehensive, providing a rich source of data for training and validating models aimed at brain tumor segmentation and classification. The inclusion of multiple *MRI* modalities ensures that models can leverage diverse features for improved accuracy and robustness.

This dataset has been pivotal in various research efforts aimed at advancing brain tumor analysis. The combination of high-quality imaging data and detailed annotations facilitates the development of sophisticated models capable of performing complex tasks such as tumor segmentation, classification, and the prediction of genetic markers. Our project leverages this dataset to explore the potential of *Vision Transformers (ViT)* and *Swin Transformers (Swin-T)* in accurately predicting the *MGMT promoter methylation* status from *MRI* scans, aiming to improve non-invasive diagnostic methods and enhance treatment decision-making in clinical practice.

CHAPTER 3
METHODOLOGY

3.1 DATA LOADING

The *BraTS* dataset serves as our primary data source. This dataset comprises brain *MRI* scans, which consist of multiple 2D *DICOM* image slices typically available in three orthogonal planes: *axial, coronal, and sagittal.*

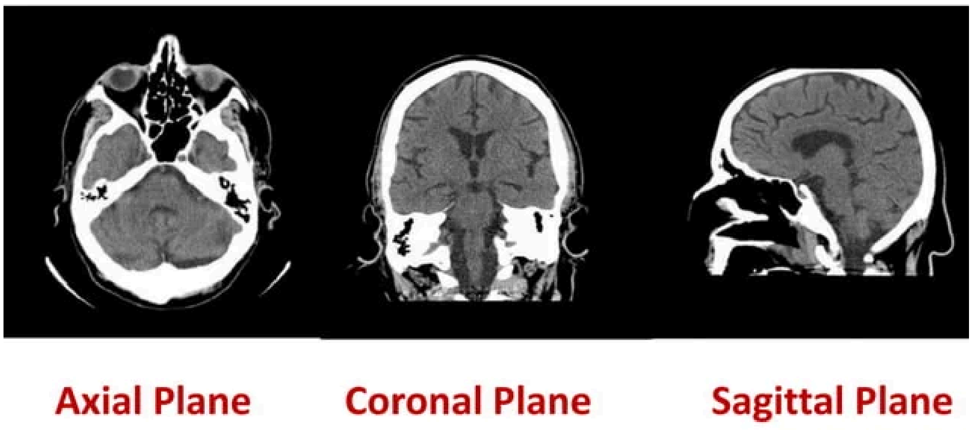


Fig 3.1.1 Orthogonal planes of 2D DICOM image slices

Axial Plane

The axial plane is parallel to the ground and perpendicular to the long axis. In terms of a brain *MRI* scan, axial slices are obtained by slicing the brain horizontally, parallel to the ground. This means the resulting images show structures from top to bottom or bottom to top. Axial images in a *DICOM MRI* scan show cross-sectional views of the brain, with structures like the cerebrum, cerebellum, brainstem, and surrounding tissues visible.

Coronal Plane

The coronal plane is perpendicular to both the ground and the long axis. In the context of a brain *MRI*, coronal slices are obtained by slicing the brain vertically, from one ear to the other. This results in images showing structures from the front to the back or vice versa. Coronal images in a *DICOM MRI* scan provide a side view of the brain, displaying structures such as the frontal lobes, temporal lobes, parietal lobes, and their relationships.

Sagittal Plane

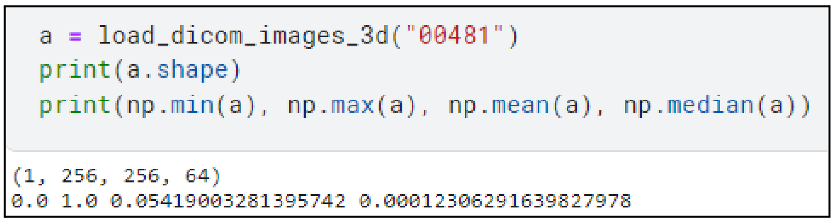
The sagittal plane is perpendicular to both the ground and the long axis, and divides into left and right halves. In a brain *MRI*, sagittal slices are obtained by slicing the brain vertically, from the midline to one side. This results in images showing structures from the center of the brain outwards to one side. Sagittal images in a *DICOM MRI* scan provide a lateral view of the brain, showing structures such as the cerebral hemispheres, corpus callosum, and midline structures like the thalamus. These different planes offer complementary views of the anatomy, aiding in diagnosis and treatment planning for various medical conditions, including brain tumors in the case of the *BraTS dataset*.

Loading Function

The objective of our *load_dicom_images_3d* function is to form a comprehensive 3D representation of the brain by aggregating the 2D *DICOM* image slices into a 3D array. This approach retains the spatial information and inter-slice relationships essential for accurate analysis.

To achieve this, we begin by loading *DICOM* images for a specific *BraTS21_ID* and *MRI_type*. We then identify the middle index within the list of *DICOM* images and load a fixed range of images based on the *num_imgs* parameter. These images are preprocessed and stacked into a 3D numpy array using the *load_dicom_images_3d* function. If the folder contains fewer images than specified by the *num_imgs* parameter, we pad the 3D array with zeros to match the desired image count.

Throughout the data loading process, we normalize the pixel values of the 3D array by subtracting the minimum value and dividing by the maximum value. Finally, we introduce an additional dimension for batch size, ensuring the data format aligns with batch processing requirements. The function then returns the preprocessed 3D array, ready for further analysis.



```
a = load_dicom_images_3d("00481")
print(a.shape)
print(np.min(a), np.max(a), np.mean(a), np.median(a))
```

```
(1, 256, 256, 64)
0.0 1.0 0.05419003281395742 0.00012306291639827978
```

Fig 3.1.2 Shape of a sample 3D array returned by *load_dicom_images_3d* function

Figure 3.1.2 Output of the *load_dicom_images_3d* function for BraTS21_ID “00481”. The resulting 3D array, stored in variable ‘a’, is analyzed for its shape, minimum and maximum values, as well as mean and median values. The shape of ‘a’ (1, 256, 256, 64) denotes the batch size, height, width, and depth of the array.

Dataset Loader

The *load_dicom_images_3d* function is designed to load 2D *DICOM* image slices and aggregate them into a 3D array. To utilize this function effectively across our extensive *BraTS dataset*, we define a class *BraTSDataset*. This class streamlines the process of loading and preprocessing *DICOM* images for each *BraTS21_ID* and *MRI_type*, ensuring consistency and efficiency.

The *BraTSDataset* class initializes various parameters such as the *file paths, target values, MRI types, label smoothing*, and the data frame containing the *metadata*. An optional *augmentation flag* is also included to enable random rotations of the images, enhancing the robustness of the dataset for training purposes.

Within the member functions of the class, we retrieve the 3D image arrays and corresponding target values. First, we load the *DICOM* images for a given index using the *load_dicom_images_3d* function. If augmentation is enabled, a random rotation is applied to the images. The images are then converted to a *PyTorch* tensor for compatibility with *deep learning models*.

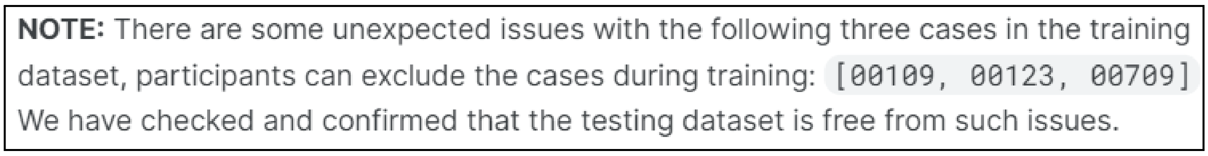

NOTE: There are some unexpected issues with the following three cases in the training dataset, participants can exclude the cases during training: `[00109, 00123, 00709]` We have checked and confirmed that the testing dataset is free from such issues.

Fig 3.1.3 Issues with three cases in the training dataset [10]

The *target* parameter is initialized to *None* by default. For the *df_train* data frame, the *target* is *MGMT_value*. In the *df_valid* (validation) and *submission* (test) data frames, the target is left as *None*. When target values are provided, they are adjusted using *label smoothing* (set to 0.01) to avoid overfitting and then converted to tensors. The method ultimately returns a dictionary containing the 3D images and their respective targets or identifiers, making it suitable for training. For validation and test data frames, the method returns a dictionary containing the 3D images and their respective *scan_id* (*BraTS21_ID*). This approach ensures efficient data loading while integrating preprocessing and augmentation, which are crucial for developing robust machine learning models.

Additionally, there were issues with three cases in the training dataset (*BraTS21_ID*: [00109, 00123, 00709]). This error was automatically handled within our *load_dicom_images_3d* function by padding the entire 3D arrays with zeros and returning them to our dataset. This approach avoided the issue. *Figure 3.1.3* depicts this specific issue within the dataset.

3.2 ORIGINAL TRANSFORMER

The *Transformer* is a sequence-to-sequence model that relies solely on *attention mechanisms*, without using any *convolutional* or *recurrent* operations. It consists of an encoder and a decoder, both composed of multiple identical layers.

Figure 3.2.1 depicts the overall architecture of the *Original Transformer* model proposed in the influential paper *"Attention Is All You Need"* by *Vaswani et al.* in 2017 [15]. The encoder on the left side takes the input sequence and processes it through multiple layers.

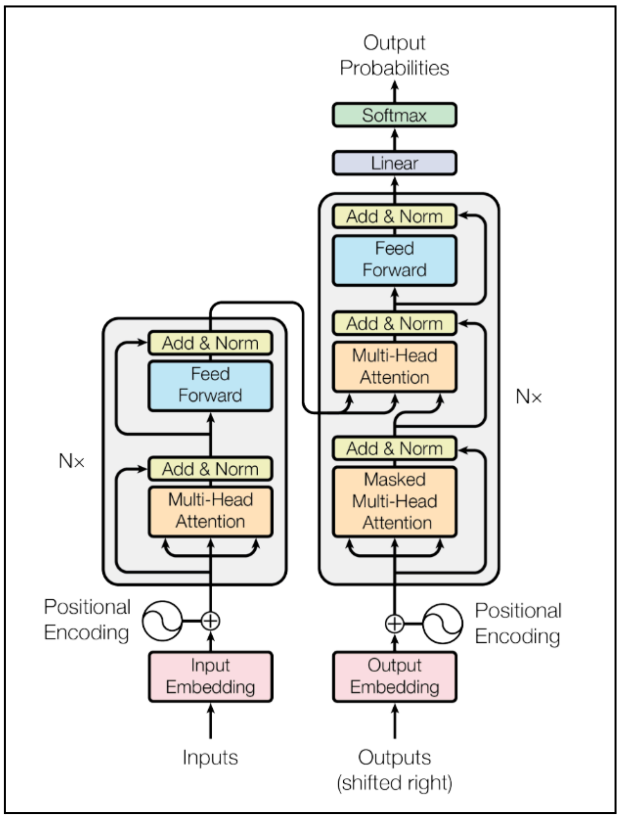


Fig 3.2.1 Original Transformer [15]

1) **Patch Embedding**: The input tokens are first converted into dense vector representations known as *embeddings*.
2) **Positional Encoding**: Since the *Transformer* does not have any *recurrent* or *convolutional* components, it requires positional information to be injected into the *embeddings* to capture the order of the sequence.
3) **Add & Norm**: Each layer in the encoder has a residual connection followed by *layer normalization*, a technique that helps with training *deep neural networks*. In traditional *neural networks,* the distribution of the inputs to a layer can change during training, which can slow down the learning process and make it harder for the network to converge. This issue is known as the *"internal covariate shift"* [16] problem. *Layer normalization* aims to address this problem by normalizing the inputs across the features (or units) within a layer, instead of normalizing across the batch dimension as in batch normalization.
4) **Multi-Head Attention**: The core component of the *Transformer* is the *multi-head attention* mechanism, which allows the model to attend to different parts of the input sequence in parallel and capture long-range dependencies.

The decoder on the right side follows a similar structure, but with an additional masked *multi-head attention* layer. This layer prevents the decoder from attending to subsequent positions in the output sequence, ensuring that predictions for a certain position can only depend on known outputs. The outputs from the encoder and decoder are combined through another *multi-head attention* layer, which allows the decoder to attend to the encoded input sequence. This is followed by a *feed-forward* layer and the final add & norm operation. Finally, a *linear layer* and a *softmax* operation are applied to produce the output probabilities over the target vocabulary for each position in the output sequence.

The *Transformer* architecture, with its reliance on *attention* mechanisms and parallelizable computations, revolutionized sequence modeling tasks in *natural language processing* and has since been adapted to various other domains, including *computer vision* and *speech processing.*

3.3 ARCHITECTURE OF VISION TRANSFORMER (ViT) MODEL

The *Transformer* architecture has set the highest standard for tasks in *Natural Language Processing (NLP)*. However, its application in *computer vision* tasks was initially limited. Traditionally, *attention* mechanisms were either used in conjunction with *Convolutional Neural Networks (CNNs)* or to substitute certain aspects of *CNNs* while keeping their core structure intact. This dependency on *CNNs* diminished with the introduction of the *Vision Transformer (ViT)* model, which applies a *pure transformer* model directly to sequences of image patches for image classification tasks.

The *Vision Transformer (ViT)* model architecture was introduced in a research paper published as a conference paper at the *International Conference on Learning Representations (ICLR)* 2021 titled *"An Image is Worth 16*16 Words: Transformers for Image Recognition at Scale"* [8]. The paper was authored by *Neil Houlsby, Alexey Dosovitskiy*, and ten more authors of the *Google Research Brain Team*. The *ViT* model has achieved highly competitive performance in benchmarks for several *computer vision* applications, such as image classification, object detection, and semantic image segmentation.

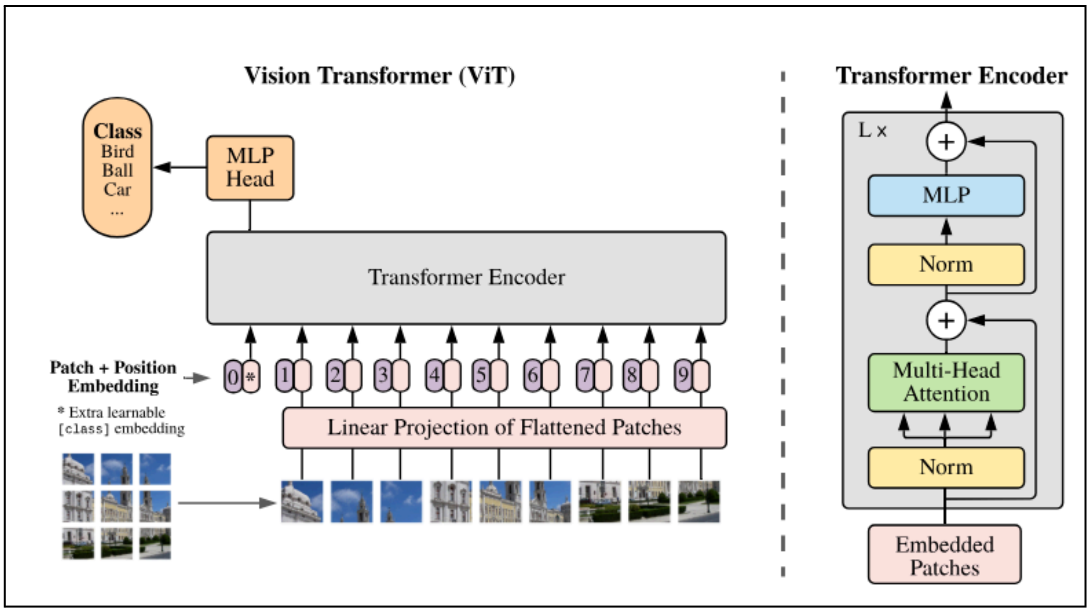


Fig 3.3.1 Model overview of Vision Transformer [8]

Figure 3.3.1 represents an overview of the *Vision Transformer (ViT)*. While the core components of *ViT* remain similar to the *Original Transformer,* there are a few key modifications and improvements made to the *Transformer Encoder* to efficiently handle the image data. This section provides an overview of the architecture of *ViT*, detailing the components and functionality of each layer.

1) **Patch Embedding**: In the *original Transformer* architecture, the input consists of a sequence of token embeddings, where tokens represent words in the context of *natural language processing (NLP)* tasks. In *Vision Transformer (ViT)*, this concept is adapted for images by partitioning the input image into a sequence of fixed-size patches. Analogous to words in *NLP*, these image patches serve as the basic units of information. Each patch is then flattened and linearly projected into a D-dimensional vector.
   a) *Class token* is an additional learnable embedding that is added to the sequence of patch embeddings. This token aggregates information about the entire image, contributing to the final classification task.
2) **Position Embedding**: Since the order of image patches matters to preserve the 2D positional information, a learnable positional embedding is added to each patch embedding, similar to the approach in *NLP tasks.*
3) **Transformer Encoding**: In *ViT*, this layer closely follows the structure of the *original transformer* model, comprising multiple identical layers. The sequence of patch embeddings, which includes the *class token*, is fed into a series of these layers. Each layer consists of a *Multi-Head Self-Attention (MSA)* mechanism and a *Multi-Layer Perceptron (MLP/ feed forward layer)*, augmented with *layer normalization* and residual connections.
   a) The *Multi-Head Self-Attention (MSA)* layer remains largely unchanged from the *original transformer,* enabling the model to assess the importance of each patch relative to all others, facilitating informed predictions.
   b) *Layer Normalization* is applied both before and after the *Multi-Head Self-Attention (MSA)* and *feed-forward* layers, contributing to the stability of the training process.
   c) *Multi-Layer Perceptron (MLP)* is a simple *feed-forward* network, often comprising two linear layers followed by a nonlinearity such as the activation function *Gaussian Error Liner Unit (GELU*). In the context of *Vision Transformer (ViT)*, this *MLP* is applied independently to each patch embedding. This process enhances the model's capacity to capture intricate patterns within individual patches.
4) **Multi-Layer Perceptron (MLP) Head**: After passing through all the transformer encoder layers, the *class token's* embedding is processed by a small *multi-layer perceptron (MLP)* head to produce the final class probabilities.

These modifications in *ViT* allowed the *Transformer* architecture to effectively process and understand image data, while still leveraging the powerful *self-attention* mechanism for capturing long-range dependencies and global context.

3.4 ARCHITECTURE OF SWIN TRANSFORMER (SWIN - T) MODEL

The *Vision Transformer (ViT)* marked a significant shift in *computer vision* by applying the *Transformer* architecture directly to image data, challenging the traditional dominance of *Convolutional Neural Networks (CNNs)*. However, *ViT's* performance was constrained by its inability to efficiently capture long-range dependencies in images due to its patch-based processing.

In response to this limitation, the *Swin Transformer (Shifted Window Transformer/ Swin-T)* emerged as a notable advancement in the field of *computer vision.* The *Swin-T,* introduced in a research paper titled *"Swin Transformer: Hierarchical Vision Transformer using Shifted Windows"* at the conference on *Computer Vision and Pattern Recognition (CVPR)* 2021 [9], represents a significant evolution of the *Transformer* architecture tailored specifically for image processing tasks.

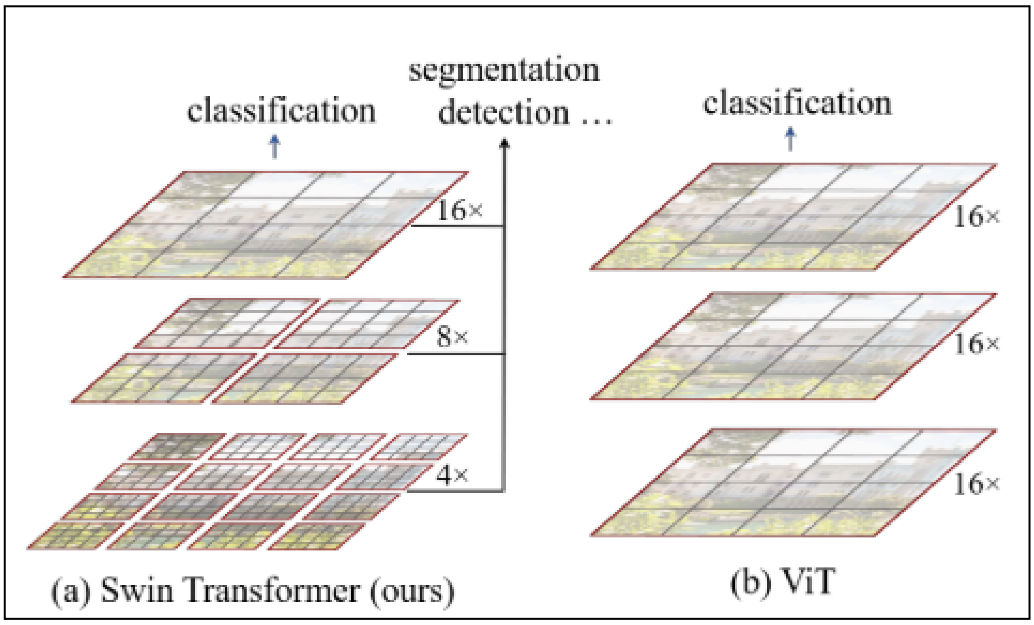


Fig 3.4.1 Hierarchical feature map architecture in *Swin-T* [9]

Authored by *Ze Liu, Yutong Lin, Yue Cao, Han Hu,* and *Liwei Wang* from *Microsoft Research Asia,* the *Swin-T* introduces a novel hierarchical architecture that operates on shifted windows instead of fixed-size patches. By recursively partitioning the input image into a *hierarchy of shifted windows*, the *Swin-T* can efficiently capture both local and global dependencies, enabling it to achieve state-of-the-art performance on various computer vision tasks. It was initially applied to the standard image classification dataset, *ImageNet* [17], which is a large-scale dataset containing millions of images across thousands of categories.

The *Swin-T*s hierarchical architecture allows it to effectively model interactions between image regions at different spatial scales, overcoming the limitations of the patch-based processing used in *ViT*. This innovation has led to significant advancements in tasks such as image classification, object detection, and semantic segmentation, further solidifying the *Transformer* architecture's position as a versatile and powerful tool in *computer vision*.

Figure 3.4.1 (a) represents the *Swin-T* which builds hierarchical feature maps by combining smaller image patches into larger ones as it goes deeper into the network. It does this by computing self-attention only within each local window of the image. This approach keeps the *computation complexity linear* as the input image size increases. Because of this efficiency and hierarchical feature representation, the *Swin-T* can be used for various tasks like image classification and dense recognition. Whereas, the *ViT* in *Figure 3.4.1 (b)* creates feature maps of a single low resolution. They compute *self-attention* globally across the entire image, resulting in a quadratic increase in computational complexity as the input image size grows.

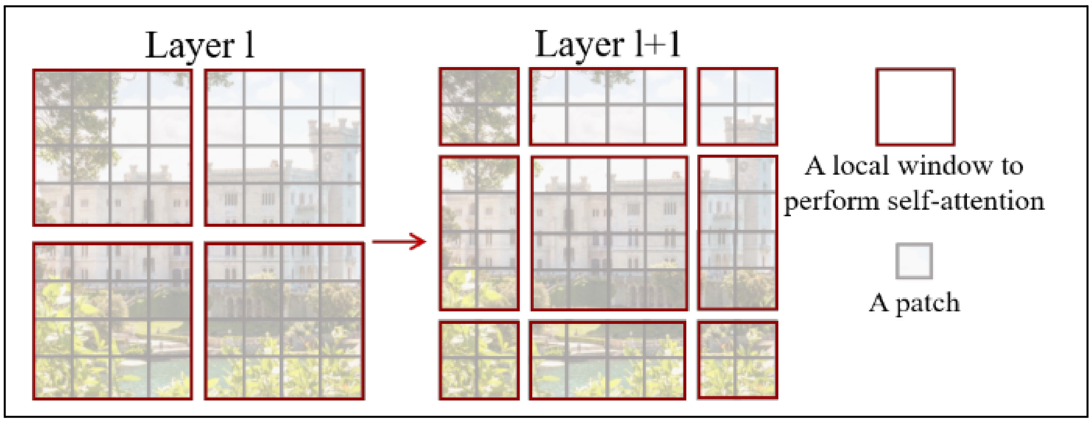


Fig 3.4.2 Shifted window approach in *Swin-T* [9]

Figure 3.4.2 shows how the *Swin-T* computes *self-attention* using a shifted window approach. On the left side of the illustration (Layer *l*), the image is divided into regular windows, and *self-attention* is calculated within each window. Each window processes information independently. Whereas on the right size of the illustration (Layer *l* + 1), the window partitioning is shifted, creating new windows. These new windows overlap with the boundaries of the previous windows from layer *l*. As a result, the *self-attention* computation in the new windows crosses the boundaries of the previous windows, allowing information flow and connection between them. This connectivity helps capture more context and dependencies between different paths of the image, improving the model's ability to understand relationships between features.

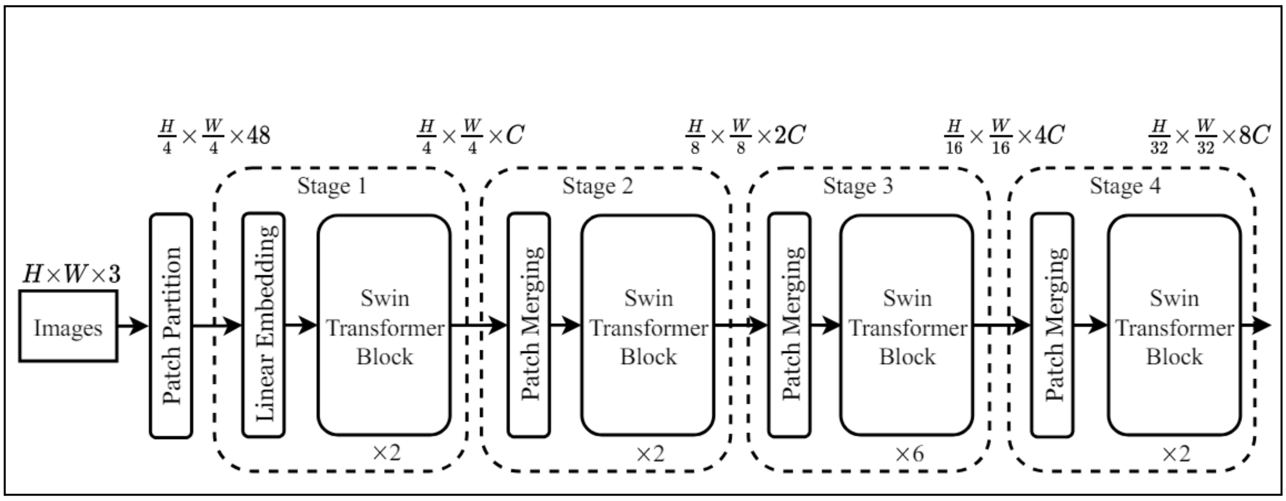

$\frac{H}{4}\times\frac{W}{4}\times48$
$\frac{H}{4}\times\frac{W}{4}\times C$
$\frac{H}{8}\times\frac{W}{8}\times 2C$
$\frac{H}{16}\times\frac{W}{16}\times 4C$
$\frac{H}{32}\times\frac{W}{32}\times 8C$
Stage 1
Stage 2
Stage 3
Stage 4
$H\times W\times3$
Images
Patch Partition
Linear Embedding
Swin Transformer Block
Patch Merging
×2
×2
×6
×2

Fig 3.4.3 Architecture of *Swin-T* [9]

Figure 3.4.3 illustrates the Architecture of the *Swin-T* model as proposed in the influential paper *"Swin Transformer: Hierarchical Vision Transformer using Shifted Windows"* [9]. While it follows a similar overall architecture to the *ViT*, it introduces several significant modifications.

This section provides a detailed explanation of each layer implemented in *Swin-T* based on the code implemented in the paper.

1) **Patch Embedding:** Similar to *ViT*, the *Swin-T* begins by partitioning the input image into *(H*W*3)* into non-overlapping patches. Where, H is the height, W is the width of the image and 3 indicates the three color channels (RGB). Each patch size is 4*4 in dimension, resulting in *(H/4)*(H/4)*48* patches.
    a) Linear Embedding is applied to project the patches into higher dimensional space with C feature channels. Thus, the output dimension is of *(H/4)*(H/4)*C*

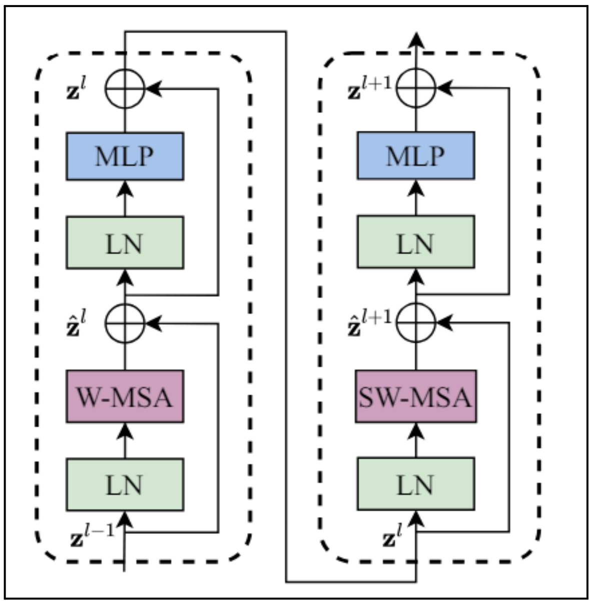


Fig: 3.4.4 Two successive *Swin Transformer Blocks* [9]

2) **Swin Transformer Block:** This layer incorporates two main innovations, particularly within the *Swin Attention module: windowed multi-head self-attention (W-MSA)* and *shifted windowed multi-head self-attention (SW-MSA),* in an alternating pattern across *transformer blocks* as depicted in *Figure 3.4.4.*
   a) *Stage 1*
      i) Input Size: The output of linear embedding *(H/4)*(W/4)*C* is processed, maintaining same number of channels *C*
      ii) Components: Two *Swin Transformer Blocks* with *window-based self-attention*.
   b) Stage 2
      i) Patch Merging: After each stage, a patch merging layer is applied, which reduces the spatial dimensions (height and width) by a factor of 2 and increases the number of feature channels.
      ii) So the feature map of size *(H/4)*(W/4)*C* is downsampled to *(H/8)*(W/8)* while doubling the number of channels to *2C* resulting in *(H/8)*(W/8)*2C*
      iii) Components: Two *Swin Transformer Blocks* .
   c) Stage 3
      i) Patch Merging: Downsamples further to *(H/16)*(W/16)*4C*
      ii) Components: Six *Swin Transformer Blocks*
   d) Stage 4
      i) Patch Merging: Final downsample to *(H/32)*(W/32)*8C*
      ii) Components: Two *Swin Transformer Blocks*
   e) *Window-based Self-Attention (W-MSA)*: Each window of size *M*M* applies self-attention within that window, reducing computational complexity compared to global self-attention used in *ViT.*
   f) *Shifted Window-based Self-Attention (SW-MSA)*: Introduces overlapping windows by shifting the windows by a certain number of patches, which enables cross-window connections and better captures local and global context.
3) **Patch Merging:** After each stage, patch merging layers are applied to downsample the spatial dimensions while increasing the feature dimension. This hierarchical representation allows the model to progressively capture more complex features.
4) **Multi-Layer Perceptron (MLP) Head:**

In the above, *C* is the base number of feature channels that determine the dimensionality of feature representations at various stages of the *Swin-T*. Each patch merging step not only reduces the spatial dimensions but also increases the number of channels, allowing the model to capture more complex features hierarchically.

In essence, the *Swin-T* enhances the *ViT* by introducing hierarchical processing, efficient self-attention mechanisms, and a novel method of capturing both local and global contexts, resulting in improved performance on image classification tasks.

Some key modifications compared in *Swin-T* to *ViT*

a) **Hierarchical Structure**: Unlike *ViT,* which processes images as a sequence of patches without changing resolution, *Swin-T* adopts a hierarchical structure with patch merging, leading to multi-scale feature maps.
b) **Window-based Self-Attention**: Reduces computational complexity by limiting self-attention to non-overlapping windows within each layer.
c) **Shifted Windows**: Enhances the model's ability to capture cross-window dependencies and global context.
d) **Absence of Class Token**: Uses global average pooling instead of a *class token* to aggregate information for classification.

3.5 ADAPTED SWIN TRANSFORMER FUNCTIONALITY

The original code of the *Swin Transformer* is extensive and complex, making it challenging to integrate into our code. Therefore, using the code of *Vision Transformer (ViT) from Scratch* [18] and referencing the official *Swin-T* code [19], we have introduced an *adapted Swin-T* model.

This section provides a detailed explanation of each layer implemented in *Swin-T* and its functional workings within our *BraTS (Swin ViT Model) - version 4* code.

1) **Patch Embedding** converts input image patches into embeddings suitable for processing by the transformer network.
    a) Initializes the module with parameters such as *image size, patch size, input channels, and embedding dimension*.
    b) Uses a 2D convolutional layer *(nn.Conv2d)* to project the input image patches into embedding vectors with a specific dimension.
    c) Token Initialization creates learnable parameters for the *class token* and *patch tokens* which represent global and local information in the image, respectively.
    d) The input image is processed through the projection layer, and rearranges the output to form a sequence of patches with embedded representations. Adds *class* and *patch tokens* to the sequence.
2) **Positional Embedding** adds positional information to the input embeddings, allowing the transformer network to understand the spatial relationship between patches.

a) Initializes the module with parameters such as *image size, patch size, embedding dimension, and dropout rate.*

b) Creates learnable parameters representing *positional* information for each patch in the input sequence.

c) Applies *dropout* to the *positional embeddings* to prevent overfitting during training.

d) Adds the *positional embeddings* to the *input embeddings*, incorporating spatial information into the input sequence.

3) **Swin Transformer Block** represents a single block within the *Swin Transformer* network. Initializes the module with parameters such as *embedding dimension, number of attention heads, MLP ratio, and dropout rates.*

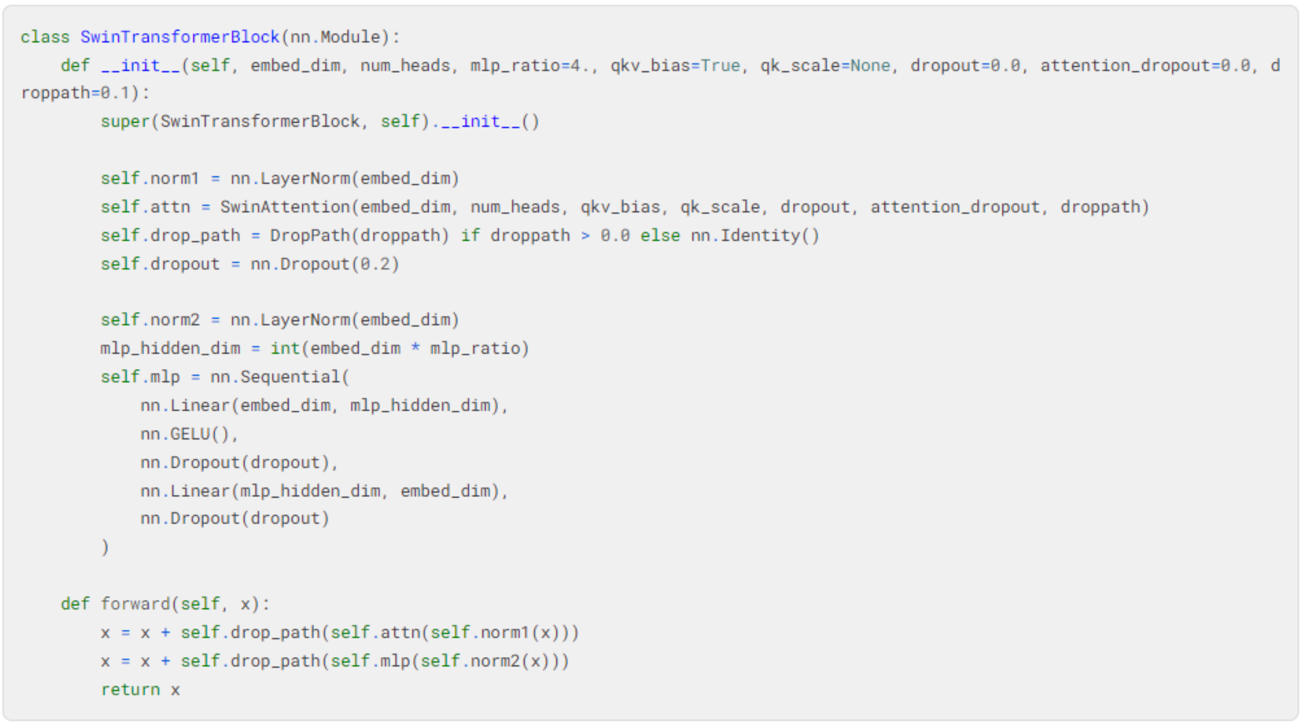
   a) Applies ***layer normalization (LN)*** to the input embeddings before processing.

   b) Utilizes the *Swin Attention* module to perform *multi-head self-attention (MSA)* on the input embeddings, capturing global and local dependencies.

   c) Applies dropout with optional drop path regularization to the output of the attention mechanism.

   d) The *MLP* employs a feedforward neural network consisting of linear layers and *GELU activation* to process the output of the attention mechanism.

   e) Adds the output of the attention mechanism and *MLP* to the input embeddings, allowing the block to capture both global and local information.



```
class SwinTransformerBlock(nn.Module):
    def __init__(self, embed_dim, num_heads, mlp_ratio=4., qkv_bias=True, qk_scale=None, dropout=0.0, attention_dropout=0.0, d
roppath=0.1):
        super(SwinTransformerBlock, self).__init__()

        self.norm1 = nn.LayerNorm(embed_dim)
        self.attn = SwinAttention(embed_dim, num_heads, qkv_bias, qk_scale, dropout, attention_dropout, droppath)
        self.drop_path = DropPath(droppath) if droppath > 0.0 else nn.Identity()
        self.dropout = nn.Dropout(0.2)

        self.norm2 = nn.LayerNorm(embed_dim)
        mlp_hidden_dim = int(embed_dim * mlp_ratio)
        self.mlp = nn.Sequential(
            nn.Linear(embed_dim, mlp_hidden_dim),
            nn.GELU(),
            nn.Dropout(dropout),
            nn.Linear(mlp_hidden_dim, embed_dim),
            nn.Dropout(dropout)
        )

    def forward(self, x):
        x = x + self.drop_path(self.attn(self.norm1(x)))
        x = x + self.drop_path(self.mlp(self.norm2(x)))
        return x
```

Fig 3.5.1 ***SwinTransformerBlock*** **module of our** ***Adapted Swin-T***

4) **Swin Attention**: The key difference between *ViT* and *Swin-T* lies in the *attention* mechanism.
    a) Initializes the module with parameters such as *embedding dimension, number of attention heads, and dropout rates.*
    b) Projects the input embeddings into *query, key, and value (QKV) vectors* using linear transformations.
    e) Calculates *attention* scores between *query* and *key vectors*, scaled by the square root of the head dimension. Applies *softmax* to obtain *attention weights* and applies *dropout* for *regularization.*
    f) Projects the weighted value vectors back to the original embedding dimension using another *linear transformation.*
    g) Performs the above steps and returns the output of the *attention mechanism.*

5) **Drop Path** applies drop path regularization to the input embeddings.
    a) Initializes the module with the *dropout* probability.
    b) During training, it applies drop path *regularization* by randomly zeroing out elements of the input embeddings with a probability. During inference, the input embeddings are unchanged.

6) **Swin Transformer** represents the entire Swin Transformer network.
    a) Initializes the module with parameters such as *image size, patch size, embedding dimension, number of layers, number of attention heads, and dropout rates.*
    b) Uses the *Patch Embedding* and *Positional Embedding* modules to embed the input image patches and add positional information.
    c) Utilizes a stack of *Swin Transformer Block* modules to process the input embeddings through multiple layers of attention and MLPs.
    d) Applies *layer normalization* to the output embeddings of the transformer layers.
    e) Uses a linear layer to map the final embeddings to the output classes.
    f) Finally processes the input through the *patch embedding, positional embedding, transformer layers*, and head to produce the final output.

3.5 TRAINING PROCEDURE

The training procedure for our *Swin-T* model is implemented within the *Trainer* class. The class is initialized with parameters including the *model, device (CPU/ GPU), optimizer, and loss criterion.* The *fit* method oversees the training and validation data loaders.

During each *epoch* of training, the model undergoes a training phase where it learns from the training dataset. Here, the *Trainer* computes the average training loss, providing us insight into how well the model is learning the patterns in the data.

Following the training phase, the model is assessed using the validation dataset. This step is crucial as it helps us understand how well the model generalizes to unseen data. In addition to computing the validation loss, the *Trainer* also calculates the *Area Under the Receiver Operating Characteristic Curve (AUC)* metric, which gives us a measure of the model's performance in binary classification tasks.

To prevent *overfitting*, a common challenge in machine learning, the training process employs early stopping. This technique halts the training if the validation loss fails to improve over a specified number of patience epochs. This strategy ensures that the model doesn't overly specialize to the training data and maintains its ability to generalize to new examples.

Furthermore, the Trainer class saves the best-performing model checkpoint as a file named *{mri}-best.pth*, where *{mri}* represents the type of *MRI data* used for training. This checkpoint includes crucial information such as the model's state dictionary, optimizer state dictionary, the best validation score achieved, and the epoch at which it occurred. This allows us to later reload the best model and continue training or use it for inference on new data.

In this way, our *Trainer* class encapsulates the entire training procedure, from data loading to model evaluation and checkpointing, ensuring that our *Swin-T* model learns effectively and generalizes well to unseen data.

3.6 HYPERPARAMETERS

During the training process for our model, *hyperparameter tuning* was done extensively to ensure effective learning and optimal performance.

Firstly, we adopt the *Adam optimizer* from the torch.optim library, with a carefully selected *learning rate* of 0.0005 and a *weight decay* of 1e-5. Through experimentation, this *optimizer* and *learning rate* combination demonstrated the most promising results compared to alternatives such as *RMSprop* and *Stochastic Gradient Descent (SGD).* Despite varying the learning rates for these optimizers, they failed to yield better *Area Under the Curve (AUC)* scores, reaffirming the effectiveness of *Adam optimizer* for our task.

For the loss function, we employ the *Binary Cross-Entropy with Logits Loss*. This loss criterion is well-suited for binary classification tasks like ours, helping the model effectively learn the intricacies of distinguishing between positive and negative instances in the data.

In configuring the data loaders, we set a *batch size* of 4 for both training and validation phases. This batch size strikes a balance between computational efficiency and model performance, allowing for efficient processing of data while providing sufficient information for the model to learn from.

The training procedure spans over 15 *epochs,* during which the model learns from the training data and is evaluated on the validation set. To prevent *overfitting* and ensure the model generalizes well, early stopping is implemented with a *patience* of 10 *epochs*. This mechanism halts the training process if the validation performance does not improve for consecutive epochs, thereby preventing the model from memorizing the training data and enabling it to generalize effectively.

Crucially, the training process is performed iteratively for each *MRI type* in the dataset. This iterative training approach ensures that the model is trained and optimized for each specific *MRI type,* capturing the unique characteristics and nuances present in the data. As a result, model checkpoint files are generated for each *MRI type*, which are stored in the modelfiles list for future reference and utilization. This iterative training strategy enhances the robustness and adaptability of our model, enabling it to perform effectively across different *MRI types* encountered in real-world scenarios.

3.7 EVALUATION METRICS

To assess the performance of our *Adapted Swin Transformer* model on the *BraTS dataset* for the detecting *MGMT promoter methylation* genetic content, the following metrics are utilized.

1) ***AUC score* of the *ROC curve***

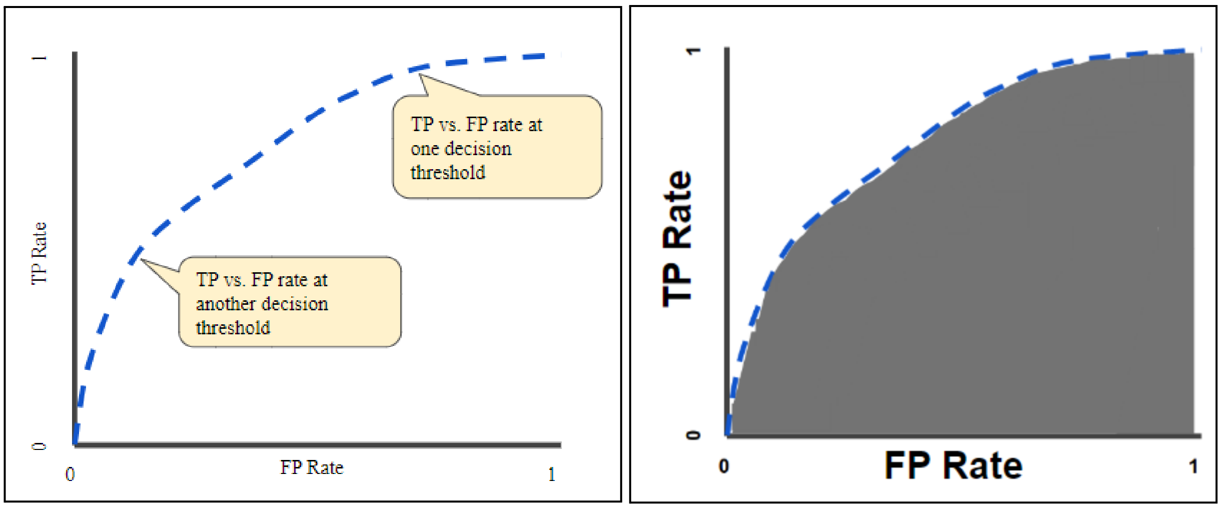


Fig 3.7.1 *ROC curve* [30]

a) The *ROC curve* (*receiver operating characteristic curve*) shows the performance of a classification model at all classification thresholds.

b) It plots the *True positive rate (sensitivity)* versus *False positive rate (specificity)* at different classification thresholds.

c) *AUC* (*Area under ROC curve*) measures the entire area beneath the entire *ROC curve*. It provides an aggregate measure of performance across all possible classification thresholds.

d) *AUC* can be interpreted as the probability by which a model ranks a random positive example more highly than a random negative example.

e) *AUC* ranges *i*n value from 0 to 1. A model whose predictions are 100% wrong has an *AUC* of 0.0 and the model whose predictions are 100% correct has an *AUC* of 1.0

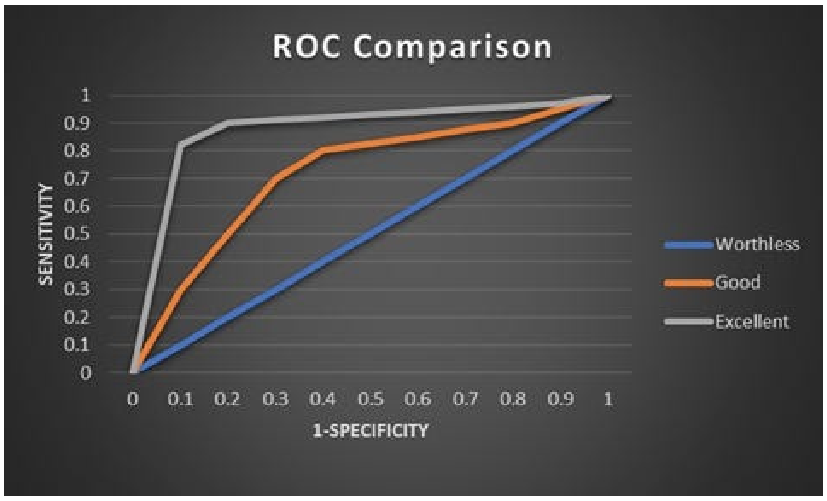


Fig 3.7.2 ROC curve comparison [31]

f) *Figure 3.7.2* represents *ROC curve* comparison. The blue line represents a worthless model as it just acts like a random classifier. Whereas the orange line curve represents a good model and the gray line curve represents an excellent model.

2) **Accuracy**

$$\boxed{\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}}$$

Fig 3.7.3 [30]

a) It serves as a fundamental metric for assessing classification models.

b) In the context of binary classification, accuracy can be computed based on the counts true positives, true negatives, false positives and false negatives, as depicted in *Fig 3.7.3*

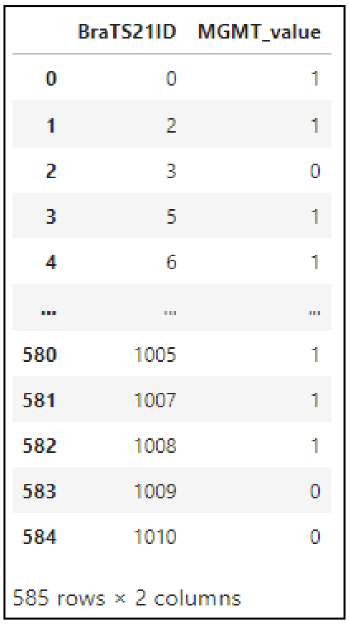
The evaluation metrics outlined above are implemented by importing in-built functions from the *scikit-learn* library.

CHAPTER 4
RESULT ANALYSIS

4.1 DATA PREPROCESSING AND VISUALIZATION

The initial steps of our work focused on data preprocessing, which included refinement, normalization, and augmentation. These steps were crucial for preparing the dataset for effective training and validation of the model.

We began by loading the *train_labels.csv* file into a dataframe *df* using the pandas library. The adjacent figure *Figure 4.1.1* illustrates the *BraTS21_ID* and corresponding *MGMT_value* of 585 patients present in the *csv* file.



	BraTS21ID	MGMT_value
0	0	1
1	2	1
2	3	0
3	5	1
4	6	1
...	...	...
580	1005	1
581	1007	1
582	1008	1
583	1009	0
584	1010	0

585 rows × 2 columns

Fig 4.1.1 *df* dataframe

Using the *train_test_split* function from the *scikit-learn library*, we divided *df* into training and validation datasets, *df_train* and *df_valid*, respectively, in an **80:20** ratio. We set the *random_state* parameter to a value 9 to ensure reproducibility and applied the *stratify* parameter to *df['MGMT_value']* to ensure proportional representation of *MGMT_values* in both sets. *Figure 4.1.2* depicts the count of *MGMT_values* in both the training and validation datasets after the split, using a *seaborn* countplot. This preprocessing ensured that our data was clean, balanced, and ready for the subsequent training and validation phases.

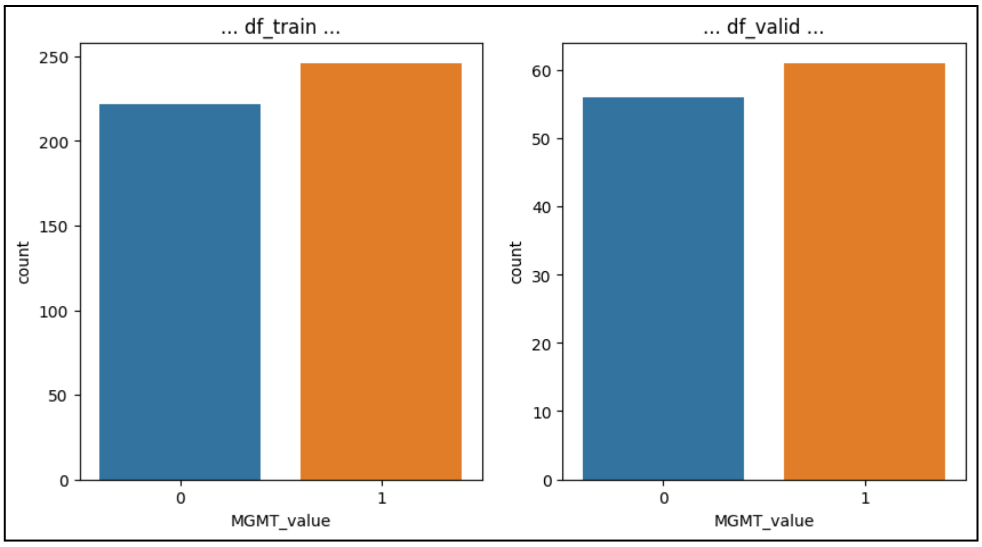
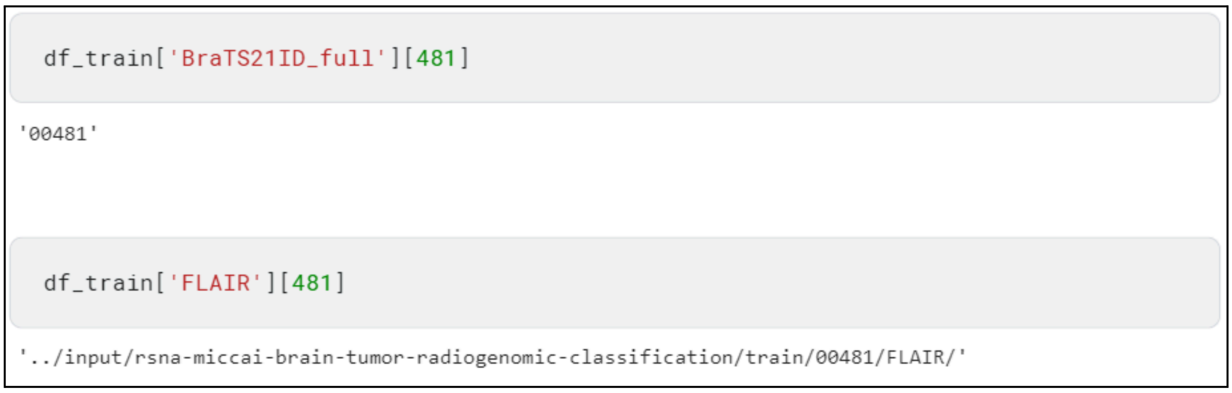


Fig 4.1.2 count of *MGMT_values* after splitting *df* into *df_train* and *df_valid*

After splitting the *train_labels.csv* dataframe into training and validation dataframes, we further preprocessed them by adding new columns. These include *BraTS21ID_full*, which involves adding leading zeros to the *BraTS21_ID* to ensure easy access to the corresponding image folders during training and validation. *Figure 4.1.3* illustrates the easy data access post refinement, while *Figure 4.1.4* showcases the modified *df_train* dataframe. In this modified dataframe, the *BraTS21_ID* has been set as the key column, and additional columns (*FLAIR, T1w, T1wCE, T2w*) have been appended. These additional columns contain the paths to each MRI type for every case id, facilitating easy access to the necessary image data.



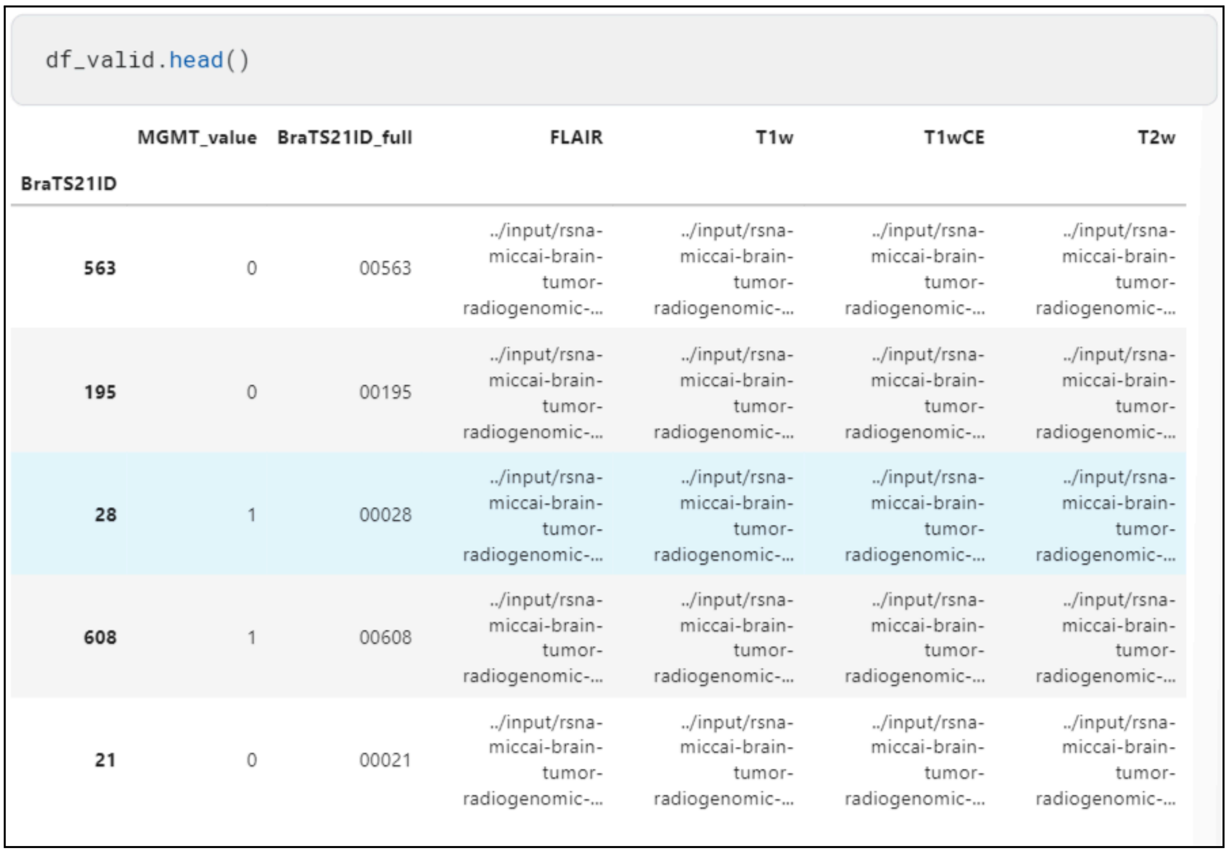
```
df_train['BraTS21ID_full'][481]
```

```
'00481'
```

```
df_train['FLAIR'][481]
```

```
'../input/rsna-miccai-brain-tumor-radiogenomic-classification/train/00481/FLAIR/'
```

Fig 4.1.3 easy data access after refinement



```
df_valid.head()
```

BraTS21ID	MGMT_value	BraTS21ID_full	FLAIR	T1w	T1wCE	T2w
563	0	00563	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...
195	0	00195	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...
28	1	00028	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...
608	1	00608	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...
21	0	00021	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...	../input/rsna-miccai-brain-tumor-radiogenomic-...

Fig 4.1.4 modified and refined *df_train*

After refining both the dataframes, two user-defined functions, *load_sample* and *visualise_sample*, are defined. Using these functions the *DICOM* images from sample cases are loaded and visualized.

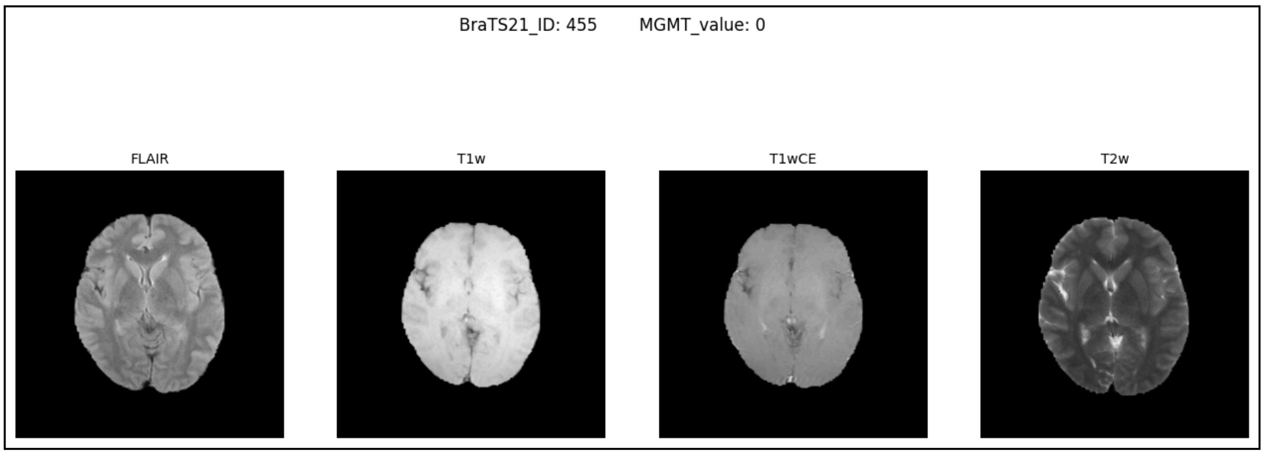


Fig 4.1.5 (a) middle image from each *MRI_type* of *BraTS21_ID* 455

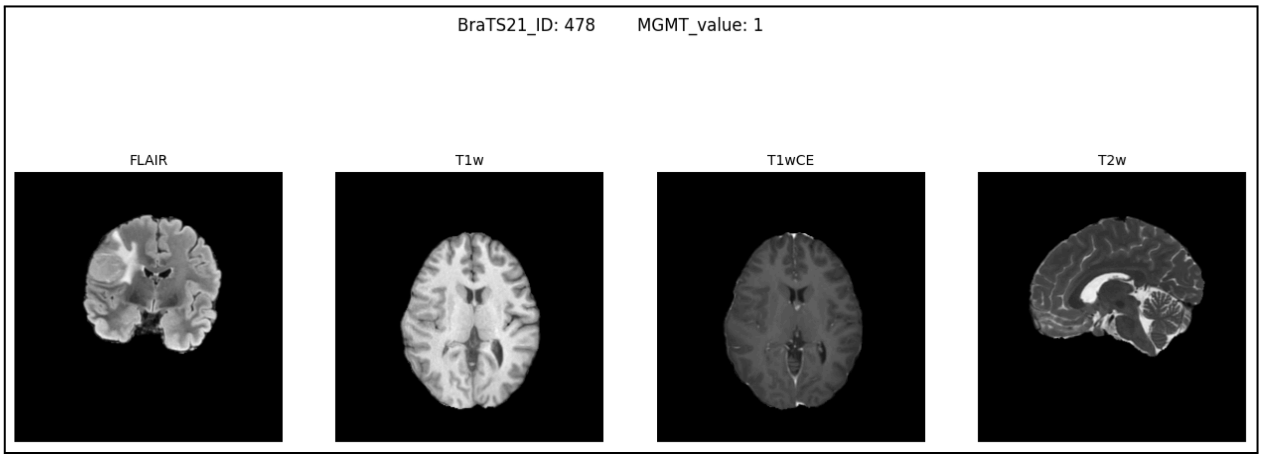


Fig 4.1.5 (b) middle image from each *MRI_type* of *BraTS21_ID* 478

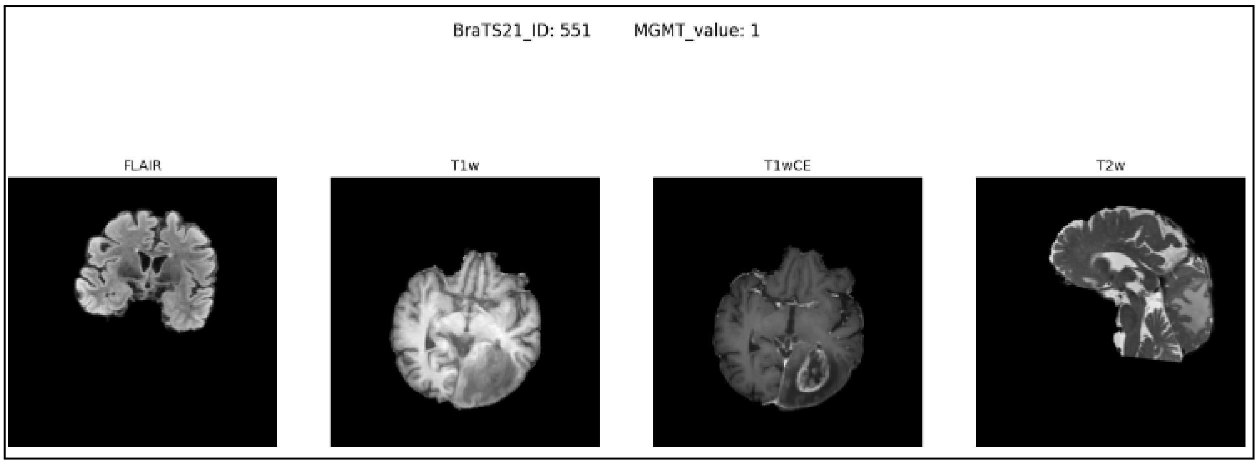


Fig 4.1.5 (c) middle image from each *MRI_type* of *BraTS21_ID* 551

The *visualise_sample* function takes *BraTS21_ID*, its corresponding *MGMT_value* as parametric inputs, and the image number. Based on these inputs, it sends the image path to the *load_sample* function. This function then reads the *DICOM* image file using *dicom.pixel_array*, normalizes the pixel values, and returns a pixel array. Finally, this pixel array data is visualized, as shown below.

In *Figure 4.1.5 (a)*, the image for *BraTS21_ID* 455 with an *MGMT_value* of 0 indicates the absence of the *MGMT promoter methylation* gene in the tumor tissue of the patient. The four MRI types displayed *(FLAIR, T1w, T1wCE, and T2w)* are all presented along the *axial plane*, which is a horizontal slice through the body.

In *Figure 4.1.5 (b)* and *Figure 4.1.5 (c)*, an *MGMT_value* of 1 indicates the presence of *MGMT promoter methylation* in patients with *BraTS21_IDs* 478 and 551, respectively. In both figures, the first three MRI types *(FLAIR, T1w, and T1wCE)* are shown along the axial plane, while the *T2w* images are displayed along the *sagittal plane*, which is a vertical slice through the brain, viewed from the side.

4.2 DATA AUGMENTATION AND LOADING

The *load_dicom_image* function reads a single *DICOM* image from the specified file path and applies the *Value of Interest Look-Up Table (VOI LUT)* transformation *(voi_lut)*. This technique is commonly used in medical imaging to enhance or modify the contrast and intensity levels of specific regions or structures within an image. By improving the visibility and differentiation of certain features or areas of interest, this transformation helps in better diagnosing and analyzing medical images. When the *voi_lut* parameter is set to *True*, the transformation adjusts the pixel values of the *DICOM* images, enhancing contrast.

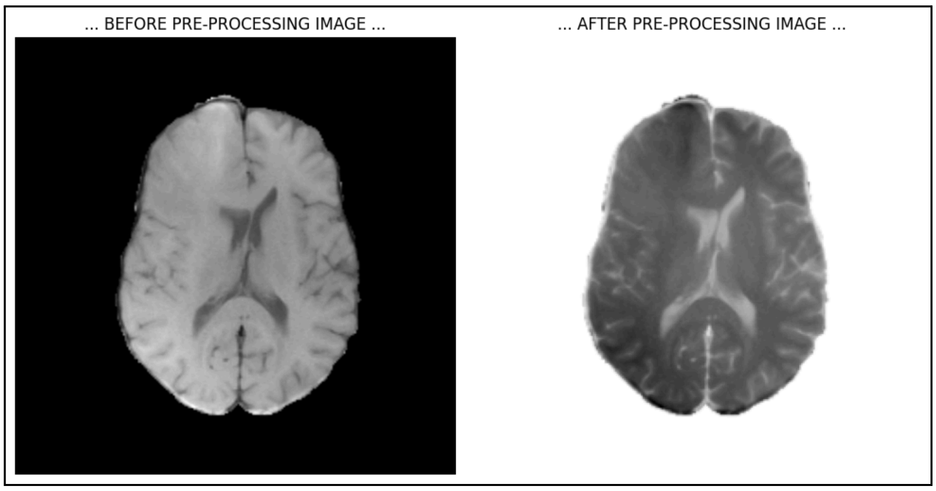

... BEFORE PRE-PROCESSING IMAGE ...
... AFTER PRE-PROCESSING IMAGE ...

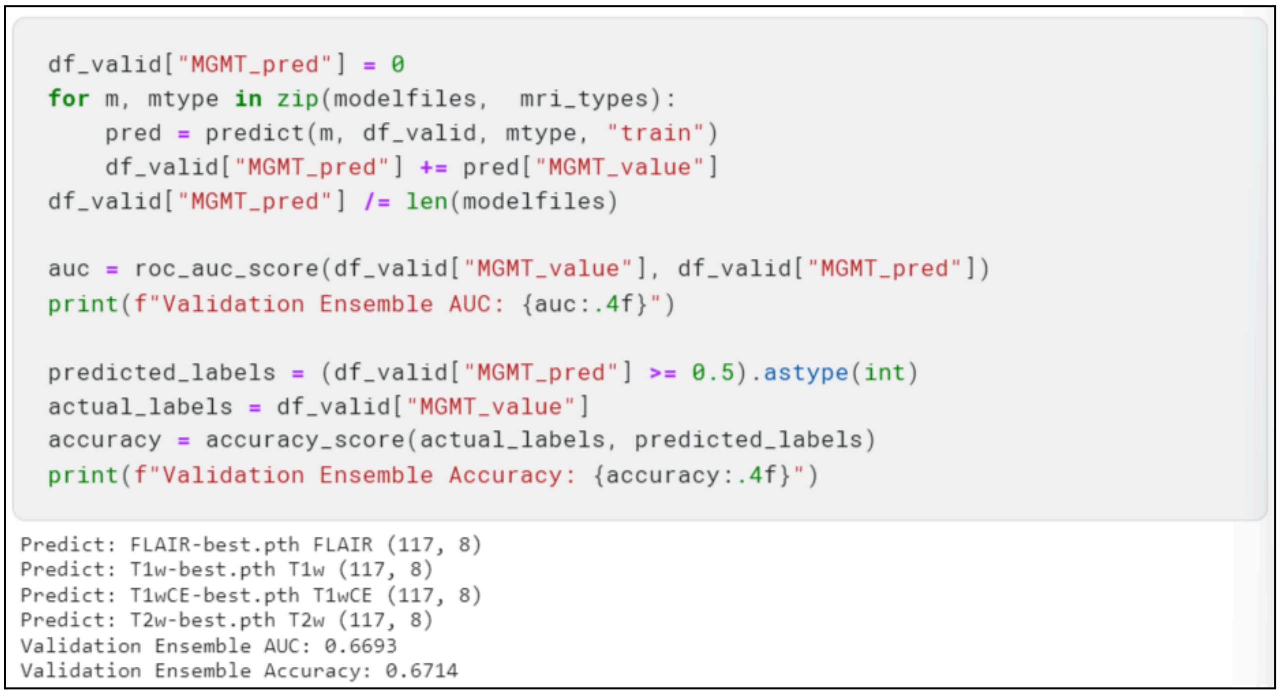
Fig 4.2.1 T1-weighted Image-18.dcm (BraTS_ID 481)

Additionally, image rotation is applied using *cv2.rotate* within the *load_dicom_image* function. The rotate parameter determines the degree of rotation: 0 for no rotation, 1 for 90 degrees clockwise rotation, 2 for 90 degrees counterclockwise rotation, and 3 for 180 degrees rotation.

Figure 4.2.1 illustrates the preprocessing and augmentation process for the *T1w Image-18.dcm (BraTS_ID 481)*. The first image shows the visualization of the *DICOM* image directly as a pixel array using *dicom.pixel_array*. The second image displays the visualization after processing it through the *load_dicom_image* function, which internally enhances the contrast.

4.3 INTERPRETATION OF MODEL PREDICTIONS

After splitting the data frame *df* into *df_train* and *df_valid* in an **80:20** ratio, a total of 585 *BraTS21_ID* data points were divided. Specifically, 468 *BraTS21_ID* data points were allocated to the *df_train* dataset, and the remaining 117 *BraTS21_ID* data points were assigned to the *df_valid* dataset. The entire 468 *BraTS21_ID* data points were used for training the model, while the remaining 117 *BraTS21_ID* data points were used for validation.



```
df_valid["MGMT_pred"] = 0
for m, mtype in zip(modelfiles,  mri_types):
    pred = predict(m, df_valid, mtype, "train")
    df_valid["MGMT_pred"] += pred["MGMT_value"]
df_valid["MGMT_pred"] /= len(modelfiles)

auc = roc_auc_score(df_valid["MGMT_value"], df_valid["MGMT_pred"])
print(f"Validation Ensemble AUC: {auc:.4f}")

predicted_labels = (df_valid["MGMT_pred"] >= 0.5).astype(int)
actual_labels = df_valid["MGMT_value"]
accuracy = accuracy_score(actual_labels, predicted_labels)
print(f"Validation Ensemble Accuracy: {accuracy:.4f}")
```

```
Predict: FLAIR-best.pth FLAIR (117, 8)
Predict: T1w-best.pth T1w (117, 8)
Predict: T1wCE-best.pth T1wCE (117, 8)
Predict: T2w-best.pth T2w (117, 8)
Validation Ensemble AUC: 0.6693
Validation Ensemble Accuracy: 0.6714
```

Fig 4.3.1 Validation ensemble AUC and accuracy after prediction on df_valid

The model was trained for 15 *epochs* with a *batch size* of four. After training, the generated model files *(mri_type-best.pth)* were used to predict *MGMT_pred* on the validation data frame *df_valid*. The average value of these predictions was then calculated as the *MGMT_pred* probability.

Subsequently, based on the predicted probability and the observed target, the *validation ensemble Area Under the Curve (AUC)* and *accuracy* were determined using the *roc_auc_score* and *accuracy_score* functions from the scikit-learn library. *Figure 4.3.1* illustrates how the validation ensemble AUC and accuracy are calculated.

Initially, *MGMT_pred* was set to 0, and a for loop was initialized. This loop calls the predict function for each model file and MRI type, storing its return value in a temporary variable pred. At each iteration, the predicted *MGMT_value* probability is accumulated in *df_valid["MGMT_pred"]*. After executing the loop, the accumulated predicted probabilities are divided by the number of model files (four in our case), effectively averaging the probabilities. The output resembles the application of the predict function on each model file *(FLAIR-best.pth, T1w-best.pth, T1wCE-best.pth, T2w-best.pth)*, and the shape of the data is (117, 8), with 117 being the *BraTS21_IDs* of the *df_valid* used for validation. Our prediction ultimately resulted in a *validation ensemble AUC of* **0.6693** and an *accuracy* of **0.6714.**

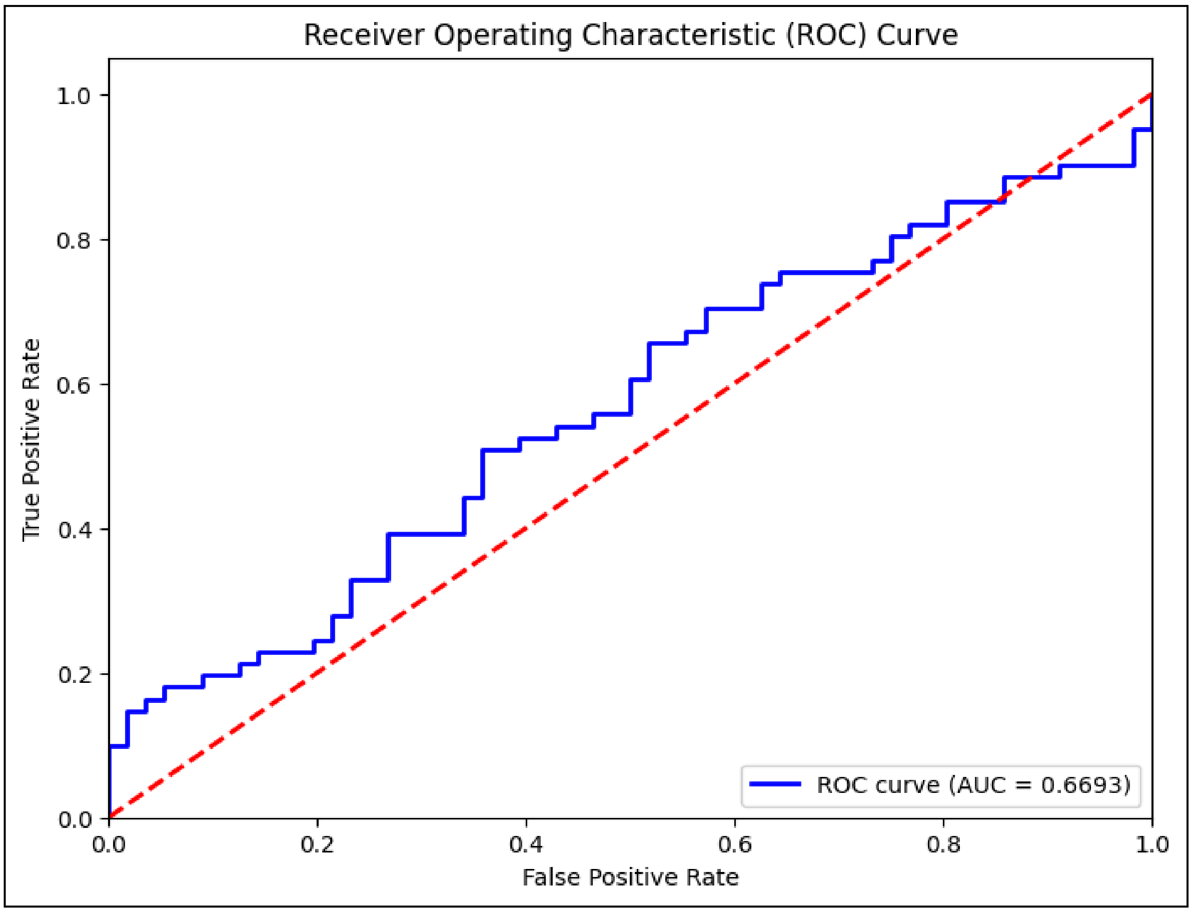


Fig 4.3.2 *ROC curve* obtained on the *df_valid*

Figure 4.3.2 depicts the *ROC curve* obtained after predicting *MGMT_pred* on *df_valid*. The *Area Under the Curve (AUC)* value of **0.6693** represents the area enclosed between the *blue ROC curve* and the *false positive rate axis*. The *red dotted line* represents the performance of a *random classifier,* whose *AUC score* is 0.5. If our entire *ROC curve* falls below this line, it indicates that the model is not a *good classifier*, as the *AUC score* would be less than 0.5.

MGMT Presence Probability Distribution Histograms	

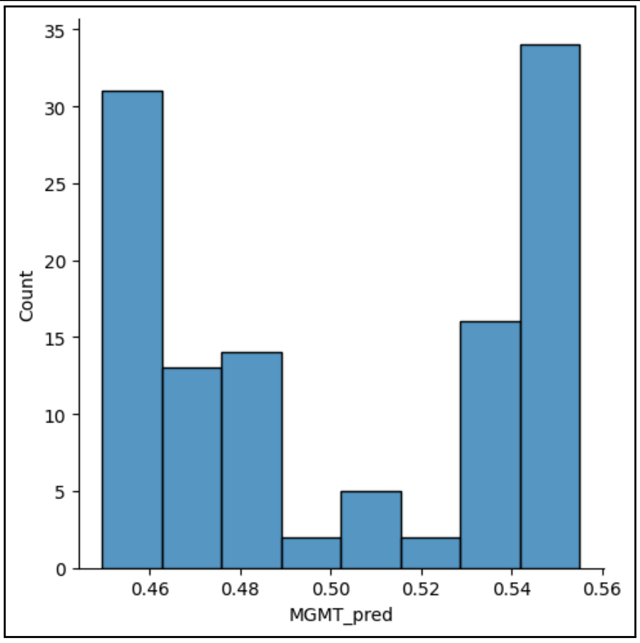
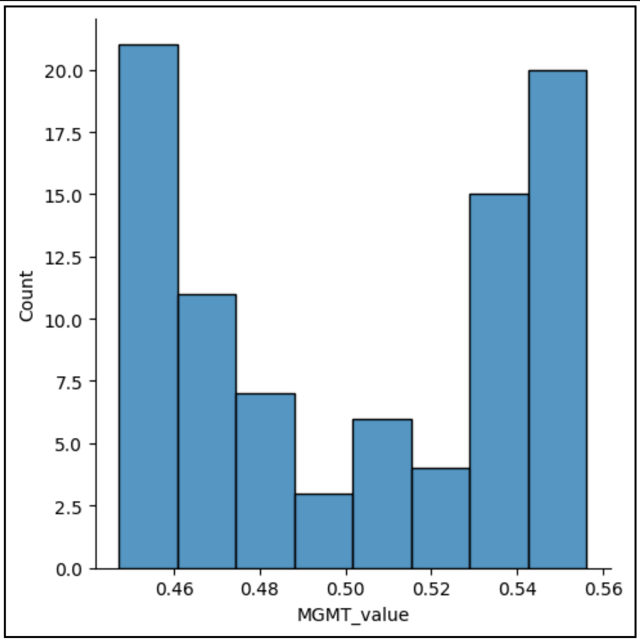

	

Fig 4.3.3 (a) ***df_valid[“MGMT_pred”]***	**Fig 4.3.3 (b)** ***submission[“MGMT_value”]***

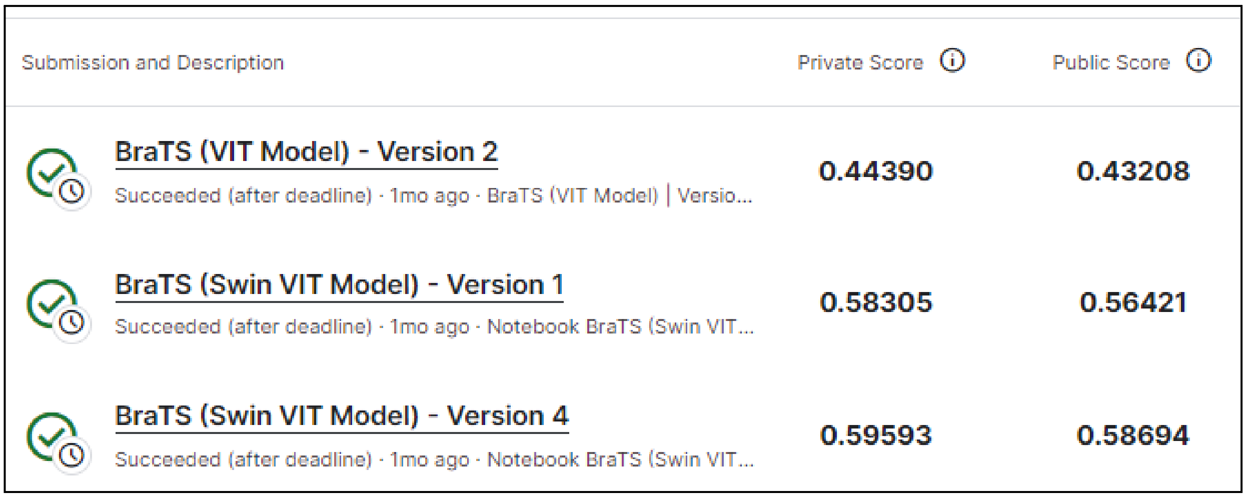
Figure 4.3.3 (a) shows the probability distribution histogram of *MGMT_pred* values, obtained after executing the *predict* function on the *df_valid* data frame. Similarly, *Figure 4.3.3 (b)* displays the probability distribution histogram of *MGMT_value* values, obtained after executing the *predict* function on the ***test*** data (*submission* dataframe). This *submission* dataframe was then converted into a *submission.csv* file for the final submission to obtain the *AUC score* of our model on the ***test*** dataset. The x-axis represents the range of *MGMT* presence probability, while the y-axis represents the count of *BraTS21_ID*s that fall within that range.

4.4 PERFORMANCE OF SWIN TRANSFORMER MODEL

After submitting the *iPython notebook* to the competition, the code ran privately for approximately three hours on a *GPU P100*. During this time, the *private* and *public AUC scores* were calculated. The *private score* is computed on approximately **78%** of the ***test*** data, while the *public score* is calculated on the remaining **22%**. The final leaderboard standings are determined solely based on the private score.

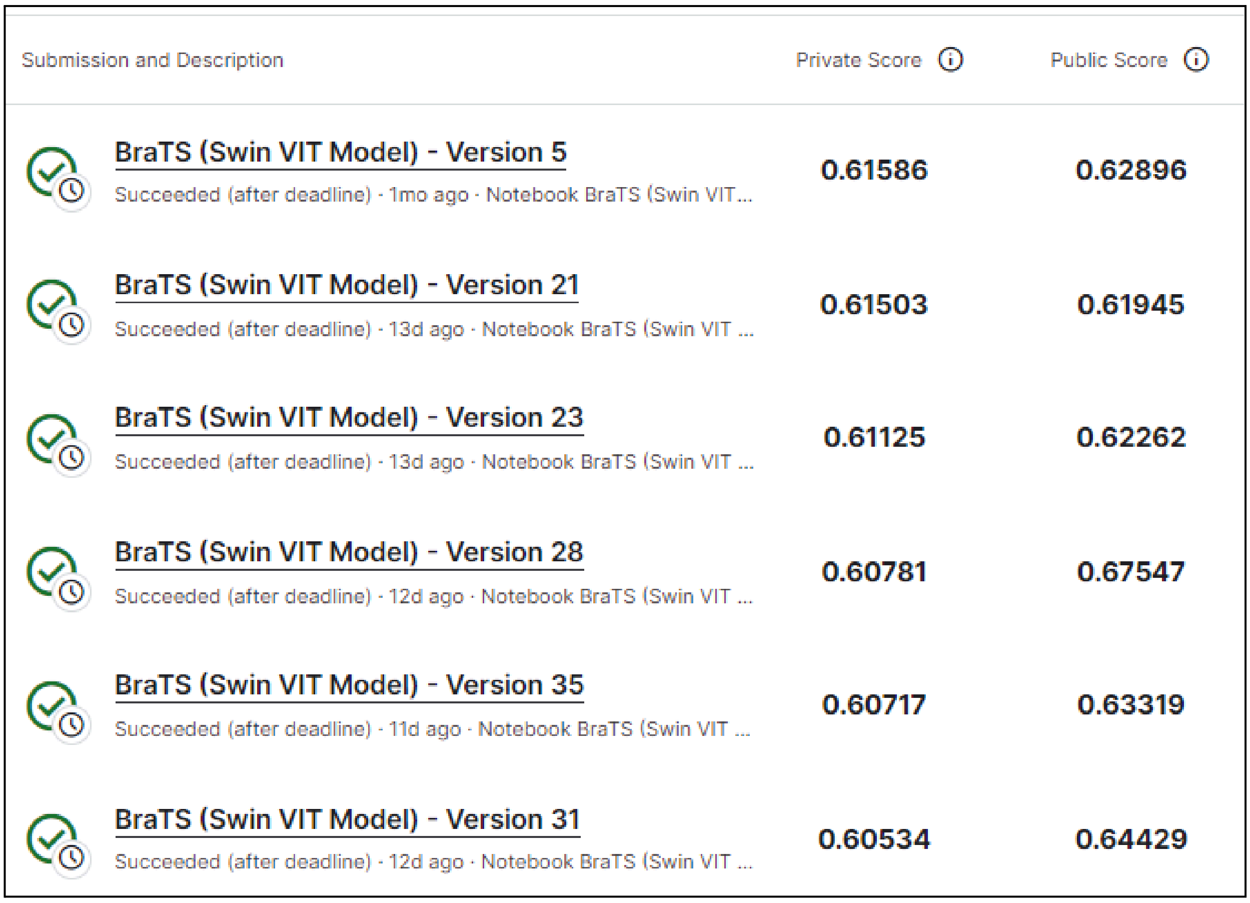
Figure 4.4.1 illustrates the *private AUC scores* (left column) and *public AUC scores* (right column) of the initial versions obtained upon submission of the codes to the competition. Initially, the submitted versions relied on a simple *Vision Transformer (ViT)* model, resulting in unsatisfactory *AUC scores.* However, upon transitioning to the *Swin Transformer* architecture, a significant improvement in performance was observed. Specifically, the *private score* increased from **0.44390** *BraTS (ViT Model) - version 2* to **0.58305** *BraTS (Swin ViT Model) - version 1.*

In the initial versions *BraTS (Swin VIT Model) - version 1, 2, 3*, the patch embedding layer employed only ***linear projection***, with adjustments made to the *learning rate*. Subsequently, transitioning to ***Conv2D*** in the patch embedding layer for *BraTS (Swin VIT Model) - version 4*, led to further enhancements, resulting in an improved *private score* of **0.59593**.



Submission and Description	Private Score	Public Score
BraTS (VIT Model) - Version 2 Succeeded (after deadline) · 1mo ago · BraTS (VIT Model) | Versio...	0.44390	0.43208
BraTS (Swin VIT Model) - Version 1 Succeeded (after deadline) · 1mo ago · Notebook BraTS (Swin VIT...	0.58305	0.56421
BraTS (Swin VIT Model) - Version 4 Succeeded (after deadline) · 1mo ago · Notebook BraTS (Swin VIT...	0.59593	0.58694

Fig 4.4.1 Initial submissions and their poor performance



Submission and Description	Private Score	Public Score
BraTS (Swin VIT Model) - Version 5 Succeeded (after deadline) · 1mo ago · Notebook BraTS (Swin VIT...	0.61586	0.62896
BraTS (Swin VIT Model) - Version 21 Succeeded (after deadline) · 13d ago · Notebook BraTS (Swin VIT ...	0.61503	0.61945
BraTS (Swin VIT Model) - Version 23 Succeeded (after deadline) · 13d ago · Notebook BraTS (Swin VIT ...	0.61125	0.62262
BraTS (Swin VIT Model) - Version 28 Succeeded (after deadline) · 12d ago · Notebook BraTS (Swin VIT ...	0.60781	0.67547
BraTS (Swin VIT Model) - Version 35 Succeeded (after deadline) · 11d ago · Notebook BraTS (Swin VIT ...	0.60717	0.63319
BraTS (Swin VIT Model) - Version 31 Succeeded (after deadline) · 12d ago · Notebook BraTS (Swin VIT ...	0.60534	0.64429

Fig 4.4.2 Submissions sorted based on *private AUC scores*

Figure 4.4.2 depicts the sorted list of best submissions based on the *private AUC scores*. Despite various parametric (*learning rate, number of epochs, patience, batch size, depth, dropout, and attention dropout*) changes and modifications to the layers of the *Swin Transformer* in multiple combinations, most attempts either led to a decrease in the score or showed no improvement.

BraTS ipynb version	***Optimizer***	***Learning rate***	***Epochs***	***Patience***	***Batch size***	***Depth***	***Patch embedding***
(VIT Model) - version 2	*Adam*	*0.001*	*10*	*10*	*4*	*2*	*Linear projection*
(Swin VIT Model) - version 1	*Adam*	*0.001*	*3*	*3*	*1*	*12*	*Linear projection*
(Swin VIT Model) - version 4	*Adam*	*0.001*	*5*	*5*	*2*	*12*	*Conv2D projection*
(Swin VIT Model) - version 5	*Adam*	*0.0005*	*10*	*10*	*4*	*16*	*Conv2D projection*
(Swin VIT Model) - version 21	*Adam*	*0.0003*	*15*	*10*	*4*	*16*	*Conv2D projection*
(Swin VIT Model) - version 23	*Adam*	*0.0002*	*25*	*10*	*4*	*16*	*Conv2D projection*

Fig 4.4.3 Parametric values of some of the important versions

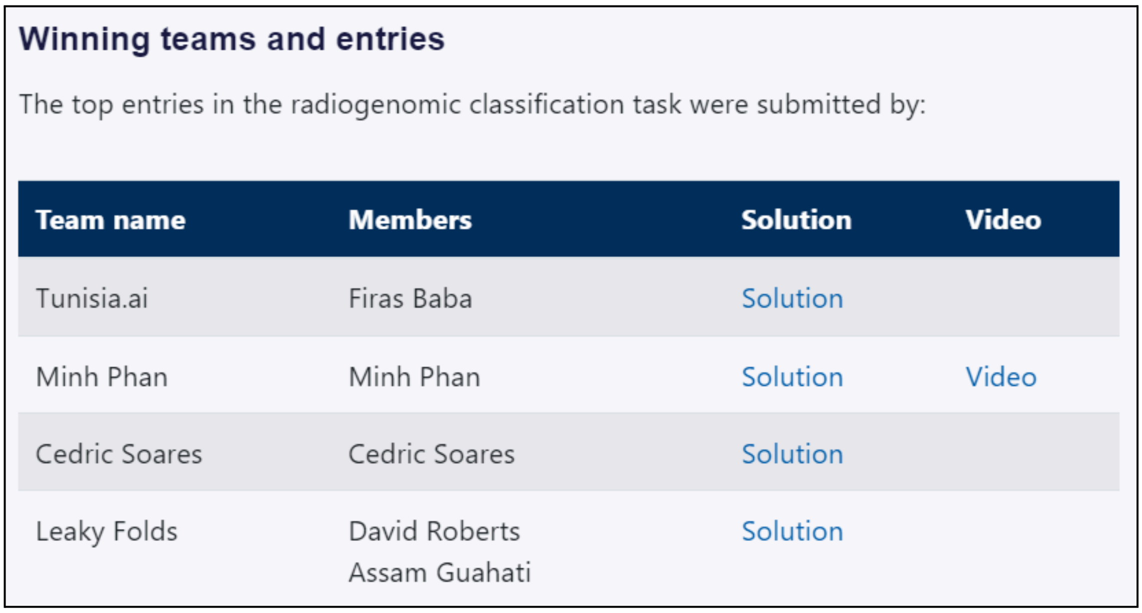
Adaptive Moment Estimation (Adam) emerged as the most effective learning rate optimization algorithm among all versions tested. While *Stochastic Gradient Descent (SGD)* and *Root Mean Square Propagation (RMSprop)* were also explored in alternate versions, they did not meet performance expectations. Notably, only the *Adam optimizer*, with a learning rate set at 0.0005, demonstrated superior performance compared to other versions.

Additionally, increasing the depth of the *Swin Transformer Block* from 12 to 16 layers led to some improvements. However, further increases in depth resulted in excessive resource and time consumption without yielding desirable outcomes.

Ultimately, only *BraTS (Swin VIT Model) - version 5* submission emerged as the best performer, achieving a *private score* of **0.61586** on the ***test*** dataset. This positioned my code as the ***fourth best model***, surpassing the score of **0.61562** achieved by the *Leaky Folds* team *(Figure 4.5.2)* among the 1556 submissions to the challenge.

4.5 COMPARISON WITH OTHER MODELS

Among the submitted solutions, the best performing model, implemented by the *[Tunisisa.ai]* team, employed an *ensemble 3D CNN Resnet10 model*, resulting in a *private AUC score* of **0.62174**. The second-best submission utilized an *EfficientNet-B0 pre-trained CNN* model for image feature extraction, followed by a *two-layered LSTM,* yielding a *private AUC score* of **0.61881**. The third-best submission incorporated the *EfficientNet-B3 model*, achieving an *AUC score* of **0.61732**. These models demonstrated relatively superior performance compared to my *Swin-T* model, which attained an *AUC score* of **0.61586**. *Figure 4.5.1* represents the winning solutions and *Figure 4.5.2* represents the *private AUC score* leaderboard of the challenge respectively.

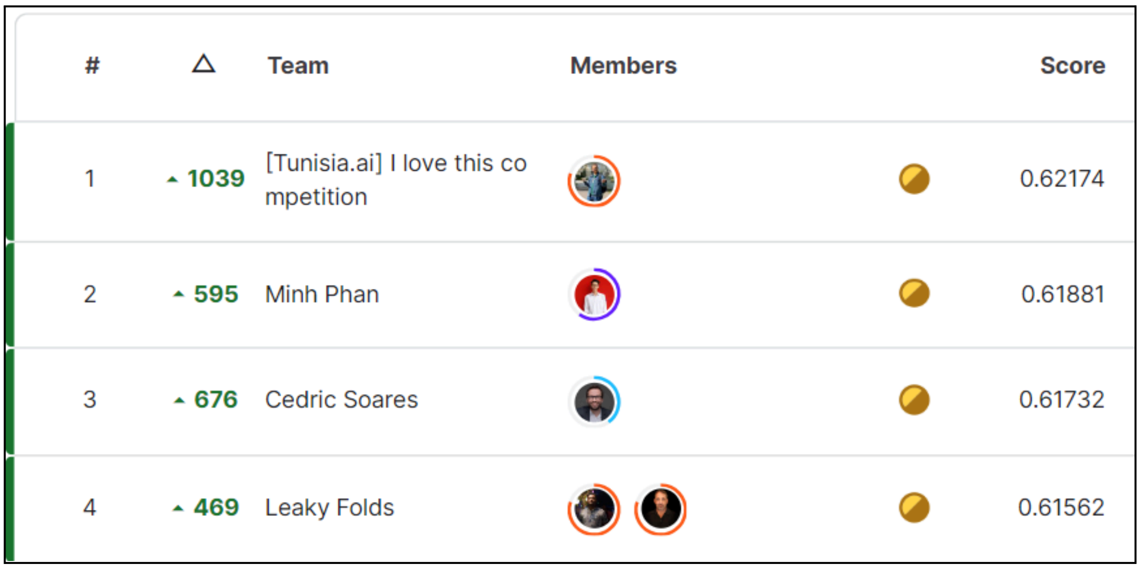


Winning teams and entries

The top entries in the radiogenomic classification task were submitted by:

Team name	Members	Solution	Video
Tunisia.ai	Firas Baba	Solution	
Minh Phan	Minh Phan	Solution	Video
Cedric Soares	Cedric Soares	Solution	
Leaky Folds	David Roberts Assam Guahati	Solution	

Fig 4.5.1 Winning solutions of the challenge [10]



#	△	Team	Members		Score
1	▲ 1039	[Tunisia.ai] I love this co mpetition		●	0.62174
2	▲ 595	Minh Phan		●	0.61881
3	▲ 676	Cedric Soares		●	0.61732
4	▲ 469	Leaky Folds		●	0.61562

Fig 4.5.2 *private AUC score* leaderboard of the challenge [10]

CHAPTER 5
CONCLUSION

5.1 INSIGHTS FROM THE RESULTS

The analysis presented in this study focused on the performance of an adapted version of *Swin Transforme*r model in the context of brain tumor radiogenomic classification. Through experimentation and evaluation, it was observed that certain models, such as the *3D CNN Resnet10* and the *EfficientNet-B0 pre-trained CNN* followed by a *two-layered LSTM*, outperformed in terms of *private AUC scores*. Despite the competitive landscape, our *adapted Swin Transformer* model also showcased promising results with a slightly *lower AUC score*. This underscores the potential of novel architectures like the *Swin Transformer* in advancing the field of radiogenomics and medical image analysis.

5.2 RECOMMENDATIONS FOR FUTURE RESEARCH

While our *adapted version of the Swin-T* model has successfully incorporated many aspects, such as its core architecture, certain features like *Hierarchical Structure* and a *Simplified Positional Embedding* have not been fully implemented and remain ripe for further development. To enhance the model's capabilities, it's recommended to explore these areas in future research endeavors.

Considering the insights gained from both our *Swin-T* model and the leading solutions in the competition, several avenues for future research emerge. Firstly, there's potential for enhancing the *Swin Transformer* architecture by fine-tuning *hyperparameters* and exploring alternative variations. Leveraging insights from the winning models, such as the *ensemble 3D CNN* approach, could inform strategies for improving the performance of our *Swin-T* model.

Ensemble learning techniques, as demonstrated by the top-performing solution, offer promise for enhancing model robustness and generalization capabilities. Exploring ensemble strategies, either by combining multiple instances of our *Swin-T* model or integrating it with other architectures, could lead to notable performance gains.

Furthermore, deeper investigation into the biological underpinnings linking imaging features to genetic subtypes presents an intriguing avenue for research. By unraveling these mechanisms, we could refine our predictive models and potentially identify novel biomarkers for more accurate classification.

Lastly, expanding the scope of classification by incorporating additional modalities, such as *functional MRI* [20] or *diffusion tensor imaging* [21], holds potential for enriching feature representation and improving classification accuracy. Integrating insights from diverse modalities could lead to a more comprehensive understanding of brain tumor pathology and facilitate more precise classification.

In conclusion, future research efforts should focus on refining the *Swin Transformer model,* exploring ensemble learning techniques, delving deeper into biological mechanisms, and incorporating additional modalities to advance the field of brain tumor radiogenomic classification in line with the competitive standards set by the recent competition.

REFERENCES

Research Papers and Articles

[1] M. A. Mazurowski, “Radiogenomics: What It Is and Why It Is Important,” *Journal of the American College of Radiology*, vol. 12, no. 8, pp. 862–866, Aug. 2015, doi: https://doi.org/10.1016/j.jacr.2015.04.019

[2] Q. Liu and P. Hu, “Extendable and explainable deep learning for pan-cancer radiogenomics research,” *Current Opinion in Chemical Biology*, vol. 66, p. 102111, Feb. 2022, doi: https://doi.org/10.1016/j.cbpa.2021.102111

[3] S. Bauer, R. Wiest, L.-P. Nolte, and M. Reyes, “A survey of MRI-based medical image analysis for brain tumor studies,” *Physics in Medicine and Biology*, vol. 58, no. 13, pp. R97–R129, Jun. 2013, doi: https://doi.org/10.1088/0031-9155/58/13/r97

[4] G. Boesch, “Vision Transformers (ViT) in Image Recognition - 2024 Guide,” *viso.ai*, Nov. 25, 2023. Available: https://viso.ai/deep-learning/vision-transformer-vit//. [Accessed: May 23, 2024]

[5] K. Han *et al.*, “A SUBMISSION TO IEEE TRANSACTION ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE 1 A Survey on Visual Transformer.” Available: https://arxiv.org/pdf/2012.12556

[6] F. Gaillard, “Methylguanine-DNA methyltransferase (MGMT) | Radiology Reference Article | Radiopaedia.org,” *Radiopaedia*. Available: https://radiopaedia.org/articles/methylguanine-dna-methyltransferase-mgmt. [Accessed: May 23, 2024]

[7] M. Weller *et al.*, “MGMT promoter methylation in malignant gliomas: ready for personalized medicine?,” *Nature Reviews Neurology*, vol. 6, no. 1, pp. 39–51, Dec. 2009, doi: https://doi.org/10.1038/nrneurol.2009.197

[8] A. Dosovitskiy *et al.*, “AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE.” Available: https://arxiv.org/pdf/2010.11929

[9] Z. Liu *et al.*, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows.” Available: https://arxiv.org/pdf/2103.14030

[10] “RSNA-MICCAI Brain Tumor Radiogenomic Classification,” *kaggle.com*. Available: https://www.kaggle.com/competitions/rsna-miccai-brain-tumor-radiogenomic-classification

[11] “Synapse | Sage Bionetworks Brain Tumor Segmentation (BraTS) Challenges” *www.synapse.org*. Available: https://www.synapse.org/#

[12] A. Pinto, S. Pereira, H. Correia, J. Oliveira, D. M. L. D. Rasteiro and C. A. Silva, "Brain Tumour Segmentation based on Extremely Randomized Forest with high-level features," 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Milan, Italy, 2015, pp. 3037-3040, doi: https://doi: 10.1109/EMBC.2015.7319032

[13] S. Bauer, L.-P. Nolte, and M. Reyes, “Fully Automatic Segmentation of Brain Tumor Images Using Support Vector Machine Classification in Combination with Hierarchical Conditional Random Field Regularization,” *Lecture Notes in Computer Science*, pp. 354–361, Sep. 2011, doi: https://doi.org/10.1007/978-3-642-23626-6_44

[14] U.Baid, et al., “The RSNA-ASNR-MICCAI BraTS 2021 Benchmark on Brain Tumor Segmentation and Radiogenomic Classification”, Available: https://arxiv.org/pdf/2107.02314, 2021

[15] A. Vaswani et al., “Attention Is All You Need,” Jun. 2017. Available: https://arxiv.org/pdf/1706.03762

[16] S. Ioffe, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” 2015. Available: https://arxiv.org/pdf/1502.03167

[17] “ImageNet,” www.image-net.org. Available: https://www.image-net.org/

[18] “Vision Transformer (ViT) from Scratch,” kaggle.com. Available: https://www.kaggle.com/code/raufmomin/vision-transformer-vit-from-scratch. [Accessed: May 23, 2024]

[19] “Swin-Transformer/models/swin_transformer.py at main · microsoft/Swin-Transformer,” GitHub. Available: https://github.com/microsoft/Swin-Transformer/blob/main/models/swin_transformer.py. [Accessed: May 23, 2024]

[20] “Magnetic Resonance, Functional (fMRI) - Brain,” Radiologyinfo.org, 2018. Available: https://www.radiologyinfo.org/en/info/fmribrain

[21] E. E. Sigmund, E. Furman-Haran, P. A. T. Baltzer, and S. C. Partridge, “Chapter 9 - Diffusion Tensor Imaging (DTI) of the Breast,” ScienceDirect, Jan. 01, 2023. Available: https://www.sciencedirect.com/science/article/abs/pii/B9780323797023000095. [Accessed: May 23, 2024]

BraTS Dataset

[22] RSNA MICCAI BraTS 2021 dataset. Retrieved from: https://www.kaggle.com/c/rsna-miccai-brain-tumor-radiogenomic-classification/data

[23] BraTS 2021 Proceedings. Retrieved from: https://www.med.upenn.edu/cbica/brats2021/proceedings.html

[24] U.Baid, et al., "The RSNA-ASNR-MICCAI BraTS 2021 Benchmark on Brain Tumor Segmentation and Radiogenomic Classification", arXiv:2107.02314, 2021(opens in a new window).

[25] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, et al. "The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)", IEEE Transactions on Medical Imaging 34(10), 1993-2024 (2015) DOI: 10.1109/TMI.2014.2377694 (opens in a new window)

[26] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J.S. Kirby, et al., "Advancing The Cancer Genome Atlas glioma MRI collections with expert segmentation labels and radiomic features", Nature Scientific Data, 4:170117 (2017) DOI: 10.1038/sdata.2017.117(opens in a new window)
[27] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, et al., "Segmentation Labels and Radiomic Features for the Pre-operative Scans of the TCGA-GBM collection", The Cancer Imaging Archive, 2017. DOI: 10.7937/K9/TCIA.2017.KLXWJJ1Q(opens in a new window)

[28] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, et al.,(opens in a new window) "Segmentation Labels and Radiomic Features for the Pre-operative Scans of the TCGA-LGG collection", The Cancer Imaging Archive, 2017. DOI: 10.7937/K9/TCIA.2017.GJQ7R0EF

Online Resources

[29] “Swin Transformer,” huggingface.co. Available: https://huggingface.co/docs/transformers/v4.40.2/en/model_doc/swin#transformers.SwinForImageClassification. [Accessed: May 23, 2024]

[30] Google, “Classification: ROC Curve and AUC | Machine Learning Crash Course,” Google Developers, 2019. Available: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

[31] R. Toshniwal, “Demystifying ROC Curves,” Medium, Jan. 21, 2020. Available: https://towardsdatascience.com/demystifying-roc-curves-df809474529a

[32] AI Coffee Break with Letitia, “An image is worth 16x16 words: ViT | Vision Transformer explained,” YouTube, Oct. 08, 2020. Available: https://www.youtube.com/watch?v=DVoHvmww2lQ&list=PLpZBeKTZRGPMddKHcsJAOIghV8MwzwQV6&index=1&ab_channel=AICoffeeBreakwithLetitia. [Accessed: May 23, 2024]

[33] AI Coffee Break with Letitia, “Swin Transformer paper animated and explained,” YouTube, Sep. 28, 2021. Available: https://www.youtube.com/watch?v=SndHALawoag&ab_channel=AICoffeeBreakwithLetitia. [Accessed: May 23, 2024]

[34] StatQuest with Josh Starmer, “ROC and AUC, Clearly Explained!,” YouTube, Jul. 11, 2019. Available:
https://www.youtube.com/watch?app=desktop&v=4jRBRDbJemM&ab_channel=StatQuestwithJoshStarmer. [Accessed: May 23, 2024]

[35] “Vision Transformer (ViT): Tutorial + Baseline,” kaggle.com. Available:
https://www.kaggle.com/code/abhinand05/vision-transformer-vit-tutorial-baseline#Vision-Transformers:-A-gentle-introduction. [Accessed: May 23, 2024]

[36] “Efficientnet3D with one MRI type,” kaggle.com. Available:
https://www.kaggle.com/code/rluethy/efficientnet3d-with-one-mri-type. [Accessed: May 23, 2024]