SAPIENZA
UNIVERSITÀ DI ROMA

nnAttention U-net for Brain Tumor Segmentation

Facoltà d'ingegneria dell'informazione, informatica e statistica
Computer Science - Informatica

Mattia Capparella
ID number 1746513

Advisor
Prof. Maria de Marsico

External advisor
Ilaria Gianoli

Academic Year 2020/2021

Thesis not yet defended

nnAttention U-net for Brain Tumor Segmentation
Tesi di Laurea Magistrale. Sapienza University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: capparella.1746513@studenti.uniroma1.it

To the infinity and beyond

Contents

1 Introduction 1

2 The Challenge 3

3 Fundamentals of magnetic resonance imaging 5
3.1 Medical Imaging . . . 5
3.2 Nuclear Magnetic Resonance . . . 5
3.3 Radio Frequency Signal Intensity . . . 6
3.4 The MR Image . . . 6
3.4.1 Spatial Characteristics . . . 6
3.4.2 Image Characteristics . . . 7
3.5 The Magnetic Field . . . 8
3.5.1 Tissue Magnetization and Resonance . . . 8
3.5.2 Gradients . . . 8
3.5.3 Nuclear Magnetic Interactions . . . 10

4 Related Work 13

5 SOTA Architectures and Procedures 15
5.1 SegResNet . . . 15
5.1.1 SegResNet Model . . . 16
5.1.2 VAE branch . . . 17
5.1.3 Loss, Optimization and Regularization . . . 18
5.1.4 Data preprocessing and augmentation . . . 18
5.1.5 Further Experiments and Results . . . 18
5.2 Attention U-Net . . . 19
5.2.1 Attention U-Net Model . . . 22
5.2.2 Loss, Optimization and Regularization . . . 23
5.2.3 Data preprocessing and augmentation . . . 23
5.2.4 Experiments and Results . . . 23
5.3 SwinUNETR . . . 23
5.3.1 UNETR . . . 27
5.3.2 Loss, Optimization and Regularization . . . 29
5.3.3 Data preprocessing and augmentation . . . 29
5.3.4 Experiments and Results . . . 30
5.3.5 Swin UNETR model . . . 32
5.3.6 Loss, Optimization and Regularization . . . 34

5.3.7 Data preprocessing and augmentation . . . 34
5.3.8 Experiments and Results . . . 34

6 Experimental Design and Result 37
6.1 Frameworks and Hardware Details . . . 37
6.2 Reproducing Previous Work . . . 38
6.3 Proposed Method . . . 38
6.4 Ablation Studies . . . 38
6.5 Results . . . 38

7 Conclusions and Future Work 39

Bibliography 41

Chapter 1

Introduction

Image segmentation (IS) is the process of partitioning an image into multiple image segments or objects. The goal of segmentation is to simplify the representation of an image into something that is more meaningful and easier to analyze. IS is a critical task in medical image analysis: it is often the first step to transform raw biomedical image into structured, valuable information ready to be used both for scientific discoveries and clinical applications including early diagnosis during preclinical phase, therapy planning, intraoperative assistance and tumor growth monitoring. Brain tumor segmentation is the process of isolating the tumor from healthy brain tissue; however, it is still a challenging task due to the irregular form and confusing boundaries of tumors.

Simply put, one major challenge is the lack of open datasets for designing and testing new algorithms, while private datasets may differ for so many aspects that comparing the result obtained by different solutions has no relevance and it's often inconclusive. On the other hand, with the availability of large common datasets, as the ones used in public challenges focused on medical images, more structured and comparable researches can be done. With the abundance of these data and the advent of the Vision Transformer (ViT) [11] architecture in late 2020, a new trend in the field of medical image segmentation has spread quickly: the proposed algorithms have become more complex both in term of capacity (estimated in number of parameters), sometimes neglecting simpler, yet promising solutions.

Being conscious of my hardware limitations, the impracticability of training very large models and even the difficulties of making continuous training sessions, I had to devise a clever way to train a "good enough model" in the most effective and reproducible way, while tracking the training processes in a fashion s.t. results of different experiments were easily comparable. For this reason I leveraged deep learning frameworks for professional AI researchers for code organization and accelerating research in Medical Imaging, with the intent to create a personal baseline to confront with, and flexibly expand my solution with new parts. Many official guides, as well as research papers present long training sessions (from hundreds to thousands epochs) and the few that show how the training process evolved, share the fact that the learning curves are very noisy for most of the time and only at the very end they reach a convergence point. The only exception seems to be the ViT based architectures, but they incur more easily into overfitting. What I tried to obtain was a flexible, relatively small model capable of learning without overfitting even with little data and in few epochs, so to have as quickly as possible a considerable amount of different "prototypes" to refine.

The work is organized as follow: ...

Chapter 2

The Challenge

The International Multimodal Brain Tumor Segmentation (BraTS) challenge is an event held in conjunction with the Medical Image Computing for Computer Assisted Intervention (MICCAI) conference. In it, prominent computational scientist and clinical researchers present their work on glioma, sclerosis and other brain injuries from every kind of point of view, but with a major focus on segmentation, prognosis and other applications for the clinical context.

Every year (from 2012), BraTS make publicy available a manually annotated dataset of preoperative brain tumor scans from different international institutions to assess the advances in the automated brain tumor segmentation task, using multiparametric MRI (mpMRI) scans. It is worth to be noted that from 2021, the challenge focuses also on a second task: the evaluation of classification methods to predict the MGMT promoter methylation status at preoperative baseline scans, since it has been identified as a strong and independent predictive factor of favorable survival in glioblastoma patients undergoing chemotherapy with alkylating agents.

Amongst brain tumors, glial tumors comprise 60% of the tumors. They are a common cause of mortality in both young and old people, with a substantial male dominance for the higher grades gliomas. Within the gliomas, Glioblastoma (GBM) and diffuse astrocytic glioma (WHO Grade 4 astrocytoma) are the most common and aggressive primary malignant brain tumors, comprising more than 16% of all primary brain and central nervous systems neoplasms. The typical survival range (prognosis) for these kind of tumors is about 1 year and the current standard of care treatment comprises surgery, followed by radiotherapy and chemotherapy. The use of MRI and diffusion tensor imaging (DTI) in preoperative planning, as well as ultrasound, CT scans, and MRI with direct stimulation during surgery, has allowed for multimodal neuronavigation and the integration of patient-specific anatomic and functional data. Despite these technologies, differentiating between normal brain and residual tumor continues to be a major challenge, and the use of an appropriate dye for fluorescence guidance has been found to be more effective then conventional neuronavigation-guided surgery alone.

The challenge requires to develop new method (or improve an existing one) being able to produce segmentation labels of the different glioma sub-regions considered for the evaluation: "enhancing tumor" (ET), "tumor core" (TC) and "whole tumor" (WT). ET is a region showing hyper-intensity in T1Gd w.r.t. healthy white matter, but also T1 modality; TC is the bulk entailing the ET and the necrotic (NCR) parts of the tumor, and it is the section that is typically resected: surgical removal of as much of a tumor as possible (tumor debulking) has the double benefit of enhancing the effects of chemo and radiation therapies and alleviating the pain due to the symptoms. The WT covers the entire extent of the tumor, entailing TC and the accumulation of fluid in the intracellular or extracellular spaces (cerebral edema, ED).

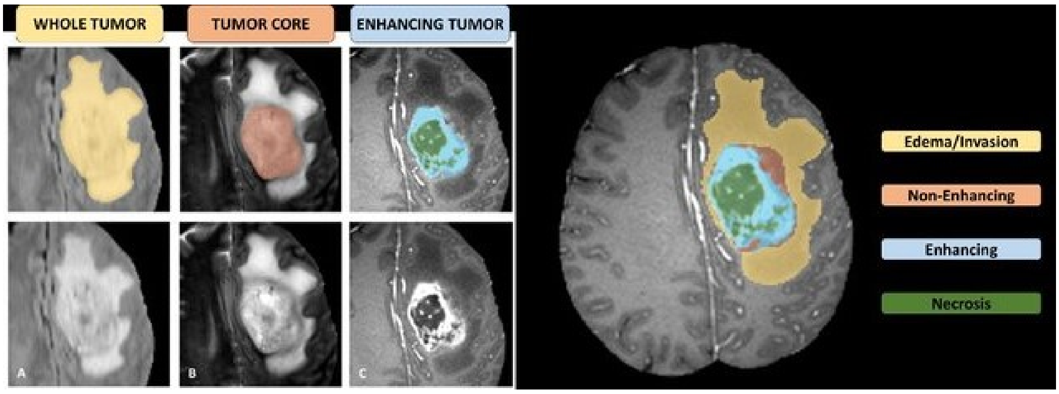


Figure 2.1. Tumor structures annotated in the different modalities (top left) and the final labels for the whole dataset (right). Image patches show from left to right: the whole tumor visible in FLAIR (A), the tumor core visible in T2 (B), the enhancing tumor structures visible in T1c (blue), surrounding the cystic/necrotic components of the core (green) (C). Segmentations are combined to generate the final labels of the tumor structures

Chapter 3

Fundamentals of magnetic resonance imaging

3.1 Medical Imaging

Medical Imaging (MI) is a process enabling the visual representation of the interior of a body for clinical analysis and medical intervention, establishes a common ground of normal anatomy and physiology to make it possible to identify abnormalities.

3.2 Nuclear Magnetic Resonance

Before deep dive into my work, let's have a small digression about the physics behind MRI in order to have an intuition on how MRI images are generated, and understand the the type of data I have worked on. Differently from computed tomography (TC) and PET scans, MRI does not use neither X-rays not ionizing radiation: it applies nuclear magnetic resonance (NMR), producing better contrast in images of soft-tissue, like the tissue in the brain.

The core idea behind NMR is to use powerful magnets to polarize and excite hydrogen nuclei of water molecules in human tissue, producing a detectable signal which result in images of the body. The specific physical characteristic of tissue visible in the image depends on how the magnetic field is being changed during the acquisition process, which consists of many repeated cycles. During these cycles the tissue magnetization is forced through a series of changes and its level that is present at the end of each cycle determines the intensity of the radio frequency (RF) signal produced and the resulting tissue brightness in the image. MR images are identified with specific tissue characteristic or blood conditions that are the predominant source of contrast; among the sources of contrast, "Magnetic characteristics of tissues" is the most common category and comprises the measurement of density of protons and the recovery process from tissue magnetization (see **Relaxation**). Images can be created in which either one of these characteristics is the predominant source of contrast: it is not usually possible to create images in which one of the tissue is the only pure source of contrast.

3.3 Radio Frequency Signal Intensity

The MRI process uses RF signals to transmit the image from the patient's body. The RF energy used is a form of non-ionizing radiation. The RF pulses are absorbed by the tissue and converted to heat and a small amount of energy is used to produce an image. The visual information that an MRI scan conveys clearly is the RF signal intensity emitted by the tissue. Bright areas correspond to tissues that emit high signal intensity and dark voids correspond to non-responding tissues. Between these two extremes resides a range of shades of gray showing contrast and differences among the tissues: when we look at an MR image, we are seeing a display of magnetized tissue, and it is the difference rates of change of magnetization level that produces much of the useful contrast at a specific "picture snapping time".

3.4 The MR Image

3.4.1 Spatial Characteristics

The MRI acquisition protocol allow the tuning of a various set of parameters to produce the appropriate spatial characteristics required by a specific clinical procedure. These characteristics include the number of slices, slice orientation and the structure within each individual slice.

Slices

A typical scan consists of a set of contiguous slices acquired simultaneously, the capacity of the set is limited by certain imaging factors and the amount of time spent for the acquisition process. The slices can be oriented in virtually any plane, but a set of oriented slices requires its own dedicated acquisition session. However, there is the possibility of acquiring 3D data from a large volume of tissue and then reconstructing slices in the different planes.

Voxels

In 3D computer graphics, a voxel represents a value on a regular grid in three-dimensional space. Each slice of tissue is subdivided into rows and columns of individual volume elements (voxel), whose size has a significant effect on image quality. Each voxel is an independent source of RF signals: this is why voxel size is a major consideration in each image acquisition.

Image Pixels

The image is also divided into rows and columns of picture elements (pixels). A pixel represent a corresponding voxel of tissue within the slice and its brightness is represent the intensity of the RF signal emitted by the tissue voxel.

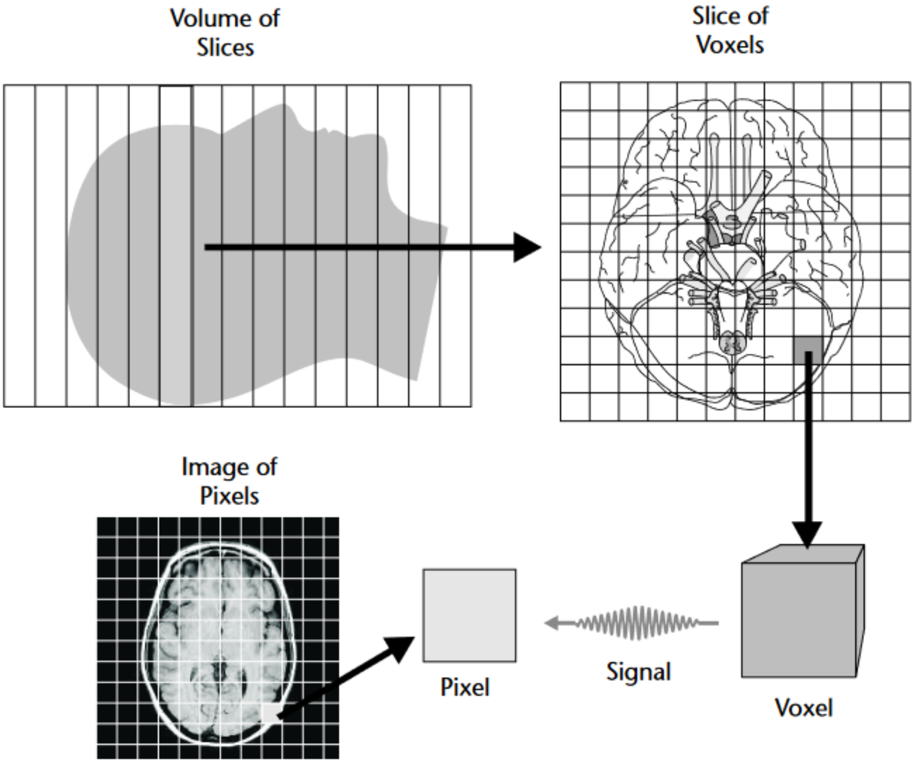


Figure 3.1. The spatial characteristics of MR images

3.4.2 Image Characteristics

Not all types of clinical procedure require images with the same characteristics, but the MRI system is powerful enough to allow for tremendous control over the specifics required and the overall image quality produced.

Contrast Sensitivity

Contrast sensitivity is the ability of an imaging process to produce an image of objects or tissues in the body that have relatively small physical differences or inherent contrast. The advantage of MRI w.r.t. other imaging techniques is that it has a high contrast sensitivity for visualizing differences among the tissues in the body because there are several sources of contrast: if a certain medical condition does not produce a visible change in one characteristic, there is the possibility that it will be visible in others.

Detail, Noise and Artifacts

The visibility of anatomical detail (spatial resolution) is limited by the blurring effect that occurs during the acquisition process. All medical imaging processes are affected by the blurring to some extent, but in MRI this is more accentuated. In addition to blurring, visual noise is a major issue in MRI: its presence limits the visibility of low contrast objects and differences among tissues. Most of the noise is the result of a form of random unwanted RF energy picked up from the patient's body, and the attempts to mitigate it involve necessarily some compromises with other characteristics. Artifacts are unwanted objects resulting in the processed image which do not represent an anatomical structure. They are produced by interactions or functions (such as movements) of the patient's body during the acquisition phase.

3.5 The Magnetic Field

The heart of the MRI system is a magnet that produces a strong, homogeneous magnetic field. The patent's body is placed inside the field during the acquisition procedure and it subjected to two distinct effects that combined create the image: tissue magnetization and tissue resonance.

3.5.1 Tissue Magnetization and Resonance

When immersed in a magnetic field, the tissue becomes temporarily magnetized by the alignment of the protons. The ability of MRI to distinguish between different types of tissue is based on the fact that different tissues, both normal and pathologic, will become magnetized to different levels or relax at different rates. The magnetic field also causes certain nuclei in the tissue to resonate in the RF range. The tissue then serves as both a radio receiver and transmitter during the imaging process.

3.5.2 Gradients

When the MRI system is in a resting state, the magnetic field is quite homogeneous over the region of the patient's body. To acquire data, the field must be distorted with gradients, *i.e.* changes in field strength from one point to another in the body. These changes are produced by a set of coils that are turned on and off accordingly to the acquisition procedure. In a MRI system there are typically three sets of gradient coils oriented so that gradients can be produced in the three orthogonal directions (x, y, z). Also, two or more coils can be used to produce a gradient in any direction. The three basic designs of coil are body, head and surface coils: surface one is used to receive signals from a small anatomical region to produce better image quality than is possible with the other designs.

Magnetic Direction

There are two principle directions that tissue is magnetized during the imaging process. Longitudinal magnetization is then the tissue is magnetized in a direction parallel to the direction of field. Transverse magnetization is when the direction of tissue magnetization is at a 90° angle w.r.t. the direction of the magnetic field and is in the transverse plane. Note that the actual direction of magnetization is not limited to longitudinal or transverse: it can have both components and have distinct characteristics that must be considered independently.

Magnetic Flipping

When a 90° pulse is applied to the longitudinal magnetization, it is reduced to zero (*saturation*) and flipped into the transverse plane, producing transverse magnetization: an *excited* condition (see **Excitation**).

Longitudinal Magnetization And Relaxation

When the magnetization is redirected by an RF pulse, it will recover its original orientation over a period of time (*relaxation time*). The convention is to specify the relaxation time in terms of the time required for the magnetization to reach the 63% of its maximum (value used for mathematical considerations and practical purposes). This time, the *longitudinal relaxation time* is named T1. Each tissue has its own T1 and the relevant aspect is that tissues with short T1 will have the highest level of magnetization (*i.e.* will be the brightest) in T1-weighted images when the picture is snapped during the relaxation period.

Transverse Magnetization And Relaxation

After the conversion to transversal magnetization, the excited condition quickly decay accordingly to a specific relaxation time named T2. Different tissues have different T2 values and the level of magnetization at snapping time will be used again as a source of contrast in the MR image. Transverse magnetization aside from contrast generation has also the purpose of generating the RF signals emitted by the tissue: each imaging cycle must conclude with transverse magnetization to produce the RF signal to form the image. Since Transverse Magnetization relaxation is actually a decay, T2 is the time required for 63% of the initial magnetization to dissipate. In general, this is the reason for a T2-weighted image appear to be a reversal of its T1-weighted version. The two factors that contribute to the dephasing of the nuclei, hence to the T2 relaxation, are the *spin-spin interactions* among the nuclei, and the inhomogeneity of the magnetic field. As a consequence, the true relaxation characteristics of a tissue are masked and the real relaxation time is named T2*, resulting much less then T2, and the transverse magnetization disappears before T2 contrast can be formed and sensed. The spin echo process is used to compensate the rapid relaxation time: a 180° pulse is applied to the tissue and the protons are flipped around an axis in the transverse plane; this cause an inversion of direction of rotation and make the fast spinning protons realigning with the slower ones, producing again a transverse magnetization (*echo event*) that builds up to a level dictated by T2 characteristics of the tissue.

Inversion Recovery and FLAIR MR imaging

Following the same principle of the spin echo process applied for the acquisition of T2 contrast, inversion recovery (IR) is a spin echo method used for specific purposes, such as obtain a high level of T1 contrast, suppress the signals of fat and fluids. This method applies an additional 180° pulse to the conventional spin echo sequence, inverting the direction of the longitudinal magnetization. The recovery of the magnetization level starts from a negative value and an additional time interval

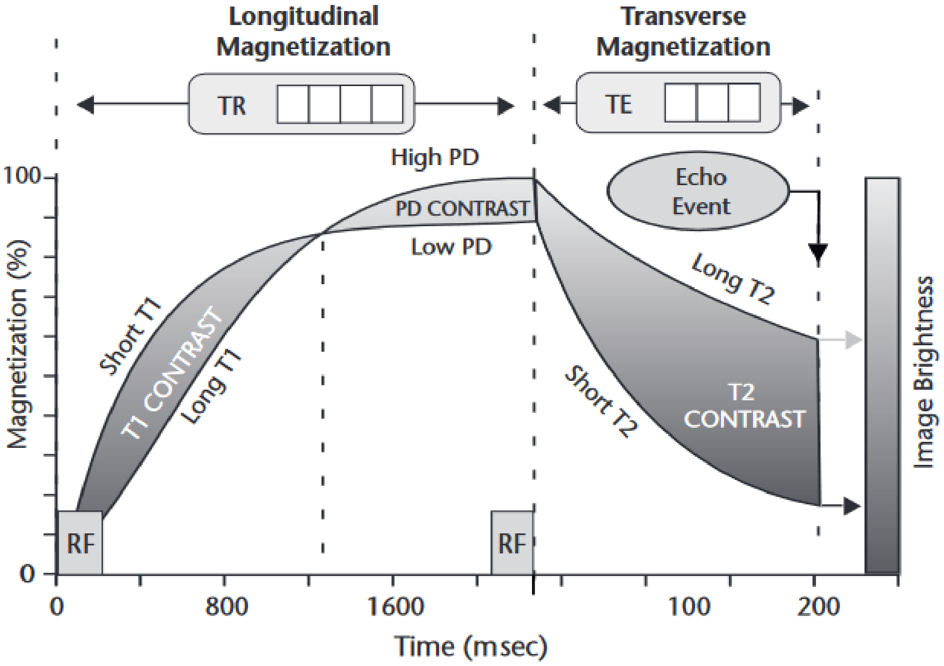
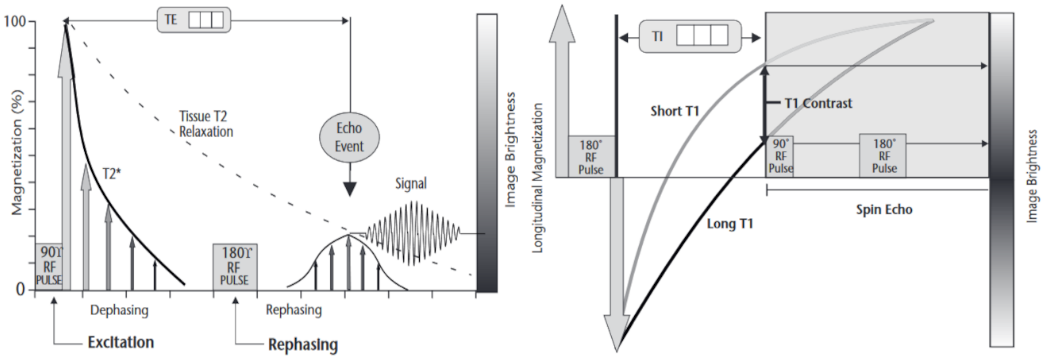


Figure 3.2. The longitudinal and transverse magnetization phases of an imaging cycle. T1 and PD contrast are produced during the longitudinal phase and T2 contrast is produced during the transverse phase

is associated with the inversion recovery pulse sequence: it is the time between the initial 180° pulse and the 90° pulse that zeros the longitudinal magnetization. This time is named Time after Inversion (TI) and can be used as a contrast control tool. The fluid-attenuated inversion recovery (FLAI) is an IR process in the spin echo methods family to null fluids in the image acquisition: it can be used in brain imaging to suppress cerebrospinal fluid (CSF). On T2 weighted images, astrocytoma and CSF have extensive areas of fairly homogeneous high signal. On T2-FLAIR, instead, the majority of these areas become relatively hypointense in signal due to incomplete suppression and, at the margins of the tumour, a rim of hyperintensity is usually seen.

3.5.3 Nuclear Magnetic Interactions

A magnetic nucleus is characterized by a magnetic moment. In the absence of a strong magnetic field, the moments are randomly oriented in space. When the magnetic nuclei are placed in a magnetic field, they experience a torque that encourages them to align with the direction of the field. In a human body, the number of nuclei capable of this alignment is proportional to the field strength. Once the nucleus is aligned, its nuclear magnetic moment precesses about the axis of the magnetic field. It is this wobbling motion that makes a nucleus sensitive and receptive to incoming RF energy when the RF frequency matches the precession rate (see **Larmor Frequency**).



(a) The spin echo process showing the use of a 180° pulse to rephase the protons and to prodecuce the echo event.

(b) The inversion recovery method with TI set to produce an image with high T1 contrast.

Excitation

In MRI a RF pulse is used that flips some of the nuclei alignment away from the direction of magnetic field, into its transverse plane. This motion places the nuclei in an *excited* state. In this state, the precession is transformed into a motion around the axis of the field and produces the RF signal that is collected to form the MR image.

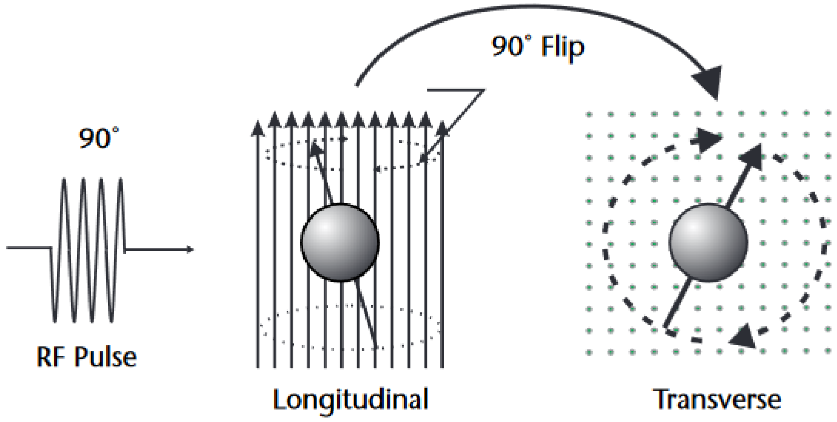


Figure 3.4. The excitation of a magnetic nucleus by the application of a pulse of RF energy.

Relaxation

As already mentioned, to acquire a full MR image a patient must undergo several cycles of MR. During these cycles, the tissue magnetization is flipped into an unstable condition and then it is allowed to recover: this recovery process is known as *relaxation*. The time required to "relax" depends on the physical characteristic of the tissue and can then be used to distinguish among normal and pathological tissues. Each tissue is characterized by two relaxation times: T1 and T2 that may be chosen to be the predominant source of contrast in a particular scan, which is named after the chosen time.

Larmor Frequency

The resonant frequency (*Larmor frequency*) of a nucleus is determined by a combination of nuclear characteristics and the strength of the magnetic field. The fact that different nuclides have different resonant frequencies means that most procedures can "tune in" with only one chemical element at a time.

Table 3.1. Larmor Frequencies for Selected Nuclides in a Magnetic Field of 1 T.

Nuclide	**Larmor Frequency (MHz)**
Hydrogen-1	42.58
Fluorine-19	40.05
Phosporus-31	17.24
Sodium-23	11.26

Chapter 4

Related Work

In 2018, top performing submission included [27] and [17]. The first proposed a semantic segmentation network for tumor subregions based on encoder-decoder architecture equipped with a variational autoencoder branch to regularize the shared decoder due to the limited size of training dataset; while the second demonstrated that a generic U-Net architecture with a few modifications such as region based training, additional training data and a combination of loss functions was enough to achieve competitive performance. In 2019, [20] devised a novel two-stage cascaded U-Net to segment the subregions of brain tumor from coarse to fine: in the first stage a variant of U-Net is used to produce a coarse prediction; in the second stage, a couple of decoders is used to refine the prediction map by concatenating a preliminary prediction map with the original input used for auto-context. For this training no additional data were used. In 2021, [13] proposed jointly an optimized U-Net architecture, a learning scheduling and a post-processing strategy to achieve the best possible results. In 2022, [36] proposed a new ensemble of multiple deep learning frameworks including: DeepSeg, nnU-net and DeepSCAN. All the three methods follow the encoder-decoder architecture, but the first is trained only on two-dimensional T2 FLAIR scans and the third is contains densely connected blocks of dilated convolution. My work was heavily inspired by [28], where is proposed a novel attention gate (AG) that automatically learns to focus on target structures of varying shapes and sizes. Before even starting my experimentation I acknowledged the difficulty for such method to perform well on this task, given the highly irregular structure of the glioma subregions and the possibility for a tumor to develop in different regions of the brain. Nonetheless, with minor modifications in the architecture and taking inspiration from [13] and [18] for training and post-processing strategies, I managed to achieve good results for the segmentation task. In order to do a fair comparison and since the fact that year after year the BraTS dataset was expanded for the challenge with new scans, I trained and validated the top performing architectures on the very same version of the dataset: the Medical Segmentation Decathlon (MSD) dataset provided by [2].

Chapter 5

SOTA Architectures and Procedures

To select the optimal neural network to work with, I conducted several ablation studies on the following networks: SegResNet and SegResNet-VAE [27], Attention U-Net [28] and SwinUNETR [14]. Here follows a brief description for each architecture and the methods used for the training.

5.1 SegResNet

ResNet

As we design new and deeper networks it becomes necessary to understand how adding layer can increase the expressiveness of the network, rather than simply increasing the complexity (always in terms of number of trainable parameters), and if we are creating something that is just different rather than better or, at least, as powerful as the previous networks. Consider $\mathcal{F}$ - the class of functions that a specific network architecture can reach - then there will be different $f \in \mathcal{F}$, all with their own set of parameters learnt during a training phase, trying to approximate at their best f^*, *i.e.* the "truth function" we would like to find. Our goal is to find some $f^*_{\mathcal{F}}$ which is our best within $\mathcal{F}$. While it is reasonable to assume that designing a more powerful class $\mathcal{F}'$ we should arrive at a better result, it is not guaranteed: if $\mathcal{F} \not\subseteq \mathcal{F}'$ then $f^*_{\mathcal{F}'}$ might be worse than $f^*_{\mathcal{F}}$. To be guaranteed that by changing the class of architecture we consider, we must assure that we work with nested function classes, *i.e.* the larger function classes must contain the smaller ones. To achieve this condition the new part of the neural network should be able to be trained into an identity function $f(x) = x$ so that the new model will be as effective as the original model. With this intuition in mind, [16] proposed the *residual network (ResNet)* along with its fundamental block, *i.e.* the *residual block* that deeply influenced the design of deep neural networks in the years to come.

Residual Block

Let $\mathbf{x}$ be the input and $f(\mathbf{x})$ the underlying mapping we want to obtain. On the left of Fig. 5.1, the portion within the dotted-line box must directly learn the mapping

$f(\mathbf{x})$. On the right, the portion within the dotted-line box needs to learn the *residual mapping* $f(\mathbf{x})-\mathbf{x}$, which is how the residual mapping is easier to learn. The solid line carrying the layer input $\mathbf{x}$ to the addition operator is called a *residual connection* (or *skip connection*). With residual blocks, inputs can forward propagate faster through the residual connections across layers.

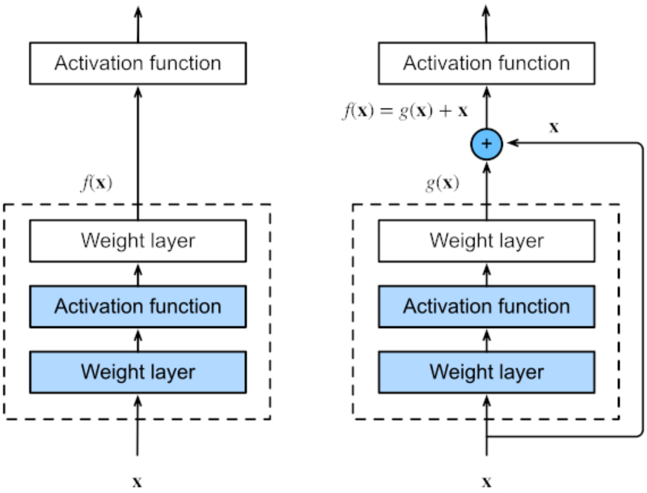


Figure 5.1. Regular block (left). Residual block (right)

Short and Long skip connections in Deep Learning

The core idea behind skip connections is to backpropagate through the identity function, by just using a vector addition. Then the gradient would simply be multiplied by one and its value will be maintained in the earlier layers. Apart the problem of vanishing gradients, the reason these kind of connections are used it that for many tasks (semantic segmentation included) there is some lower semantic information that is extracted in the initial layers and it is desirable to push it forward so that future layers can learn from it. In practical terms, when using additive skip connections, the dimensionality of the feature maps has to be same: this is reason why they are used in two kinds of setups - *short* and *long* skip connections. Short skip connections are used with consecutive convolutional layers that do not change the input dimension. Long skip connections are used within typically symmetrical architectures (*e.g.* encoder-decoder architecture) where the spatial dimensionality is reduced in the encoder part and is gradually increased in the decoder part.

5.1.1 SegResNet Model

The segmentation approach follows an asymmetrical encoder-decoder based CNN architecture, with a larger encoder for feature extraction and a smaller decoder for segmentation mask reconstruction. The encoder part uses ResNet blocks made of two convolutions with Group Normalization (GN) and ReLU, followed by additive identity skip connection. GN is used instead of Batch Normalization (BN) since it showed better performance when batch size is small (in this case is 1). At each downsample step, the image dimensions are halved and simultaneously the feature size is doubled. All the convolutions are $3 \times 3 \times 3$ with initial number of filters equal to 32, while the encoder endpoint has 256 filters and the resulting map is 8 times spatially smaller than the input image. No additional downsizing was performed to preserve more spatial content. The decoder has a single block per each spatial level and the first operation performed is upsizing: the number of features are progressively halved while the spatial dimension are doubled using *non trainable* 3D bilinear upsampling algorithm. Then, the upsampled data is concatenated by addition with the encoder output of the equivalent spatial level. The end of the decoding path restores the both the original spatial size and number of features and it is followed by a $1 \times 1 \times 1$ convolution into 3 channels (predictions for each tumor subregion) and a sigmoid function as the final activation.

5.1.2 VAE branch

An additional branch is added to the encoder endpoint to reconstruct the original image, similar to auto-encoder architecture. The branch is added to guarantee additional guidance and regularization to the encoder part, since the training dataset is limited. From the encoder endpoint output, the input is reduced to a low dimensional space of 256 (128 to represent the mean μ and 128 to represent std σ^2). Then a sample is drawn from the Gaussian distribution with the given parameters and reconstructed into the input image dimensions following the same decoder architecture, except there are no skip connections from the encoder.

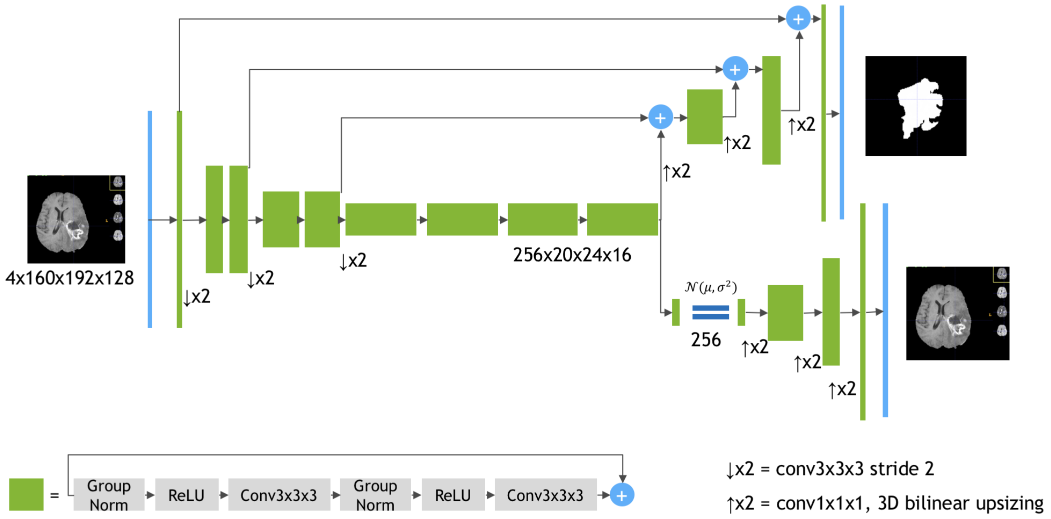


Figure 5.2. Schematic visualization of the Myronenko's SegResNet-VAE architecture for the BraTS task. The *top-right* head has three channels for the segmentation maps of the three tumor subregions (WT, TC, ET). The *bottom-right head* is the VAE that reconstructs the input image into itself, used at training time to regularize the shared encoder.

5.1.3 Loss, Optimization and Regularization

In the original paper the loss function consisted of 3 terms:

$$\mathbf{L} = \mathbf{L}_{dice} + 0.1 * \mathbf{L}_{\mathrm{L2}} + 0.1 * \mathbf{L}_{\mathrm{KL}} \tag{5.1}$$

$\mathbf{L}_{dice}$ is a soft dice loss [26] applied to the decoder output p_{pred} to match the segmentation mask p_{true}:

$$\mathbf{L}_{dice} = \frac{2 * \sum p_{true} * p_{pred}}{\sum p_{true}^2 + \sum p_{pred}^2 + \epsilon} \tag{5.2}$$

Since the output of the segmentation decoder has 3 channels, the losses are simply added together.

$\mathbf{L}_{\mathrm{L2}}$ is an L2 loss on the VAE branch output I_{pred} to match the input image I_{input}:

$$\mathbf{L}_{\mathrm{L2}} = \| I_{input} - I_{pred} \|_2^2 \tag{5.3}$$

$\mathbf{L}_{\mathrm{KL}}$ is standard VAE penalty term, a KL divergence between the estimated normal distribution $\mathcal{N}(\mu, \sigma^2)$ and a prior distribution $\mathcal{N}(0, 1)$, whose closed formed representation is:

$$\mathbf{L}_{\mathrm{KL}} = \frac{1}{N} \sum \mu^2 + \sigma^2 - log\sigma^2 - 1 \tag{5.4}$$

where N is the number of image voxels. The hyperparameter weight to balance the 3 terms of Equation 5.1 was found after empirical experimentation.

An Adam optimizer was used with initial learning rate of $\alpha_0 = 1e{-}4$, progressively decreased according to:

$$\alpha = \alpha_0 * (1 - \frac{e}{N_e})^{0.9} \tag{5.5}$$

where e is an epoch counter and N_e is the total number of epochs (300 in this case). During training phase a batch size of 1 was used, along with random order sampling.

They used L2 norm regularization on the convolutional kernel parameters with a weight of $1e - 5$ and spatial dropout with a rate of 0.2 after initial convolution. Some experiments focused on other dropout placements (even after each convolution) but did not find any additional accuracy improvements.

5.1.4 Data preprocessing and augmentation

The images were normalized to have zero mean and unit std based on *non-zero voxels* only. They also applied a random, channel-wise, intensity shift (-0.1..0.1 of image std) and scale (0.9..1.1) on input image channels and with a probability of 0.5 an axis mirror flip (for all 3 axes) was performed.

5.1.5 Further Experiments and Results

Before getting to their final model, the author experimented several network architectures: they tried a larger batch size of 8 to use BatchNorm, but this forced to use a small image crop size leading to worse results. They also tried more data augmentation techniques such as histogram matching, affine transformations and random filtering but they did not show additional improvements. Post-processing techniques were tried to produce finer segmentation mask, but they did not reveal beneficial. Increasing the network depth also was inconclusive, and the only consisting improvement came from using the NVIDIA Volta V100 32GB GPU (instead of the 16GB version) that allowed to double the number of features per layer.

The network was eventually trained from scratch on NVIDIA Tesla V100 32GB GPU, using BraTS 201 training dataset (285 cases) without any additional in-house data and a random crop of size 160 × 192 × 128. Here are reported the results of their approach on BraTS 2018 validation (66 cases) 5.1 and test (191 cases) 5.2 sets. Aside from evaluating a single model, they also ensambled a set of 10 models trained from scratch, resulting in a 1% improvement. Time-wise, each training epoch on a single GPU takes 9 min. Training the model for 300 epochs takes 2 days.

Table 5.1. BraTS 2018 validation dataset results. Mean Dice and Hausdorff measurements of the proposed segmentation method. EN - enhancing tumor core, WT - whole tumor, TC - tumor core

	Dice			Hausdorff (mm)		
Tumor sub-region	**ET**	**WT**	**TC**	**ET**	**WT**	**TC**
Single model	0.8145	0.9042	0.8596	3.8048	4.4834	8.2777
Ensamble of 10 models	0.8233	0.9100	0.8668	3.9257	4.5160	6.8545

Table 5.2. BraTS 2018 testing dataset results. Mean Dice and Hausdorff measurements of the proposed segmentation method. EN - enhancing tumor core, WT - whole tumor, TC - tumor core.

	Dice			Hausdorff (mm)		
Tumor sub-region	**ET**	**WT**	**TC**	**ET**	**WT**	**TC**
Ensamble of 10 models	0.7664	0.8839	0.8154	3.7731	5.9044	4.8091

5.2 Attention U-Net

U-Net

U-Net is an architecture for semantic segmentation consisting of a contracting path and an expansive path. The first path follows the typical architecture of a convolutional neural network (CNN) comprising of convolutions, rectified linear unit (ReLU) activations and max pooling operations with stride 2 for downsampling. Here, the number of feature channels is doubled at each downsampling step. In the second path, at each step the feature map is upsampled and the number of feature channels is halved, then combined with the corresponding feature map in the contracting path (see *skip connections*), convoluted and activated again with the ReLU. The final layer, a 1 × 1 convolution, is used to map each feature vector to the desired number of classes. Additionally, deep-supervision can be added, which is computing loss functions for outputs from lower decoder levels.

Attention Gate

Introduced in [28], *Attention Gates* (AGs) focus on targeted structures of varying shapes and size while suppressing feature activations in irrelevant regions for a specific task. Given the input feature map $\mathcal{X}$ and the gating signal $\mathcal{G} \in \mathbb{R}^{C' \times H \times W}$ which is collected at a coarse scale and contains contextual information, the attention gate uses additive attention to obtain the *gating coefficient*. Both the input $\mathcal{X}$ and the gating signal $\mathcal{G}$ are first linearly mapped to an $\mathbb{R}^{F \times H \times W}$ dimensional space and then the output is squeezed in the channel domain to produce a spatial attention weight map $S \in \mathbb{R}^{1 \times H \times W}$. The overall process can be written as:

$$S = \sigma(\varphi(\delta(\phi_x(X) + \phi_g(G)))) \tag{5.6}$$

$$Y = SX \tag{5.7}$$

where φ, ϕ_x and ϕ_g are linear transformations implemented as 1×1 convolutions.

The AG has a such lightweight design that its ability to guide model's attention to relevant regions comes without a significant increase in computing cost and number of parameters. In this way, the necessity of using an external localisation model can be eliminated while maintaining high prediction accuracy. Similar attention mechanism have been proposed for natural image classification [19] and captioning [1] to perform adaptive feature pooling, where model predictions are conditioned only on a subset of selected image regions. But in this paper this design is generalized and includes an *image-grid based gating* mechanism allowing attention coefficients to be specific to local regions a ready to be used for dense label predictions.

Attention Gate for Image Analysis

To sufficiently enlarge the receptive field, the feature-map grid is downsampled multiple times throughout the CNN architecture. As a result, the features on the coarse spatial grid level model both location and global relationship between tissues. On the other hand, it is harder to reduce false-positive predictions for small objects that show large shape variability and to improve the accuracy many segmentation framework rely on additional object localisation models so that localization and segmentation become two consecutive but separate steps. However it is showed that the integration of AGs in a standard CNN model is sufficient to achieve the same objective without the need of training multiple models.

Attention coefficients, $\alpha_i \in [0, 1]$ identify salient image regions and suppress feature responses to preserve only the activated area relevant for the task. The output of AGs is the element-wise multiplication of input feature-maps and attention coefficients: $\hat{x}^l_{i,c} = x^l_{i,c} \cdot \alpha^l_i$. In a default setting, a single scalar attention value is computed for each pixel vector $x^l_i \in \mathbb{R}^{F_l}$ where F_l corresponds to the number of feature-maps in layer l. In case of multiple semantic classes, they propose to learn multi-dimensional attention coefficients. Thus, each AGs learns to focus on a subset of target structures. The authors suggest to use additive attention to obtain the gating coefficient: although computationally more expensive, it has experimentally shown to achieve higher accuracy than multiplicative attention [25]. Additive attention is formulated as:

$$q^l_{att} = \psi^T(\sigma_1(W^T_x x^l_i + W^T_g g_i + b_g)) + b_\psi \tag{5.8}$$

$$\alpha_i^l = \sigma_2(q_{att}^l(x_i^l, g_i; \Theta_{att})) \tag{5.9}$$

where σ_1, σ_2 correspond to ReLU and sigmoid activation functions respectively. Θ_{att} is a set of parameters containing: linear transformations $W_x \in \mathbb{R}^{F_l \times F_{int}}, W_g \in \mathbb{R}^{F_g \times F_{int}}, \psi \in \mathbb{R}^{F_{int} \times 1}$ and bias term $b_\psi \in \mathbb{R}, b_g \in \mathbb{R}^{F_{int}}$. Note that in image captioning and classification, the activation function used to normalize the attention coefficients (σ_2) is often the softmax function; however, sequential use of softmax yields sparser activation at the output: for this reason a sigmoid is chosen instead. The gating signal proposed in this paper is not a *global single vector* for all image pixels but a grid signal conditioned to image spatial information. For each skip connection, the signal aggregates information from multiple image scales, increasing the grid-resolution of the query signal (coming from below level).

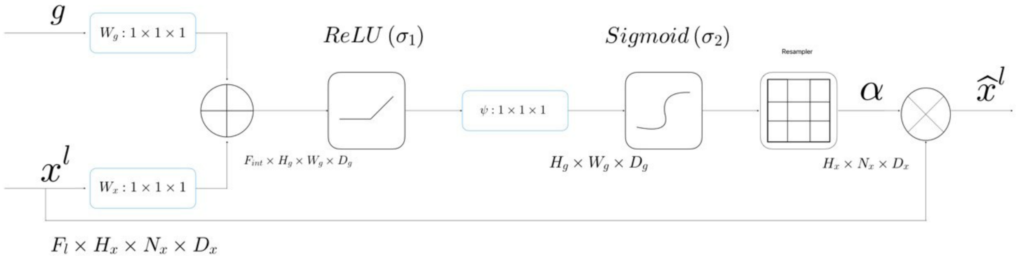

g
$W_g : 1 \times 1 \times 1$
$ReLU\,(\sigma_1)$
$Sigmoid\,(\sigma_2)$
Resampler
α
$\widehat{x}^l$
$\psi : 1 \times 1 \times 1$
x^l
$F_{int} \times H_g \times W_g \times D_g$
$H_g \times W_g \times D_g$
$H_x \times N_x \times D_x$
$W_x : 1 \times 1 \times 1$
$F_l \times H_x \times N_x \times D_x$

Figure 5.3. Schematic visualization of the Oktay's Additive Attention Gate. Input features $\mathbf{x}^l$ are weighted by attention coefficients α computed in the AG. Spatial regions are selected by analysing both the activations and contextual information extracted from the *gating signal* $\boldsymbol{g}$ collected at coarser scale.

Deep Supervision

During the training phase the authors leveraged the deep-supervision technique [22], [23] to force the intermediate feature-maps to be semantically discriminative at each image scale. At different scales, this help to ensure that attention units responses are influenced by a large range of image foreground content. The intuition behind the deep-supervision mechanism comes from a simple observation: a discriminative classifier can achieve better performance if trained on highly discriminative features rather than less discriminative features. Thus, if these features are the hidden layer feature maps of the network, the overall performance of the classifier can serve as a proxy of both the hidden and upper maps quality and we can use it in the feedback loop to properly update the hidden layer filters to favor highly discriminative power. When the proxy for the maps quality is good, we expect to learn good features with much more easy than if we only relied on the backpropagation from the final output layer alone. A deeply-supervised net (DSN) enforces direct and early supervision for both the hidden and output layer by introducing *companion objective* to the individual hidden layers, which is used as an additional constraint to the learning process.

5.2.1 Attention U-Net Model

The segmentation approach follows a symmetrical encoder-decoder based CNN architecture, but in contrast to the SOTA (2018) CNN segmentation frameworks, it is proposed a 3D-model to capture sufficient semantic context.

The input image is progressively filtered and downsampled by a factor of 2 at each scale in the encoding part. A downsampling stage consists of a convolution block comprising two convolutions, each one equipped with a Batch 3D Normalization (BN3d) layer and a ReLU activation function, followed by a Max Pooling layer. All the convolutions are $3 \times 3 \times 3$ with initial number of filters equal to 64, while the encoder endpoint has 1024 filters and the resulting maps is 8 time spatially smaller than the input image.

The output of each downsample stage (except the bottleneck) is propagated through skip connections, filtered by a corresponding AG and then concatenated with the output of the upsampling stage at the immediate lower level. The AG filtering consists of feature selectivity based on contextual information extracted in coarser scales.

An upsampling stage consist of an upsampling mechanism followed by the aforementioned convolution block. The upsampling method can either involve a 3D transposed convolution operator with a learnable kernel, or a trilinear upsampling algorithm. From bottom to the output layer, the number of features are progressively halved, while the spatial dimensions are doubled using the desired upsampling mechanism. At each stage the hidden maps of the lower layer are at first filtered with the coarse hidden features coming from the encoder residual connection in the AG, then upsampled and concatenated with the AG output. The outputs of the hidden layers are then used for deep-supervision: they are concatenated along *feature dimension* and then convolved altogether to produce the final output maps, *i.e.* the predicted segmentation masks.

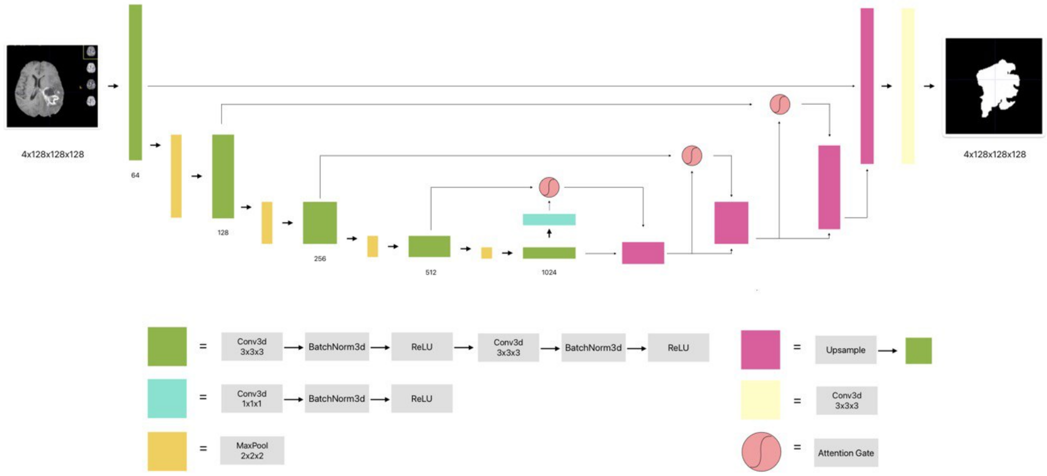


Figure 5.4. Schematic visualization of the Oktay's Attention U-Net architecture adapted for the BraTS task. This implementation follows the one integrated in MONAI [6] framework, where the *deep supervision* branch has been removed.

5.2.2 Loss, Optimization and Regularization

In the original paper the loss function consisted of a single term:

$$\mathbf{L} = \mathbf{L}_{dice} \tag{5.10}$$

where $\mathbf{L}_{dice}$ is a custom soft dice loss applied to the decoder output p_{pred} to match the segmentation mask p_{true}:

$$\mathbf{L}_{dice} = \frac{2 * \sum p_{true} * p_{pred}}{\sum p_{true} + \sum p_{pred} + \epsilon} \tag{5.11}$$

that *w.r.t.* 5.2 does not make use of the squared terms at the denominator.

The model was trained using an Adam optimizer with initial learning rate of $\alpha_0 = 1e - 4$, progressively decreased by an hyperparameter γ every *step* epochs:

$$\alpha = \alpha_0 * \gamma^{\lceil \frac{e}{step} \rceil} \tag{5.12}$$

where e is an epoch counter. They trained their model for 1000 epochs, using a batch size of 2. As regularization tricks they added a L2 penalty term equal to $1e - 6$, but did not use any Dropout layer over the network.

5.2.3 Data preprocessing and augmentation

Input data was processed using standard data augmentation techniques (affine transformations, axial flips, random crops) and intensity values of the scans were linearly scaled to obtain a normal distribution $\mathcal{N}(0, 1)$.

5.2.4 Experiments and Results

To demonstrate the applicability of the modularity and Independence of the proposed AG component, the Attention U-Net model was trained for a challenging abdominal CT multi-label segmentation task. The difficulty of the task came manly from the shape-variability and poor tissue contrast, two factors highly related to the BraTS challenge. The authors compared the proposed model against the standard U-Net model, proving that the performance improvement was consistent and significant for different metrics (Dice score, Precision and Recall) over different types of target, also when using less training data and competing with similar or less network capacity.

5.3 SwinUNETR

Transformer

Before the advent of the Transformer [30], most of the *sequence transduction* models were based on complex recurrent or convolutional neural networks with an encoder-decoder architecture. These Transformers, albeit reusing the same kind of architecture abandon the recurrence and instead relies entirely on the so-called attention mechanism to draw global dependencies between input and output and favouring both parallelization and convergence speed. The encoder is composed of a stack of identical layers comprising two sub-layers: a multi-head self attention mechanism and position-wise fully connected feed-forward network. The decoder is as well composed of a stack of identical layers, but in addition to the two encoder sub-layers, it adds a multi-head attention layer over the output of the encoder stack. The first self-attention sub-layer is modified to prevent position from attending to subsequent positions: this masking ensures that predictions for position i can depend only on the known outputs at position less than i.

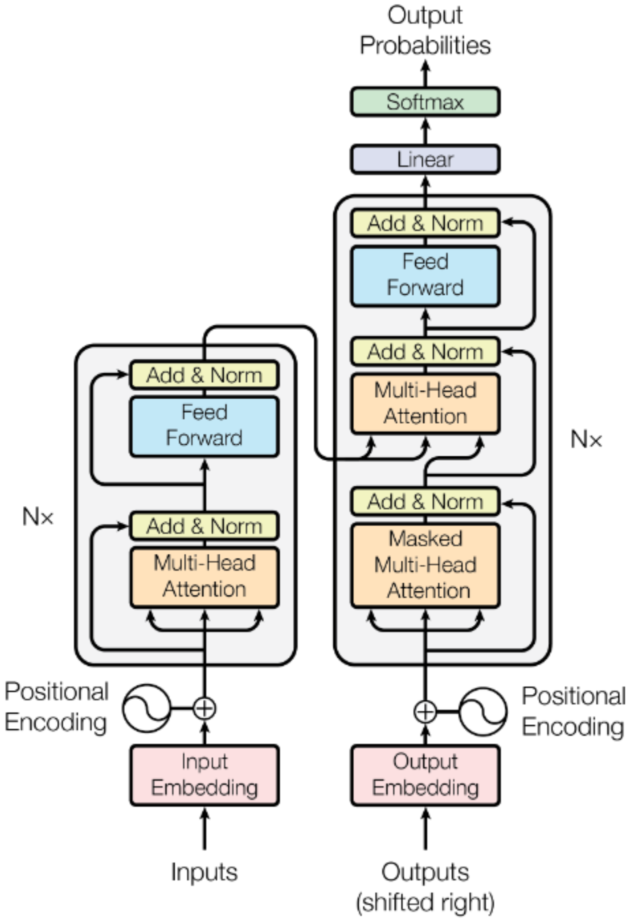


Figure 5.5. The Transformer - model architecture.

The attention mapping can be described as mapping a query and a set of key-value pairs to an output, where they all are vectors. The output is a weighted sum of the values, whose weights are computed by a compatibility function of the query with the corresponding key. In practice, the attention function is computed on a set of queries simultaneously, packed together into a matrix Q, the keys and the values into matrices K and V respectively:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V \tag{5.13}$$

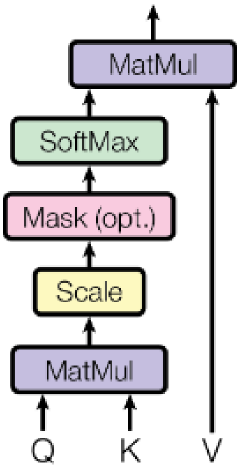
where d_k is the dimension of both queries and keys.

The two most commonly used attention functions are additive attention [3] and dot-product (multiplicative) attention. Additive attention computes the compatibility function using a feed-forward network with a single hidden layer. While being similar in theoretical complexity, dot-product attention is much faster and more space-efficient in practice. Additive attention works best without scaling for larger values of d_k [5]. Probably, for large values of d_k the dot products grow large in magnitude, pushing the softmax function into region where it has extremely small gradients.

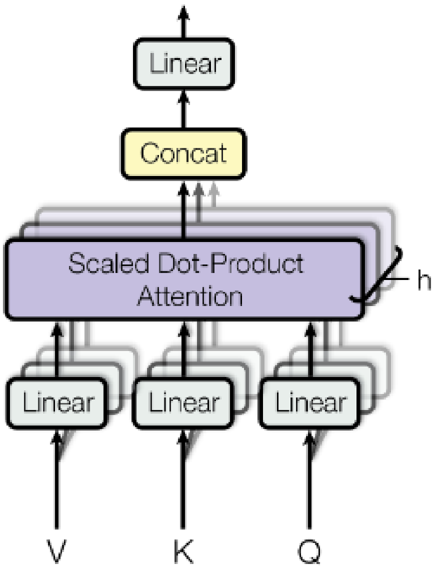
Instead of performing a single attention function with d_{model}-dimensional keys, values and queries, it is beneficial to linearly project the same vectors h times with different learned linear projections to $d_k, d_k and d_v$ dimensions, respectively. On each of these projected versions of queries, keys and values the attention function is performed in parallel, yielding d_v-dimensional output values. These are concatenated and once again projected. Multi-head attention allows the model to jointly attend to information from different representations subspaces at different positions:

$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head_1}, \ldots, \mathrm{head_h})W^O \tag{5.14}$$

where $\mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$ and the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_{\mathrm{model}} \times d_k}, W_i^K \in \mathbb{R}^{d_{\mathrm{model}} \times d_k}, W_i^V \in \mathbb{R}^{d_{\mathrm{model}} \times d_v} \mathrm{and} W_i^O \in \mathbb{R}^{hd_v \times d_{\mathrm{model}}}$.



(a) Scaled Dot-Product Attention.



(b) Multi-Head Attention consists of several attention layers running in parallel.

When comparing self-attention, recurrent and convolutional layers for modeling a sequence transduction task, three main factors must be considered: the computational complexity per layer, the amount of parallelizable computation and the path length between long-range dependencies in the network. One key factor affecting the ability to learn such dependencies is the length of the paths forward and backward signals have to traverse: the shorter these paths between any combination of positions in the input/output sequences, the easier is it to learn long-range dependencies. A self-attention layer connects all positions with a constant number of sequentially executed operations, whereas a recurrent layer requires $O(n)$ sequential operations. In terms of computational complexity, self-attention layers are faster than recurrent layers when the sequence length n is smaller than the representation dimensionality d, which is the most common case. A single convolutional layer with kernel width $k < n$ does not connect all pairs of input and output positions. Doing so requires a stack of $O(n/k)$ convolutional layers in the case of contiguous kernels, or $O(log_k(n))$ in the case of dilated convolutions [21], increasing the length of the longest paths between any two positions in the network. Convolutional layers are generally more expansive than recurrent layers by a factor of k. Separable convolution [8], however, decrease the complexity to $O(k \cdot n \cdot d + n \cdot d^2)$.

As side benefit, self-attention could yield more interpretable models and the multi-head attention mechanism can lend itself naturally to the resolution of different tasks.

Vision Transformer and UNETR

While becoming the de-facto standard for NLP tasks, the convolutional architectures has remained predominant in the Vision domain, albeit multiple works tried either combining the CNN-like architectures with self-attention mechanism [7], [33], or removing the convolutions entirely [11]. The most successful attempt was [11], where the authors experimented with applying the standard Transformer directly to images, with the fewest possible modification. The core idea was to split an image into patches (tokens), convert them in a sequence of linear embeddings and feed the sequence to a Transformer. When trained on mid-sized dataset such as ImageNet [10] underperform *w.r.t.* CNNs of comparable size, but this outcome may be not surprising: the Transformer architecture lacks some of the inductive biases that characterize the CNNs, such as translation equivariance and locality and as a consequence they tend to badly overfit on the training data. To get desirable results and surpass the need of those biases, training on larger dataset (14M-300M images) is required: the Vision Transformer (ViT) achieves excellent result when pre-trained at sufficient scale and then fine-tuned to tasks with fewer examples.

The model design follow the original by [30] with minor modifications to enable the Transformer to handle 2D images: the image $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ is reshaped into a sequence of flattened 2D patches $\mathbf{x}_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$, where (H, W) is the resolution of the original image, C is the number of channels, (P, P) is the resolution of each image patch, and $N = (H \times W)/P^2$ is the resulting number of patches. These get all flattened and embedded into a D-dimensional space via a trainable linear projection. The Transformer encoder consists of alternating layers of multiheaded self-attention (MSA) and MLP blocks. Layernorm (LN) is applied before every block, and residual connections after every block. The MLP contains two layers with a GELU activation function. The classification head is implemented by a MLP with one hidden layer at pre-training time and by a single layer at fine-tuning time. Similar to BERT's `[class]` token, a learnable embedding ($\mathbf{x}_{class}$) is prepended to the sequence of embedded patches (Eq.5.15), whose state at the output of the Transformer encoder

($\mathbf{z}_L^0$) serves as the image representation $\mathbf{y}$ (Eq.5.18).

The whole set of equations describing the Vision Transformer architecture is:

$$\mathbf{z}_0 = [\mathbf{x}_{class};\, \mathbf{x}_p^1\mathbf{E};\, \mathbf{x}_p^2\mathbf{E};\, \cdots;\, \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}, \tag{5.15}$$

$$\mathbf{z'}_l = \mathrm{MSA}(\mathrm{LN}(\mathbf{z}_{l-1})) + \mathbf{z}_{l-1}, \tag{5.16}$$

$$\mathbf{z}_l = \mathrm{MLP}(\mathrm{LN}(\mathbf{z'}_l)) + \mathbf{z'}_l, \tag{5.17}$$

$$\mathbf{y} = \mathrm{LN}(\mathbf{z}_L^0) \tag{5.18}$$

ViT has less image inductive bias than a CNN: in this architecture principle of locality, two-dimensional neighborhood structure and translation equivariance are baked into each layer throughout the whole model. In ViT, the self-attention layers are global and only the MLP layers are local and translationally equivariant: the position embeddings at initialization time carry no information about the spatial relations inter patches, that must be learned during the training phase.

Analyzing the transfer performance from pre-training to downstream task, a few patterns can be observed: ViTs surpass ResNets on the performance/compute tradeoff using from two to four time less compute to achieve similar performance. Also, large variants of the ViT architecture vanish the differences with the hybrids model comprising CNN as feature extractor before the patch embedding phase.

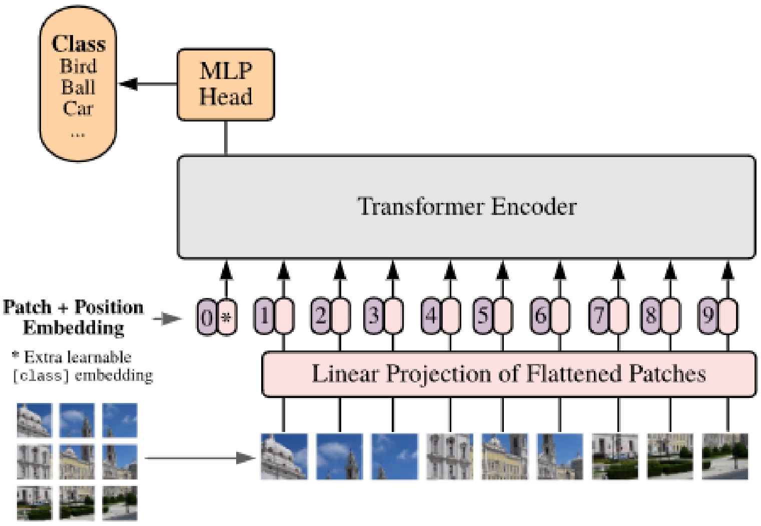

Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches

Figure 5.7. Vision Image Transformer (ViT) architecture overview. An image is splitted into fixed-size patches, linearly embedded, a position embeddings is added and the resulting sequence of vectors are is fed to a standard Transformer encoder.

5.3.1 UNETR

Following the intuition that the Transformer architecture could have been easily adapted from the NLP to the VIsion domain, the authors in [15] reformulate the task of 3D segmentation as a 1D sequence-to-sequence (seq2seq) prediction problem and use a transformer as the encoder to learn contextual information from the embedded input patches to capture long-range dependencies, and a CNN-based encoder to predict the segmentation outputs. The novel architecture - UNEt TRansfomer (UNETR) - is named after the architectures that inspired them: the U-Net and the Transformer.

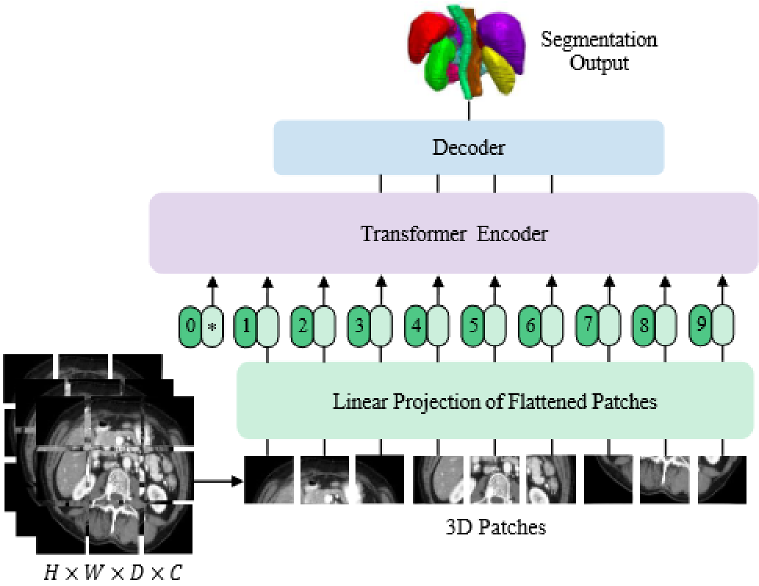


Figure 5.8. UNEt TRansformer (UNETR) architecture overview. The model consists of a transformer encoder that directly utilizes 3D patches and is connected to a CNN-based decoder via skip connection.

In contrast with recently hierarchical vision transformers [9], [32], [35], [24] with varying resolutions and spatial embeddings, the size of representation in UNETR encoder remain fixed in all transformer layers, whereas in the decoder deconvolutions and convolutions operations are used to change the resolution of extracted features. UNETR uses a contracting path consisting of a stack of transformers connected to the expanding path encoder via skip connections. A 1D sequence is created from a 3D input volume $\mathbf{x} \in \mathbb{R}^{H \times W \times D \times C}$ with resolution (H, W, D) and C input channels by dividing it into flattened uniform non-overlapping patches $\mathbf{x}_v \in \mathbb{R}^{N \times (P^3 \cdot C)}$ where (P, P, P) denotes the resolution of each patch and $N = (H \times W \times D)/P^3$ is the length of the sequence. Then, a linear layer is used to project the patches into a K-dimensional embedding space. The set of equations describing the UNETR architecture is:

$$\begin{aligned}
\mathbf{z}_0 &= [\mathbf{x}_v^1\mathbf{E};\, \mathbf{x}_v^2\mathbf{E};\, \cdots;\mathbf{x}_v^N\mathbf{E}] + \mathbf{E}_{pos}, && (5.19)\\
\mathbf{z}'{}_l &= \mathrm{MSA}(\mathrm{LN}(\mathbf{z}_{l-1})) + \mathbf{z}_{l-1}, && (5.20)\\
\mathbf{z}_l &= \mathrm{MLP}(\mathrm{LN}(\mathbf{z}'{}_l)) + \mathbf{z}'{}_l, && (5.21)\\
& && (5.22)
\end{aligned}$$

note: *w.r.t.* the standard ViT architecture, here there is no need of the class token nor the final image representation $\mathbf{y}$ since the transformer backbone is designed for semantic segmentation. By similarity with U-Net architecture, a sequence representation $\mathbf{z}_i (i \in 3, 6, 9, 12)$, with size $\frac{H \times W \times D}{P3} \times K$ is extracted from the transformer and reshaped into a $\frac{H}{P} \times \frac{W}{P} \times \frac{D}{P} \times K$ tensor.

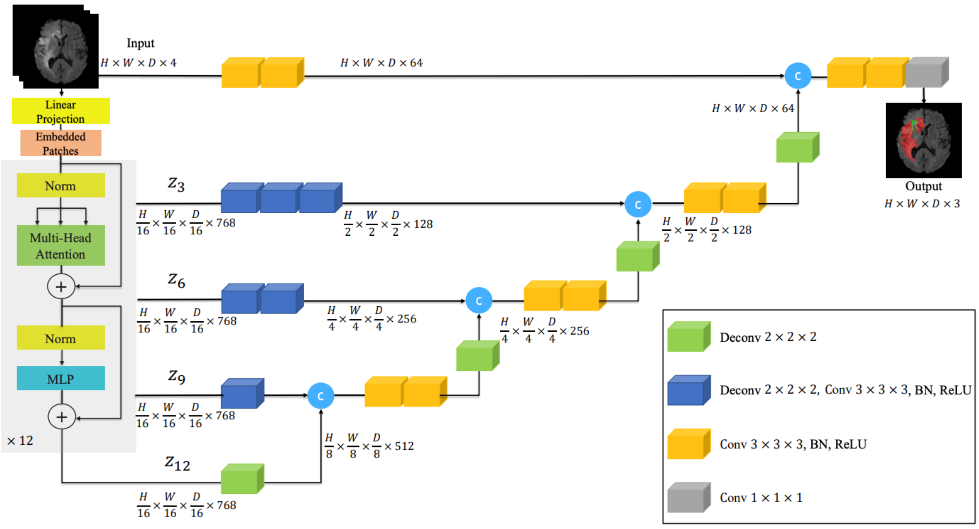

Input
H × W × D × 4
H × W × D × 64
Linear Projection
Embedded Patches
Norm
Multi-Head Attention
+
Norm
MLP
+
× 12
z_3
$\frac{H}{16} \times \frac{W}{16} \times \frac{D}{16} \times 768$
z_6
$\frac{H}{16} \times \frac{W}{16} \times \frac{D}{16} \times 768$
z_9
$\frac{H}{16} \times \frac{W}{16} \times \frac{D}{16} \times 768$
z_{12}
$\frac{H}{16} \times \frac{W}{16} \times \frac{D}{16} \times 768$
$\frac{H}{2} \times \frac{W}{2} \times \frac{D}{2} \times 128$
$\frac{H}{4} \times \frac{W}{4} \times \frac{D}{4} \times 256$
$\frac{H}{8} \times \frac{W}{8} \times \frac{D}{8} \times 512$
c
H × W × D × 64
$\frac{H}{2} \times \frac{W}{2} \times \frac{D}{2} \times 128$
$\frac{H}{4} \times \frac{W}{4} \times \frac{D}{4} \times 256$
Output
H × W × D × 3
Deconv 2 × 2 × 2
Deconv 2 × 2 × 2, Conv 3 × 3 × 3, BN, ReLU
Conv 3 × 3 × 3, BN, ReLU
Conv 1 × 1 × 1

Figure 5.9. Schematic of UNETR architecture. A 3D input volume is divided into a sequence of uniform non-overlapping patches and projected into an embedding space. The sequence is added with a position embedding and used as an input to a transformer model. The encoded representations of different layers in the transformer are extracted and propagated to a decoder via skip connections to predict the final segmentation

5.3.2 Loss, Optimization and Regularization

The loss used in the paper is a combination of soft dice loss and cross-entropy loss:

$$\mathbf{L} = 1 - \mathbf{L}_{dice}^{\mathrm{UNETR}} + \mathbf{L}_{CE}^{\mathrm{UNETR}}, \tag{5.23}$$

$$\mathbf{L}_{dice}^{\mathrm{UNETR}} = \frac{2}{J} \frac{\sum p_{true} * p_{pred}}{\sum p_{true}^2 + \sum p_{pred}^2} \tag{5.24}$$

$$\mathbf{L}_{CE}^{\mathrm{UNETR}} = -\frac{1}{I} \sum_{i=1}^{I} \sum_{j=1}^{J} p_{true}^{i,j} * log(p_{pred}^{i,j}) \tag{5.25}$$

where I is the number of voxels and J is the number of classes. The model was trained using an AdamW optimizer with initial learning rate $\alpha_0 = 1e - 4$ for 20.000 iterations and batch size of 6.

5.3.3 Data preprocessing and augmentation

The authors used the Medical Segmentation Decathlon (MSD) dataset by [2], which for the brain tumour segmentation sub-task comprises 484 training examples of multi-modal multi-site MRI data as illustrated in Ch. 4. The voxel spacing of

MRI data is $1.0 \times 1.0 \times 1.0$ mm^3 and their intensity is pre-processed with z-score normalization. For the augmentation part, the authors used random rotation, flip in axial, sagittal and coronal views, random scale and shift intensity, plus a random crop of $128 \times 128 \times 128$.

5.3.4 Experiments and Results

The encoder follows the ViT-B16 [11] architecture with $L = 12$ layers, embedding size of $K = 768$ and a patch resolution of $16 \times 16 \times 16$. No pre-trained weights for the transformer backbone were used since it did not demonstrate any performance improvements. Compared to other CNN and transformer-based solutions (see Table 5.3), UNETR outperforms the closest baseline by 1.5% on average over all semantic classes and in particular performs considerably better in segmenting TC sub-region.

Table 5.3. Quantitative comparisons of the segmentation performance in brain tumor task of the MSD dataset. EN - enhancing tumor core, WT - whole tumor, TC - tumor core

	Dice			Hausdorff (mm)		
Tumor sub-region	**ET**	**WT**	**TC**	**ET**	**WT**	**TC**
UNet	0.561	0.766	0.665	11.122	9.205	10.243
AttUNet	0.543	0.767	0.683	10.447	9.004	10.463
UNETR	**0.585**	**0.789**	**0.761**	**9.354**	**8.266**	**8.845**

Swin Transformer

Challenges in adapting the Transformer architecture from language to vision come from the differences between the two domains, mainly for the large scale variations of visual elements and the possibility to have a wide range of resolutions of pixels (voxels) in images (volumes) compared to words in text. In most of the existing Transformer-based models the tokens are all of a fixed scale: a property unsuitable for vision tasks that may require dense prediction at the pixel level, and this would be intractable in case of high-resolution images (volumes), as the computational complexity of the self-attention is quadratic to input size. By addressing this issue, the authors in [24] propose a general-purpose Transformer backbone, the Shifted window Transformer, which constructs hierarchical feature maps and has linear computational complexity to input size, achieved by computing self-attention locally within non-overlapping windows that partition an image (see Fig. 5.10).

The key idea behind Swin-Transformer is the alternating partition scheme between consecutive self-attention layers in the encoder part: the *shift* bridges windows in consecutive layers modeling high correlations in visual signals that significantly enhance model power. The proposed architecture outperforms the ViT [11] and ResNe(X)t [16], [34] models on the three tasks of image classification, object detection and semantic segmentation on ImageNet (both 1k and 22K versions), COCO and ADE20K datasets. Despite being a pioneering solution in the Vision domain, not only the ViT architecture requires large-scale training datasets to perform well, but also it is unsuitable for use a general-purpose backbone network on dense vision

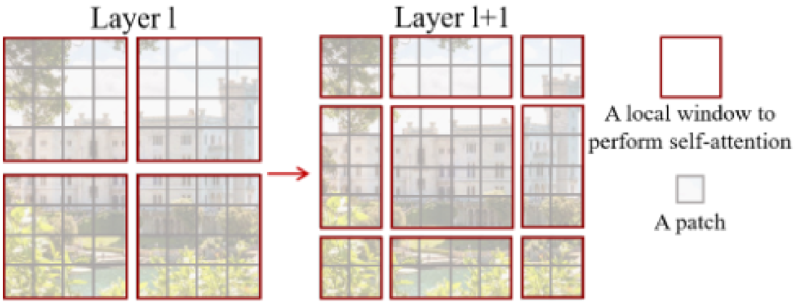


Figure 5.10. An illustration of the *shifted window* approach proposed Swin Transformer architecture. In layer l (left), a regular window partitioning scheme is adopted, while in the next layer $l+1$ (right), the window partitioning is shifted, resulting in new windows: the SA computation in the new windows crosses the boundaries of the previous one, providing connections among them.

task, due to its low-resolution feature maps and the quadratic increase in complexity with input size. The general architecture of the Swin Transformer (see Fig. 5.11) is composed by an initial patch splitting module and then a sequence of stages equipped with a linear embedding layer a patch merging layer and a series of blocks with modified self-attention computation (*Swin Transformer blocks*, see Fig. 5.12). The embedding happens only at "Stage 1", where the original patches get embedded into an arbitrary dimension C. In the other stages the patch merging layer reduces the number of tokens in order to produce a hierarchical representation of the features: as a result, the proposed network can replace many traditional CNN backbones for various Vision tasks.

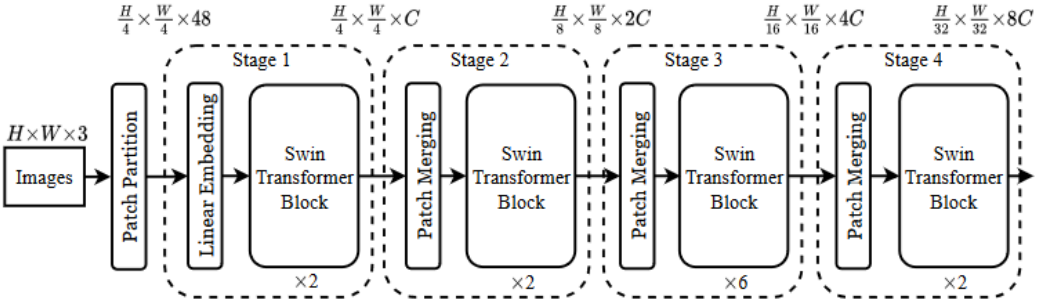


Figure 5.11. The architecture overview of the Tiny version of the Swin Transformer (Swin-T)

The alternating partition scheme happens in consecutive Swin Transformer Blocks: the first module uses a regular partitioning strategy starting from the top-left corner of the image, then the next module adopts a partition that is shifted from that of the preceding layer by $(\lfloor\frac{M}{2}\rfloor, \lfloor\frac{M}{2}\rfloor)$ pixels from the regularly partitioned windows. With this approach, the formulation of consecutive layers inside a block become:

$$\begin{aligned}
\hat{\mathbf{z}}^l &= \text{W-MSA}(\text{LN}(\mathbf{z}^{l-1})) + \mathbf{z}^{l-1}, && (5.26)\\
\mathbf{z}^l &= \text{MLP}(\text{LN}(\hat{\mathbf{z}}^l) + \hat{\mathbf{z}}^l, && (5.27)\\
\hat{\mathbf{z}}^{l+1} &= \text{SW-MSA}(\text{LN}(\mathbf{z}^l)) + \mathbf{z}^l, && (5.28)\\
\mathbf{z}^{l+1} &= \text{MLP}(\text{LN}(\hat{\mathbf{z}}^{l+1}) + \hat{\mathbf{z}}^{l+1} && (5.29)\\
& && (5.30)
\end{aligned}$$

where W-MSA and SW-MSA denote window based multi-head self-attention using regular and shifted window partitioning configurations, respectively. Supposing each window contains $M \times M$ patches, the computational complexity of a global MSA module and a window based one on an image of $h \times w$ patches are:

$$\begin{aligned}
\Omega(\text{MSA}) &= 4hwC^2 + 2(hw)^2C, && (5.31)\\
\Omega(\text{W-MSA}) &= 4hwC^2 + 2M^2hwC, && (5.32)\\
& && (5.33)
\end{aligned}$$

where the former is quadratic to patch number hw, and the latter is linear when M is fixed, making the window-based version scalable.

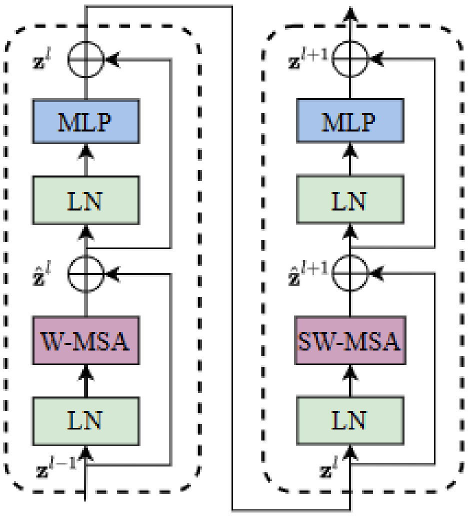


Figure 5.12. Two Successive Swin Transformer Blocks. The layer l (left) incorporates the standard W-MSA layer, the next layer $l+1$ (right) is equipped instead with the shifted version of the *window multihead self-attention*: the SW-MSA layer.

5.3.5 Swin UNETR model

With the advancement in adapting the Transformer architecture to the Vision domain, the authors of [14] propose a novel architecture named Swin UNEt Transformer (Swin UNETR), which utilizes a U-shaped network with a Swin Transformer as the encoder (hierarchical feature extractor) connected to a CNN-based decoder, for the task of multimodal 3D brain tumor segmentation in the 2021 edition of the BraTS challenge. The Swin UNETR model proved to have greater capability of learning multi-scale contextual representation and modeling long-range dependencies *w.r.t* ViT-based approach at fixed resolution. The input to the Swin UNETR model $\mathbf{x} \in \mathbb{R}^{H \times W \times D \times S}$ is a token with a patch resolution of (H', W', D') and dimension of $H' \times W' \times D' \times S'$. A patch partition layer (see Fig. 5.13) creates a sequence of 3D tokens with dimension of $\lceil \frac{H}{H'} \rceil \times \lceil \frac{W}{W'} \rceil \times \lceil \frac{D}{D'} \rceil$, embedded into a space with dimension C. The alternating partitioning scheme follows from the one proposed in [24], but in this case is applied to a 3D volume instead: windows of size $M \times M \times M$ partition a 3D token into $\lceil \frac{H'}{M} \rceil \times \lceil \frac{W'}{M} \rceil \times \lceil \frac{D'}{M} \rceil$ regions at a layer l, then at layer $l+1$ the partitioning windows get shifted by $(\lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor)$ voxels.

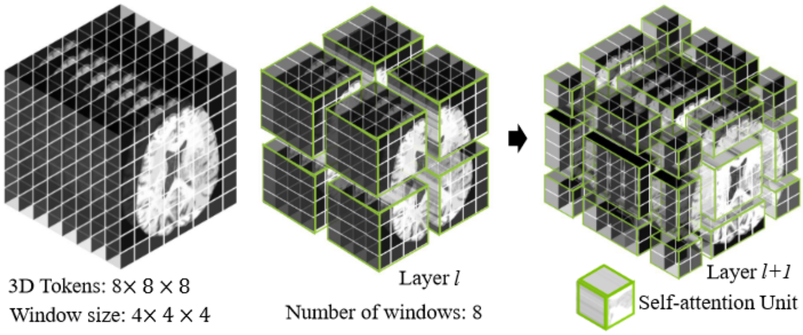


Figure 5.13. An illustration of the *shifted window* mechanism proposed for the Swin UNETR architecture.

The set of equation describing the computations inside the encoder is:

$$\hat{\mathbf{z}}^l = \text{W-MSA}(\text{LN}(\mathbf{z}^{l-1})) + \mathbf{z}^{l-1}, \tag{5.34}$$

$$\mathbf{z}^l = \text{MLP}(\text{LN}(\hat{\mathbf{z}}^l)) + \hat{\mathbf{z}}^l, \tag{5.35}$$

$$\hat{\mathbf{z}}^{l+1} = \text{SW-MSA}(\text{LN}(\mathbf{z}^l)) + \mathbf{z}^l, \tag{5.36}$$

$$\mathbf{z}^{l+1} = \text{MLP}(\text{LN}(\hat{\mathbf{z}}^{l+1})) + \hat{\mathbf{z}}^{l+1} \tag{5.37}$$

The Swin UNETR encoder has a patch size of $2 \times 2 \times 2$ and a feature dimension of $2 \times 2 \times 2 \times 4 = 32$, taking into account the 4 modalities of MRI images (FLAIR, T1, T1c, T2).The patches are then projected into a 48-dimensional space and pass through each of the 4 stages of the encoder. Each stage comprises two Transformer blocks and a patch merging layer that enables the building of hierarchical feature maps by reducing their resolutions and, in combination with a linear layer, doubling their size. Following the U-shape network, at each stage $i (i \in 0, 1, 2, 3, 4)$ and in the bottleneck the features get first reshaped into size $\frac{H}{2^i} \times \frac{W}{2^i} \times \frac{D}{2^i}$ then fed into a residual block comprising of two $3 \times 3 \times 3$ convolutional layers and an InstanceNorm (IN) layer. The resolution of the feature maps is increased once more by 2 using a deconvolutional layer and the maps are concatenated with the ones propagated via skip connection at the same level. The final segmentation outputs are computed by using a $1 \times 1 \times 1$ convolutional layer to get the desired number of output channel, and are activated by the sigmoid activation function.

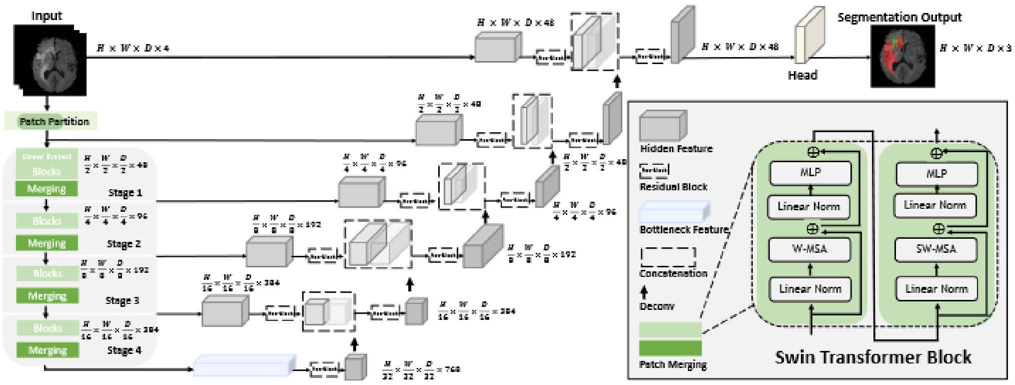


Figure 5.14. Overview of the Swin UNETR architecture. The input to our model is 3D multi-modal MRI images partitioned into non-overlapping patches for computing the self-attention. The encoded feature representations are fed to a CNN-decoder via skip connection at multiple resolutions. Final segmentation output consists of 3 output channels corresponding to ET,WT and TC sub-regions.

5.3.6 Loss, Optimization and Regularization

In the original paper was used the soft Dice loss function computed in a voxel-wise manner as:

$$\mathbf{L} = 1 - \mathbf{L}_{dice}^{\text{Swin UNETR}} + \mathbf{L}_{CE}^{\text{Swin UNETR}}, \tag{5.38}$$

$$\mathbf{L}_{dice}^{\text{Swin UNETR}} = \frac{2}{J} \frac{\sum p_{true} * p_{pred}}{\sum p_{true}^2 + \sum p_{pred}^2} \tag{5.39}$$

where J is the number of classes; y_{pred}andy_{true} denote the probability of output and one-hot encoded ground truth respectively. The model was trained using an AdamW optimizer with initial learning rate $\alpha_0 = 8e - 4$ and batch size of 1. All models were trained for 800 epochs using a cosine annealing learning rate scheduler.

5.3.7 Data preprocessing and augmentation

The 2021 BraTS dataset comprises 1251 training examples, each with the 4 MRI modalities (FLAIR, T1, T1-c, T2) rigidly aligned, skull-stripped and resampled to a $1 \times 1 \times 1$ mm^3 isotropic resolution. All the images were normalized with z-score normalization and unit standard deviation according to non-zero voxels. Additionally, random axis mirror flip with a probability of 0.5 for all 3 axes, random per channel intensity shift in the range $(-0.1, 0.1)$ and random scale of intensity in the range $(0.9, 1.1)$ to input image channels were applied. Random crops of $128 \times 128 \times 128$ were taken from 3D image volumes.

5.3.8 Experiments and Results

The performance of Swin UNETR has been compared against the winning models of previous years such as SegResNet [27], nnU-Net [18] and TransBTS [31].

Table 5.4. Quantitative comparisons of the segmentation performance in brain tumor task of the 2021 BraTS challenge dataset. EN - enhancing tumor core, WT - whole tumor, TC - tumor core

	Dice			Hausdorff		
Tumor sub-region	**ET**	**WT**	**TC**	**ET**	**WT**	**TC**
SegResNet	0.883	0.927	0.913	unk	unk	unk
nnU-Net	0.883	0.927	0.913	unk	unk	unk
TransBTS	0.868	0.915	0.893	unk	unk	unk
Swin UNETR	**0.891**	**0.933**	**0.917**	6.016	5.831	3.770

Chapter 6

Experimental Design and Result

6.1 Frameworks and Hardware Details

All the experiments was conducted using Python as the only programming language: from data visualization to models implementation, training and validation. Aiming for reproducibility, flexibility in early development and assistance with the aspect related to DL research I relied heavily on PyTorch Lightning [12] to organize PyTorch code avoiding boilerplate and to be able to swap components of the *"training pipeline"* (models, optimizers, LR scheduler, losses...) and perform ablation studies with ease. The models are implemented with PyTorch [29]: an optimized machine learning framework based on the Torch library which provides two high-level features: Tensor computing with strong acceleration via graphics processing units (GPUs) and Deep neural networks built on a tape-based automatic differentiation system. For the (re)implementation and usage of medical related components (dataset, network architectures, segmentation losses...) I used MONAI [6] - the Medical Open Network for AI - an open-source foundation for deep learning in healthcare imaging, part of the PyTorch-Ecosystem. It is domain-optimized, community-backed and provides many useful resources to start learning and experimenting with medical research. Before even starting visualizing the data I should have worked with, I realized that the resources of my machine were not sufficient to carry on training the models as reviewed in Ch. 5 and that's why I moved onto Kaggle, an online platform for data scientists and machine learning practitioners, that allows users to collaborate with other users, find and publish datasets and use GPU integrated notebooks for free (with some limitations). In particular, it provides resources on cloud and allows for a quota of 30 GPU-hours weekly and 12 continuous hours of active session with a clear view of the available resources and the remaining time, whereas other platforms, such as Google Colab, are much less informative, stable and defined: in many cases the quota of training hours can vary weekly, according to the amount of requests that a platform is receiving. To train effectively at no-cost and to have comparable results without resorting to split training sessions, Kaggle was the only viable way. The types of GPU provided by Kaggle are the NVIDIA Tesla P100 GPU (16 GB) and T4 GPU x 2 (8 GB). I always trained on the former, since 16 GB was the minimum memory requirement for training most of the considered architectures. Lastly, in order to efficiently keep track of the experiments (model architecture, parameters, hyper-parameters, results and annotations), store the artifacts generated during the training and to be able to recover a training session in case of failure, I used Weights & Biases [4]. Weights & Biases, also known as WandB, is an MLOps tool for performance visualization and experimental tracking of machine learning models, is a cloud-based service that allows you to host your experiments in a single central repository or to run them on-premises setup.

6.2 Reproducing Previous Work

As already mentioned, in order to make comparison as fair as possible and to have a baseline to confront with, I trained from scratch all the models described in Ch. 5 using the same training dataset: the Medical Segmentation Dataset [2] (MSD) sub-task: "Task01_BrainTumour" obtainable with MONAI's class DecathlonDataset and ready to be used for training, validation and testing. For the hyperparameter setting I abide to the guidance of the original papers when possible, making changes only when forced by hardware limitations.

6.3 Proposed Method

6.4 Ablation Studies

6.5 Results

Chapter 7

Conclusions and Future Work

Bibliography

[1] Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., and Zhang, L. Bottom-up and top-down attention for image captioning and visual question answering (2018). arXiv:1707.07998.

[2] Antonelli, M., et al. The medical segmentation decathlon. *Nature Communications*, **13** (2022). Available from: https://doi.org/10.1038%2Fs41467-022-30695-9, doi:10.1038/s41467-022-30695-9.

[3] Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate (2016). arXiv:1409.0473.

[4] Biewald, L. Experiment tracking with weights and biases (2020). Software available from wandb.com. Available from: https://www.wandb.com/,.

[5] Britz, D., Goldie, A., Luong, M.-T., and Le, Q. Massive exploration of neural machine translation architectures (2017). arXiv:1703.03906.

[6] Cardoso, M. J., et al. MONAI: An open-source framework for deep learning in healthcare. (2022). doi:https://doi.org/10.48550/arXiv.2211.02701.

[7] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. End-to-end object detection with transformers (2020). arXiv:2005.12872.

[8] Chollet, F. Xception: Deep learning with depthwise separable convolutions (2017). arXiv:1610.02357.

[9] Chu, X., Tian, Z., Wang, Y., Zhang, B., Ren, H., Wei, X., Xia, H., and Shen, C. Twins: Revisiting the design of spatial attention in vision transformers (2021). arXiv:2104.13840.

[10] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. IEEE (2009). doi:10.1109/CVPR.2009.5206848.

[11] Dosovitskiy, A., et al. An image is worth 16x16 words: Transformers for image recognition at scale (2021). arXiv:2010.11929.

[12] Falcon, W. Pytorch lightning: The lightweight pytorch wrapper for high-performance ai research. scale your models, not the boilerplate. Available from: "https://www.pytorchlightning.ai"repository-code:"https://github.com/Lightning-AI/lightning".

[13] Futrega, M., Milesi, A., Marcinkiewicz, M., and Ribalta, P. Optimized u-net for brain tumor segmentation (2021). arXiv:2110.03352.

[14] Hatamizadeh, A., Nath, V., Tang, Y., Yang, D., Roth, H., and Xu, D. Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images (2022). arXiv:2201.01266.

[15] Hatamizadeh, A., Tang, Y., Nath, V., Yang, D., Myronenko, A., Landman, B., Roth, H., and Xu, D. Unetr: Transformers for 3d medical image segmentation (2021). arXiv:2103.10504.

[16] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition (2015). arXiv:1512.03385.

[17] Isensee, F., Kickingereder, P., Wick, W., Bendszus, M., and Maier-Hein, K. H. No new-net (2019). arXiv:1809.10483.

[18] Isensee, F., et al. nnu-net: Self-adapting framework for u-net-based medical image segmentation (2018). arXiv:1809.10486.

[19] Jetley, S., Lord, N. A., Lee, N., and Torr, P. H. S. Learn to pay attention (2018). arXiv:1804.02391.

[20] Jiang, Z., Ding, C., Liu, M., and Tao, D. *Two-Stage Cascaded U-Net: 1st Place Solution to BraTS Challenge 2019 Segmentation Task*, pp. 231–241. Springer Cham (2020). ISBN 978-3-030-46639-8. doi:10.1007/978-3-030-46640-4_22.

[21] Kalchbrenner, N., Espeholt, L., Simonyan, K., van den Oord, A., Graves, A., and Kavukcuoglu, K. Neural machine translation in linear time (2017). arXiv:1610.10099.

[22] Lee, C.-Y., Xie, S., Gallagher, P., Zhang, Z., and Tu, Z. Deeply-supervised nets (2014). arXiv:1409.5185.

[23] Li, R., et al. A comprehensive review on deep supervision: Theories and applications (2022). arXiv:2207.02376.

[24] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows (2021). arXiv:2103.14030.

[25] Luong, M.-T., Pham, H., and Manning, C. D. Effective approaches to attention-based neural machine translation (2015). arXiv:1508.04025.

[26] Milletari, F., Navab, N., and Ahmadi, S.-A. V-net: Fully convolutional neural networks for volumetric medical image segmentation (2016). arXiv:1606.04797.

[27] Myronenko, A. 3d mri brain tumor segmentation using autoencoder regularization (2018). arXiv:1810.11654.

[28] Oktay, O., et al. Attention u-net: Learning where to look for the pancreas (2018). arXiv:1804.03999.

[29] Paszke, A., et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc. (2019). Available from: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[30] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need (2017). arXiv:1706.03762.

[31] Wang, W., Chen, C., Ding, M., Li, J., Yu, H., and Zha, S. Transbts: Multimodal brain tumor segmentation using transformer (2021). arXiv:2103.04430.

[32] Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P., and Shao, L. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions (2021). arXiv:2102.12122.

[33] Wang, X., Girshick, R., Gupta, A., and He, K. Non-local neural networks (2018). arXiv:1711.07971.

[34] Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. Aggregated residual transformations for deep neural networks (2017). arXiv:1611.05431.

[35] Xu, W., Xu, Y., Chang, T., and Tu, Z. Co-scale conv-attentional image transformers (2021). arXiv:2104.06399.

[36] Zeineldin, R. A., Karar, M. E., Burgert, O., and Mathis-Ullrich, F. Multimodal cnn networks for brain tumor segmentation in mri: A brats 2022 challenge solution (2022). arXiv:2212.09310.