Master thesis : Diagnosis of neurodegenerative diseases with deep learning : The case of Alzheimer's disease

Auteur : Backès, Lucas
Promoteur(s) : Phillips, Christophe; Louppe, Gilles
Faculté : Faculté des Sciences appliquées
Diplôme : Master : ingénieur civil en science des données, à finalité spécialisée
Année académique : 2021-2022
URI/URL : http://hdl.handle.net/2268.2/15982

LIÈGE université
Sciences Appliquées

DIAGNOSIS OF NEURODEGENERATIVE DISEASES WITH DEEP LEARNING :

The case of Alzheimer's disease

Author: Lucas Backes

Academic year 2021 - 2022

A dissertation submitted in partial fulfillment
of the requirements for the degree of

Master of Science in Data Science and Engineering

Conducted under the supervision of Professors :

Louppe Gilles	-	g.louppe@uliege.be
Phillips Christophe	-	c.phillips@uliege.be

Acknowledgements

First and of foremost I would like to express my gratitude to my advisors Professor Christophe Phillips and Professor Gilles Louppe who were always available to help me throughout this work. Thank you to Antoine Wehenkel, Oscar Darias, Mathieu Taquet and my father for their useful feedbacks regarding the codes and the manuscript. Thank you to my friends for their encouragements and for their ability to make me laugh during stressful times.

Finally, thank you to my family for their love and support but more especially for their comfort during this final year.

Abstract

Alzheimer's disease (AD) is one of the most common neurodegenerative diseases in the world and the most common cause of dementia. In recent times, accurate and early detection of AD plays a vital role in patient care and further treatment. Lately, studies on AD diagnosis has attached a great significance to artificial-based diagnostic algorithms. During this master thesis we explore how deep learning models can handle neuroimages in order to identify and predict the evolution of the disease. Different from the traditional machine learning algorithms, deep learning does not require manually extracted features but instead utilizes 3D image processing models to learn features for the diagnosis and the prognosis of AD. The contribution of this work relies on a more rigorous preprocessing phase involving skull-stripping and intensity normalization of the medical images. The hippocampus, a brain area critical for learning and memory, is especially affected at early stages of Alzheimer's disease. In some parts of this work, It will be used as a region of interest for our algorithms that will consist in convolutional neural networks, the typical image classifier models, and vision transformers, a novel deep learning architecture.

Contents

Chapter 1

Introduction

Inventors have long dreamed of creating machines that outperform humans in tasks whether intellectual or manual. When programmable computers were first conceived, many people wondered if such machines might become intelligent. Nowadays, we look towards intelligent software to automate routine labor, drive our cars, understand speech or images and make diagnosis in medicine.

Machine learning techniques in the healthcare sector offer automated diagnosis tools that support a doctor's diagnosis and recommendations to a patient. Such systems must be designed in close cooperation with medical professionals in order to validate the outcomes and have a thorough understanding of the medical requirements.

Alzheimer's disease is a complex brain disease still not completely understood. Indeed, still many research questions related to this disease stay unresolved, e.g. the ability to predict if an individual is likely to develop the disease or not several years later.

In this field, machine learning can help to analyse brain images and to build interpretable diagnosis systems. Dealing with the extremely small size and high dimension of the datasets that are frequently accessible in this domain presents a problem in the implementation of machine learning algorithms.

In this master thesis, we investigate the potential of deep learning methods, a particular machine learning family, to help researchers using brain imaging to address issues concerning Alzheimer's disease.

This manuscript is divided into six chapters :

After this introduction, chapter two will provide the necessary medical background required to understand this work. Explaining Alzheimer's disease as well as describing neuroimaging data.

Chapter 3 proposes an overview of the field of deep learning. It starts by explaining classical supervised learning approach then gradually works its way up to explaining how state of the art model works and how they can be used to deal with medical images. Finally it presents an overview of the related work literature both in deep learning and in classical machine learning.

Chapter 4 comprises the entire methodology carried out for this endeavour. At first, the preprocessing pipeline will be carefully explained step by step. Then, the model architectures used for inference will be described based on the knowledge acquired from the previous chapter.

Chapter 5 contains the experiment procedure conducted along this work. We will first discuss the protocol on which empirical results were obtained, the evaluation metrics used to assess them, then finally they will be reported and discussed.

Chapter 6 will end this master thesis by exploring the potential future of AI assisted diagnosis as well as its ethics before a final conclusion.

Chapter 2

Neurodegenerative Disease

During this first chapter, we will go over the medical background necessary to understand this work. We open this section with a brief description of Alzheimer's disease then follow up with the description of the medical tools available to assess it as well as a potential brain region of interest that could be relevant for the diagnosis and finally talk about the large american database ADNI.

2.1. Alzheimer Disease

Alzheimer's disease (AD) is a neurodegenerative brain disorder in which neurons in specific parts of the brain are irreparably lost as the disease progresses. Memory and behavioral problems result from the neural loss.

Because life expectancy in developed countries has been steadily rising over the past few decades, this condition, which primarily affects elderly people, has become more and more common. Although some researchs have been focusing on the creation of drugs postponing the onset of specific symptoms, there is currently no drug to prevent the disease or stop its progression. The neurodegenerative disorder is currently incurable.

A clinical testing procedure, created for the evaluation of cognitive deficits, is most frequently used to diagnose patients (Kelley and Petersen, 2007). A consortium (the American Psychiatric Association) has approved disease assessment and established a detailed list of requirements to be met in order to be classified as a diseased person. These criteria specifically address memory impairment and issues with executive processes (such as making plans, organizing activities, etc.). Clinical assessments such as the Clinical Dementia Rating scale (CDR) and Mini-Mental State Examination (MMSE) provide an accurate numerical assessment of the course of cognitive impairment.

Once an individual is suspected of Alzheimer's disease, brain imaging is often used to figure out whether disease progression has already caused brain damages. In fact, magnetic resonance imaging (MRI) or positron emission tomography (PET) data clearly show the effects of the disease on the brain (Frisoni et al., 2010; Silverman et al., 2001).

Alzheimer's disease shows different stages of evolution: the cognitively normal stage, the mild cognitive impairment stage (MCI) and finally dementia. The MCI phase, transitional stage between normal aging and the preclinical phase of dementia, causes cognitive impairments with minimal impact on instrumental activities of daily life (Petersen et al., 1999). The evolution of the disease through these different stages is characterized by changes in biomarkers. Clinically speaking, neuropsychological assessments will give different results depending on the time line. Brain atrophy, a natural result of aging that is highlighted by magnetic resonance imaging, is displayed much more severely in AD patients (Figure 2.1). On top of that, disease evolution will cause hypometabolism in specific brain regions. These consequences of the disease will notably be observable with FDG-PET images which provide information about functional deficiency in addition to the anatomical information seen with structural MRI.

Thus, a significant scientific issue is to understand how MCI develops into the demented stage. Treatments could be started sooner to prevent the onset of symptoms and to enable the investigation of additional clinical trials to slow down the illness development if the disease progression was anticipated from a prodromal stage.

Additionally, family members could be better prepared for the onset of the condition. Machine learning could assist in determining whether a patient with MCI was likely to experience full-blown AD as a result of their deficiencies. The earlier a disease is identified, the sooner it may be treated to slow or stop its course and the onset of symptoms.

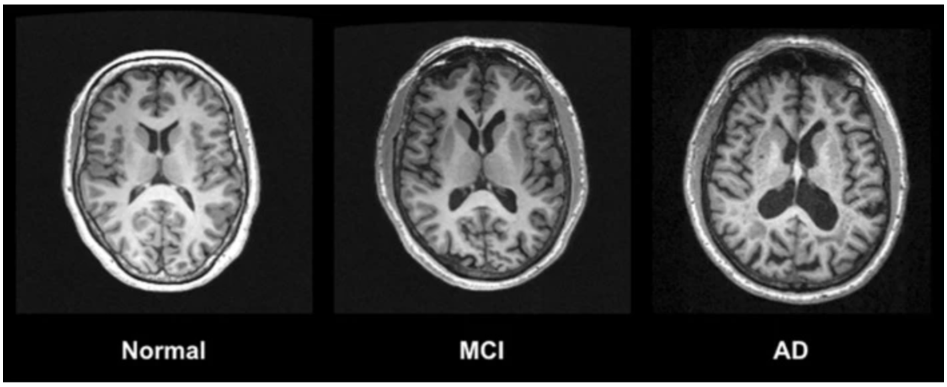


Figure 2.1: MRI imaging sequence shows decreased gray matter volume in an AD patient(right) compared to a healthy control (left) and intermediate grey matter decline in a patient with MCI (middle). From (Chandra et al., 2019)

2.2. MRI scans

Magnetic resonance imaging (MRI) is one of the most commonly used imaging tool in neurology and neurosurgery. MRI provides exquisite details of brain, spinal cord and vascular anatomy, and has the advantage of being able to visualize anatomy in 3 dimensions (see Figure 2.2), for example along the axial (from top to down), sagittal (from side to side) and coronal planes (from front to back). Not to mention that MRI procedures are not invasives.

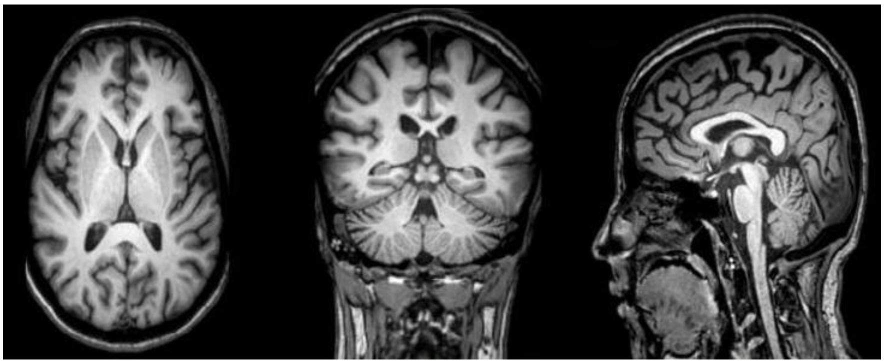


Figure 2.2: MRI planes for MRI head scan Axial (left) Coronal (middle) Sagittal (right). MR scanner can generate three types of orientations of human head.
From (Padmanaban et al., 2020)

MRI is based on the principle of nuclear magnetic resonance, a property exhibited by some nuclei which absorb and emit an electromagnetic radiation when they are under the influence of a strong magnetic field. To drive the contrast in the acquired images, the sequence used to generate image relies on two key parameters, namely the **Repetition Time (TR)** and the **Time to Echo (TE)**.

Two important tissue properties have their two distinct relaxation times, T1 and T2, which can be used to contrast images. The time constant T1, also known as longitudinal relaxation time, determines the rate at which excited protons return to equilibrium. It is an indicator of how long it takes for spinning protons to return to their energy state equilibrium and regrow longitudinal (i.e. along the the main field) magnetization.

The time constant T2, also known as transverse relaxation time, determines the rate at which excited protons go out of phase with each other. It is a measure of the time taken for spinning protons to loose phase coherence thus describing the disappearance of transversal (i.e. perpendicular the the main field) magnetization.

T1- and T2-weighted scans are the most popular MRI sequences, where the T1 and T2 characteristics of tissue are primarily responsible for determining the contrast and brightness in these images. Short TE and long TR timings are used to create T1-weighted images. On the other hand, relatively longer TE and shorter TR periods are used to create T2-weighted pictures. A third kind of MRI is the Fluid-Attenuated Inversion Recovery (FLAIR) MRI. This type of scan is a special inversion recovery sequence with a long inversion time. This removes signal from the cerebrospinal fluid (CSF) in the resulting images. Brain tissue on FLAIR images appears similar to T2 weighted images with grey matter brighter than white matter but CSF is dark instead of bright. The three types of scans can be observed on Figure 2.3.

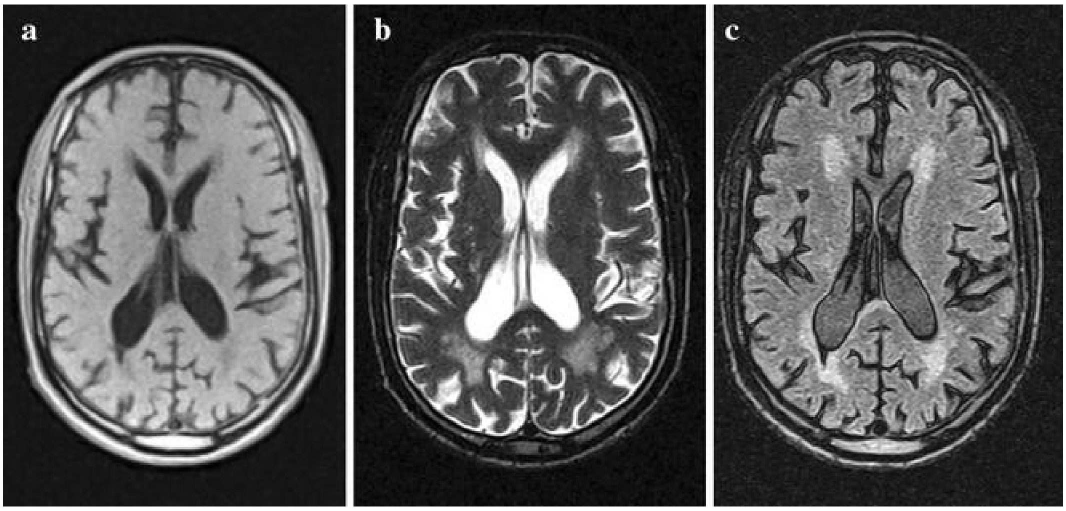


Figure 2.3: Example of magnetic resonance imaging (MRI) T1-weighted (a), T2-weighted (b) and FLAIR (c) axial images. From (Akinyemi et al., 2015).

Among T1-weighted MRI scans we will only focus on MPRAGE (Magnetisation Prepared Rapid Gradient Echo) images. Fast gradient echoes are characterized by their rapid sampling time, high signal intensity and image contrast while approaching steady state (the echo is collected during the time when tissues are experiencing T1 relaxation).

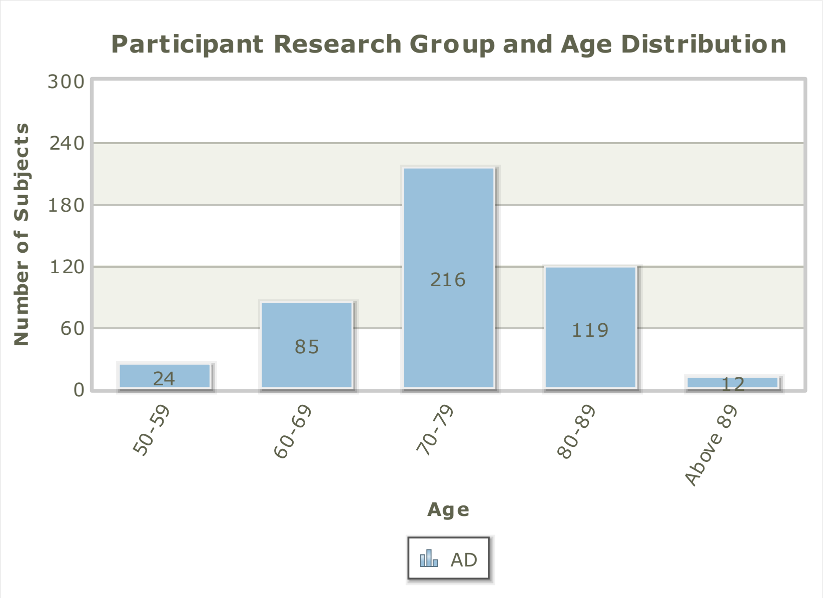


Figure 2.4: Demography of AD patients inside the ADNI database

2.3. The ADNI database

The Alzheimer's Disease Neuroimaging Initiative (ADNI) [1], launched in 2003 as a public-private partnership, is a multisite study that aims to improve clinical trials for the prevention and treatment of Alzheimer's disease (AD). It unites researchers around the globe with study data as they work to define the progression of AD. Data from MRI and PET scans, genetic testing, cognitive assessments, CSF and blood biomarkers, and other sources are gathered, verified, and used by ADNI researchers as disease predictors. Since its launch more than a decade ago, the landmark public-private partnership has made major contributions to AD research, enabling the sharing of data between researchers around the world.

2.4. The Hippocampus

The hippocampus is a small, curved formation in the brain that plays an important role in the limbic system. The hippocampus is involved in three primary functions: the formation of new memories, learning, and emotions. The hippocampus is one of the brain areas affected by Alzheimer's disease. In the early stages of AD, the hippocampus shows rapid loss of its tissue, which is associated with the functional disconnection with other parts of the brain.

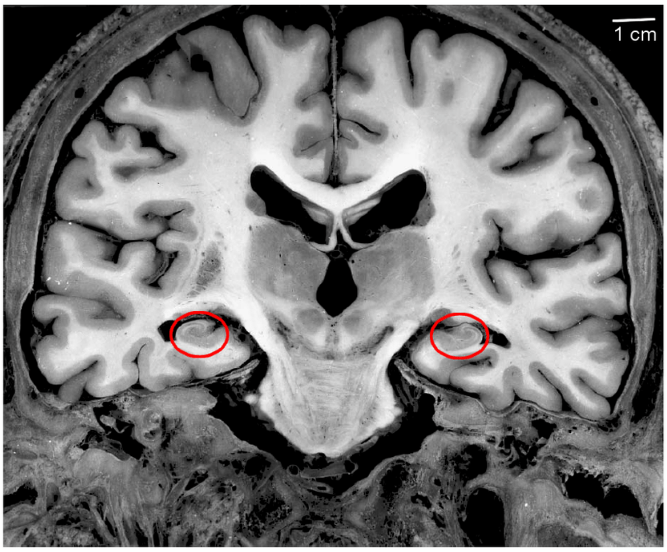


Figure 2.5: Coronal image of the hippocampus (red circles) in a brain

[1] `https://adni.loni.usc.edu/`

In the progression of AD, atrophy of medial temporal and hippocampal regions are the structural markers in magnetic resonance imaging. However, even for an expert neuroradiologist, tracing the hippocampus and measuring its volume is a time consuming and extremely challenging task. Accordingly, the development of reliable fully-automated algorithms is of paramount importance.

Chapter 3

Deep Learning

The number of applications using machine learning methods has increased steadily since its inception in the 1960s. Today, the science of machine learning affects every aspect of our life. In fact, it plays a role in how movies are recommended to us, how credit card theft is found, how different persons are identified in pictures, and how marketing efforts are oriented. The field of Statistics is constantly challenged by the problems that science and industry bring to its door. With the evolution of computers and the information age, statistical problems have exploded both in size and complexity. Vast amounts of data are being generated in many fields, and the statistician's job is to make sense of it all: extract important patterns/trends, and understand what the data is telling us. We call this "data learning" or Artificial Intelligence.

Most learning problems can be roughly categorized as either supervised or unsupervised. Indeed, computer scientists have still not been able to create an artificial general intelligence (AGI) totally autonomous and capable of general intelligent action including abstract learning.

On the one hand, supervised learning is a machine learning approach that is defined by its use of labelled datasets. These datasets are designed to train or "supervise" algorithms into classifying data or predicting outcomes accurately. Using both the labels associated to the inputs and the outputs, the model can measure its accuracy and learn over time.

On the other hand, unsupervised learning algorithms are not provided pre-assigned labels or scores for the training data. They must then self-discover any naturally occurring patterns in that training data set. This new family of machine learning methods includes among others:

- Clustering, where the algorithm tries to group its training examples into categories with similar features.
- Principal component analysis, where the algorithm finds ways to compress the training data set by identifying which features are most useful for discriminating between different samples.

3.1. Supervised Learning

As we will be dealing with labeled dataset in this work, we found ourselves in a supervised learning context. More formally, we are looking for a function f describing the relationship between the values of m inputs $(x_1, x_2, ..., x_m) = \boldsymbol{x} \in \mathcal{X}$, with $\mathcal{X}$ being the input space, and the value of an output variable $y \in \mathcal{Y}$, with $\mathcal{Y}$ the output space, such that:

$$f : \mathcal{X} \to \mathcal{Y}$$

This function f will then be exploited to predict the output of previously unseen input data. Most machine learning problems, whether in a supervised or unsupervised context, can essentially be boiled down to the minimization of a mathematical function called the loss. The loss function will essentially quantify how much the algorithm's predictions are wrong compared to the ground truth provided by the training dataset.

$$\mathcal{L}(f(\boldsymbol{x}), y)$$

Indeed, during the training phase of a particular algorithm, each of its prediction for a given sample will be compared to the actual outcome of that sample. Every mistake will eventually be sum up in the loss function. This yields the mathematical goal of the algorithm, minimizing that function. Every model has parameters so the minimization is done by tuning, adjusting those parameters accordingly to reach an optimum of the loss function.

In the case of a binary classification model, a common loss function is the binary cross entropy.

$$\mathcal{L} = y \cdot \log f(\boldsymbol{x}) + (1 - y) \cdot \log(1 - f(\boldsymbol{x})) \tag{3.1}$$

3.2. Deep Learning

A subset family of machine learning methods are based on artificial neural networks. Such networks allow computer to learn from experience and understand the world in terms of a hierarchy of concepts, with each concept defined through its relation to simpler ones. By gathering knowledge from experience, this approach avoids the need for human operators to formally specify all the knowledge that the computer needs. If we draw a graph showing how simple concepts are built on top of each other in order to enable computers to learn more complicated concepts, the graph will be deep with many layers. For this reason, this approach is refer to as deep learning. In this section, the different model architectures are briefly explained from the plain vanilla multi-layer perceptron to the convolutional neural network.

3.2.1. Multi-layer perceptron

The intuitive idea behind deep learning models is that they are engineered systems inspired by the biological brain and therefore are intended to be computational models of biological learning, mimicking how learning happens or could happen in the brain. The quintessential deep learning models are called **Multilayer Perceptron** (MLPs) or feedforward neural network (see Figure 3.1). Their goal, like any supervise learning algorithm, is to approximate some function f^*. For example, for a classifier, $y = f^*(\mathbf{x})$ maps an input $\mathbf{x} \in \mathcal{R}^d$ where d is the dimension of the input to a category $y \in \Delta^C$ where C is the number of classes. A feedforward network defines a mapping $y = f(\mathbf{x}; \mathbf{w}, \mathbf{b})$ and learns the value of the parameters $\mathbf{w}$ and $\mathbf{b}$, respectfully weights and bias, that result in the best function approximation.

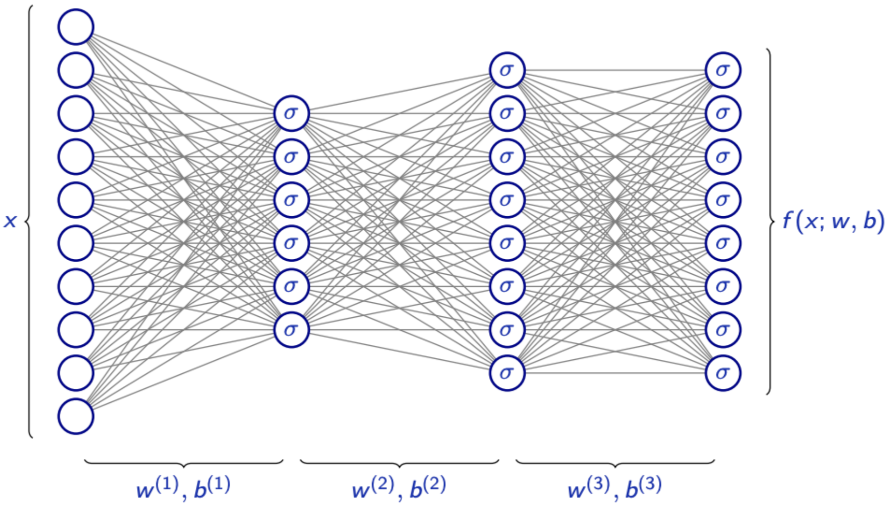


Figure 3.1: Multi-Layer Perceptron [1]

Each neuron has its own weights $\mathbf{w}$ and bias $\mathbf{b}$. If we take as example a neuron i from the first hidden layer, its value will be computed as $h(i) = \sigma(\mathbf{w}^{1,i}\mathbf{x} + b^{1,i})$ where $\mathbf{w}^{1,i}$ represent a weight vector between the previous layer (the input) and the i^{th} neuron, $b^{1,i}$ represent the bias (a scalar value) associated with the i^{th} neuron of the first hidden layer. While σ represents the activation function.

3.2.2. Convolutional neural network

Convolutional networks (Bengio and Lecun, 1997), also known as CNNs, are a specialized kind of neural network for processing data that has a known grid-like topology. This means that these neural networks are equipped with inductive biases tailored for vision. In statistics and machine learning, inductive biases refers to the set of assumptions that the algorithm or model uses to predict outputs. In the case of CNN, those assumptions are ones such as *invariance to translation* which can be a useful property if we care more about whether some

[1] Credit: François Fleuret (`https://fleuret.org/dlc/`

feature is present than exactly where it is. For instance, when determining whether an image contains a face, we do not need to know the location of the eyes with pixel-perfect accuracy. We just need to know that there is an eye on the left side of the face and an eye on the right. Another relevant assumption is the *hierarchical compositionality* which express the idea that each images is composed of smaller patterns that could be easier to spot. For example, an ”8” is composed of two circles, one on the bottom and one on the top.

CNNs have been tremendously successful in practical applications. The name ”convolutional neural network” indicates that the network employs a mathematical operation called **convolution** which is a specialized kind of linear operation (see equation 3.2) .

$$(x \circledast u)[i] = \sum_{m=0}^{w-1} x_{m+i} u_m \tag{3.2}$$

1 dimensional convolution

Consider an input signal of width W with a kernel of smaller size w. Figure 3.2 illustrate how Convolving this input with the kernel consists in repeating a dot product between the kernel (green) and a vector of same length (gray) extracted from the input vector.

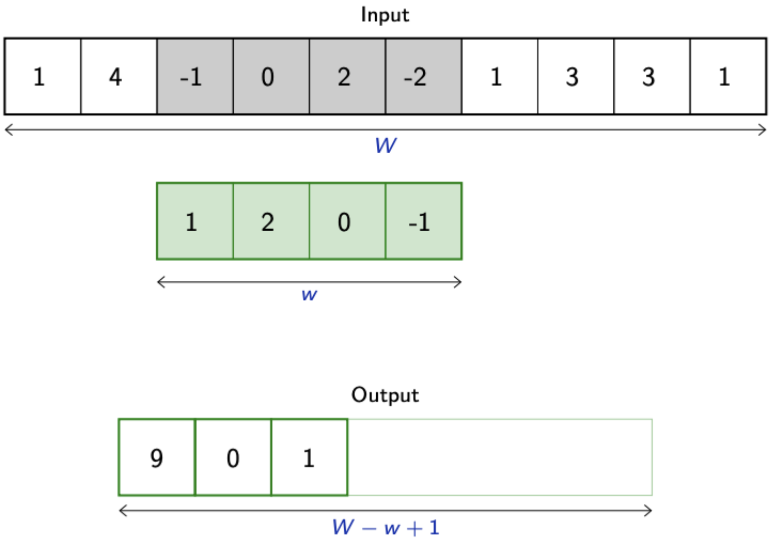


Figure 3.2: 1-dimensional convolution [2]

One can notice two main results in this operation. First, the structure of the signal is preserved: convolving a 1D signal with a 1D kernel produces a 1D output. Second, the resulting tensor is smaller than the input.

[2] Credit: François Fleuret (`https://fleuret.org/dlc/`

2 dimensional convolution

We are now dealing with an input of 3 dimensions $C \times H \times W$. This could represent a 2 dimensional input with C channels such as RGB images. It could also be a 3 dimensionnal input with 1 channel such as medical scan. The kernels applied to this input must have C channels has well as the input signal. The green (red) output result from the convolution of the green (red) kernel with the input signal.

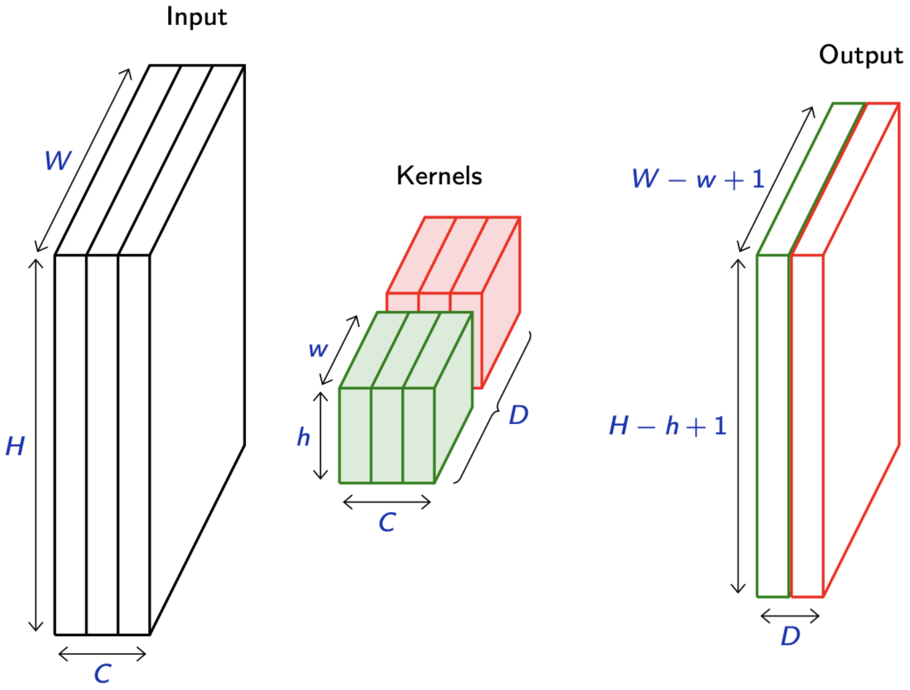


Figure 3.3: 2 dimensional Convolution Layer with two kernels [3]

Convolution leverages three important ideas that can help improve the performance of a neural network:

- Sparse representation
- Parameter sharing
- Equivariant representation

On top of that, convolution provides a means for working with inputs of variable size.

Fully connected layers use matrix multiplication by a matrix of parameters with a different parameter describing the interaction between each input unit and each output unit. This means that every output unit interacts with every input unit. Convolutional networks, however, typically have **sparse interactions**.

[3] Credit: François Fleuret (`https://fleuret.org/dlc/`

This is accomplished by making the kernel smaller than the input. For example, when processing an image, the input image might have thousands or millions of pixels, but we can detect small, meaningful features such as edges with kernels that occupy only tens or hundreds of pixels. This implies that we need to store fewer parameters, which both reduces the memory needs of the model and enhance its statistical efficiency. It also means that computing the output requires fewer operations. These improvements in efficiency are usually quite large. If there are m inputs and n outputs, then matrix multiplication requires $m \times n$ parameters, and the algorithms in practise have $\mathcal{O}(m \times n)$ runtime. By limiting the number of connections each outputs may have to k, then the sparsely connected approach requires only $k \times n$ parameters and $\mathcal{O}(k \times n)$ runtime.

Parameter sharing refers to using the same parameter for more than one function in a model. In a traditional neural net, each element of the weight matrix is used exactly once when computing the output of a layer. In a convolutional neural net, each member of the kernel is used at every position of the input, except perhaps some of the boundary pixels. The parameter sharing used by the convolution operation means that rather than learning a separate set of parameters for every location, we learn only one set.

Convolution has three additional parameters : padding, strides and dilation.

Padding

The padding which specified the size of a zeroed frame added around the input. It is useful to control the spatial dimension of the feature map, for example to keep it constant across layers (Figure 3.4).

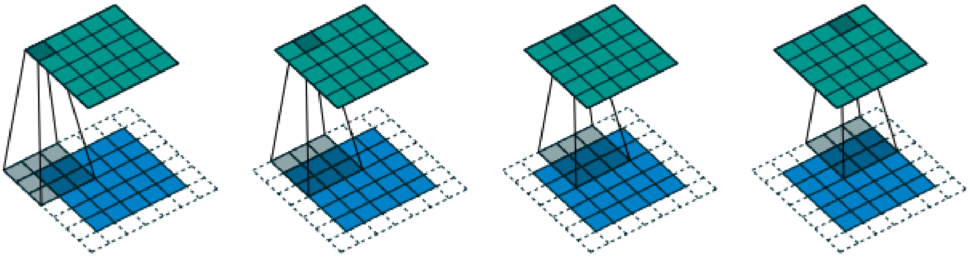


Figure 3.4: Convolving a 3×3 kernel over a 5×5 input using half padding and unit strides [4]

Strides

The stride specifies a step size when moving the kernel across the signal. It is useful to reduce the spatial dimension of the feature map by a constant factor (Figure

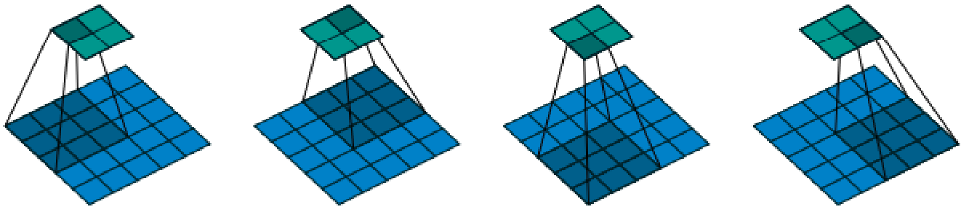


Figure 3.5: Convolving a 3 × 3 kernel over a 5 × 5 input using 2 × 2 strides [5]

Dilation

The dilation modulates the expansion of the filter without adding weights but by adding rows and columns of zeros between coefficients. Having a dilation coefficient greater than one increases the units receptive field size without increasing the number of parameters.

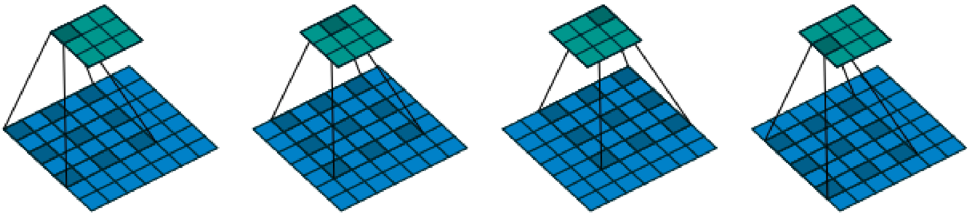


Figure 3.6: Convolving a 3 × 3 kernel over a 7 × 7 input with a dilation factor of 2 [6]

Pooling

A typical layer of convolutional network consists of three main stages :

- The layer performs several convolutions in parallel to produce a set of linear activations.
- Each linear activation is run through a nonlinear activation function, such as the rectified linear activation function. This stage is sometimes called the **detector stage**.
- Finally, a **pooling function** is used to modify the output of the layer further.

A pooling function replaces the output of the net at a certain location with a summary statistic of the nearby outputs. For example, the max pooling (Zhou and Chellappa, 1988) operation reports the maximum output within a rectangular neighborhood. There are other popular pooling functions such as the average of a rectangle neighborhood or a weighted average based on the distance from the central pixel.

[6] Dumoulin and Visin (2016)

Figure 3.7 shows an example of max-pooling in 1 dimension. In all cases, pooling helps to make the representation approximately invariant to small translations of the input. Which means that if the latter is translated by a small amount, the values of most of the pooled outputs will not change. The use of pooling can be viewed as adding an infinitely strong prior that the function the layer learns must be invariant to small translation. Just like the other inductive bias, when this assumption is correct, it can greatly improve the statistical efficiency of the network.

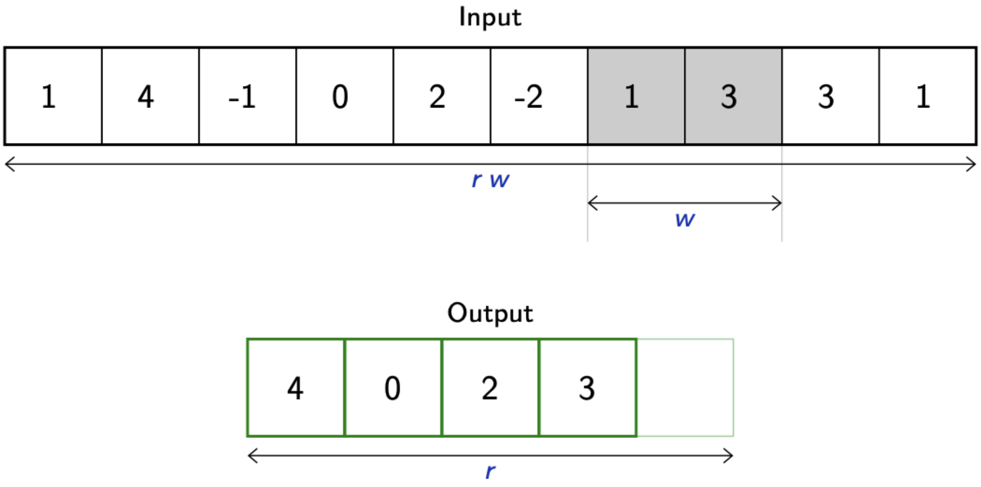


Figure 3.7: Example of max-pooling in 1 dimension with a kernel of size 2 [7]

[7] Credit: François Fleuret (`https://fleuret.org/dlc/`

3.3. Context Attention

Natural language processing is an important sub-field of AI and has historically been tackled with encoder-decoder based neural machine translation system. These systems used Recurrent neural network (RNNs) to encode and decode the data. RNNs process an input sequence one element at a time, maintaining in their hidden units a ‘state vector’ that implicitly contains information about the history of all the past elements of the sequence.

In 2017, an article intitled ”Attention is all you need” (Vaswani et al., 2017) introduced a new type of architecture containing only attention mechanisms : The Transformer. Their work demonstrated that for translation tasks the Transformer could be trained significantly faster than architectures based on recurrent or convolutional layers.

The following will explain the encoder and the decoder’s job sequentially. On a high level the encoder maps an input sequence into an abstract continuous representation that holds all the learned information regarding that input. The decoder then takes that continuous representation and step by step generates a single output while also being fed the previous output.

3.3.1. The encoder

Input embedding

The first step of a transformer network is to feed our input into a word embedding layer which can be thought of as a lookup table to grab a learned vector representation of each word. Since neural networks learn with numbers, each word maps to a vector with continuous values to represent that word. The dimension of the input vector is $d_{model} = 512$.

Positional Encoding

Because a transformer encoder has no recurrence like recurrent neural networks, we must inject some information about the relative or absolute position of the tokens in the sequence. This is achieved by using positional encoding and adding it into the input embeddings. This means the position embeddings vector and the input vectors must be of the same dimension.

$$PE_{pos,2i} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

$$PE_{pos,2i+1} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

Where pos is the position of the token in the sentence and i is the dimension index of the encoding vector of dimension d_{model}. This successfully gives the network information on the position of each vector. The sin and cos were chosen because they have linear properties the model can easily learn to attend to.

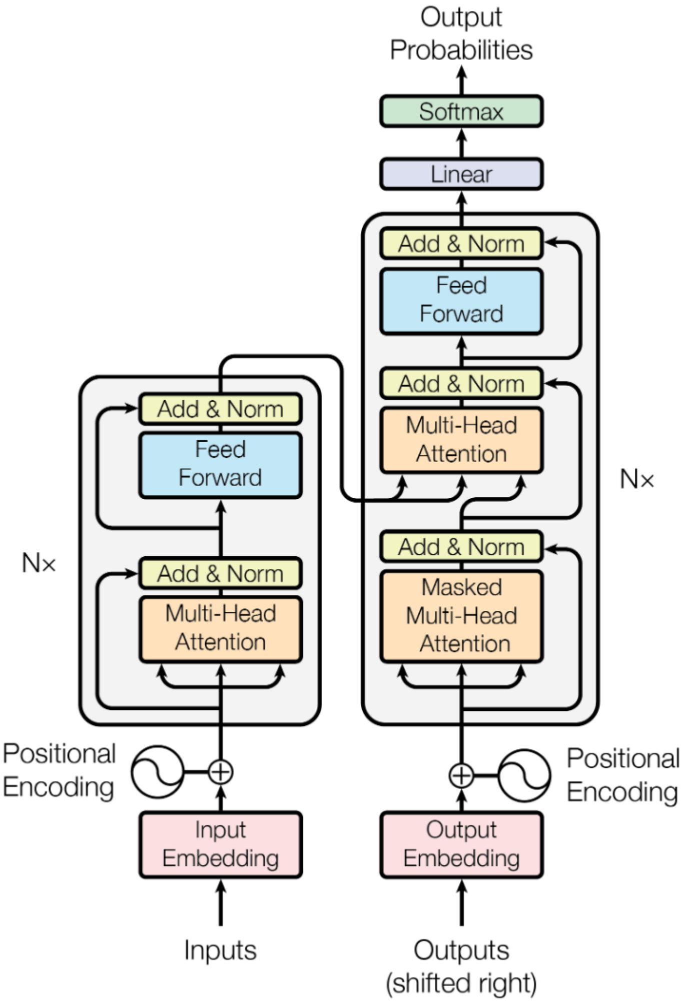


Figure 3.8: The Transformer

Multi-headed Attention

In the beginning of the encoder, multi-headed attention (Figure 3.9) applies a specific attention mechanism called self attention which allows a model to associate each individual word in the input to other words in the input. To achieve self attention we feed the input into three distinct fully connected layers to create the query, key and values vectors.

The queries and keys undergoes a dot product multiplication to produce a score matrix which determines how much focus should a word put on other words. Each word will have a score related for every other words. The higher the score the more the focus, this is how queries are mapped to keys. Then the score gets scaled down by getting divided by $\sqrt{d_k}$. This is to allow for more stable gradients as multiplying values can have exploding effects. Next you take the softmax of the scaled scores to get the attention weight which gives you probability values between 0 and 1.

$$softmax(x)_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

By doing the softmax the higher scores gets heightened and the lower scores are suppressed. This allows the model to be more confident on which words to attend to. Then, the attention weights gets multiplied by the value vector to get an output vector. The higher softmax scores will keep the value of the words the model learned is more important. The lower scores will drown out the irrelevant words.

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

To make this multi-headed attention computation we need to split the queries, keys and values into h vectors before applying self attention. Split vectors then goes to the same self attention process individually. Each one is called a head. Each head produces an output vector which gets concatenate into a single vector before going through the final linear layer. In theory, each head would learn something different therefore giving the encoder model more representation power.

To sum it up, multi-headed attention is a module in the transformer network that computes the attention weights for the input and produces an output vector with encoded information on how each words should attend to all other words in the sequence.

Residual Connection, Layer Normalization & Pointwise Feed Forward

Next step, the multi-headed attention output vector is added to the original vector. This is called a residual connection. The output of the residual connection goes through a layer normalization before the normalized residual output gets fed into a point-wise feed-forward network for further processing. The point-wise feed-forward networks are a couple of linear layers with a Relu activation in between. This output is again added to the input of the feed-forward network and further normalized.

The residual connections help the network train by allowing gradients to flow through the network directly while the normalization layers are used to stabilize the network which results in substantially producing the training time necessary. Finally the point-wise feed-forward network is used to further process the attention output potentially giving it a richer representation.

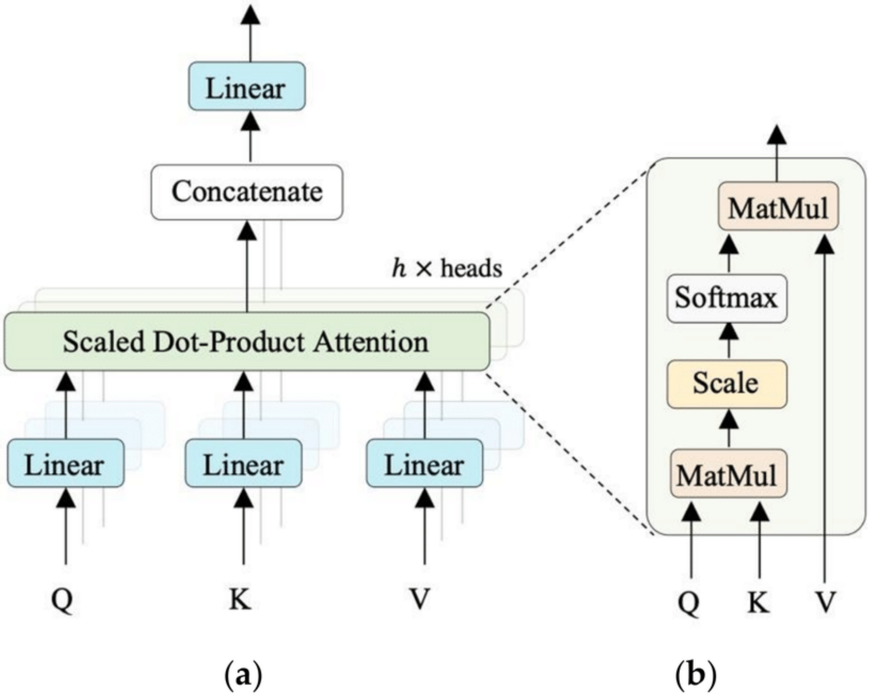


Figure 3.9: Multi-headed attention

3.3.2. The decoder

The decoder's job is to generate text sequences. The decoder has similar sub-layers as the encoder : 2 multi-headed attention layers, a point-wise feed-forward layer with residual connection and layer normalization after each sub layers. The decoder ends with a linear layer that acts like a classifier and a softmax to get the word probabilities.

The decoder is said to be autoregressive which specifies that the output variable depends linearly on its own previous values. In this case, the decoder takes a list of previous outputs as inputs as well as the encoder's output that contains the attention information from the inputs.

Output embedding & positional encoding

The input goes to an embedding layer and a positional encoding layer to get positional embeddings in the same way as the encoder.

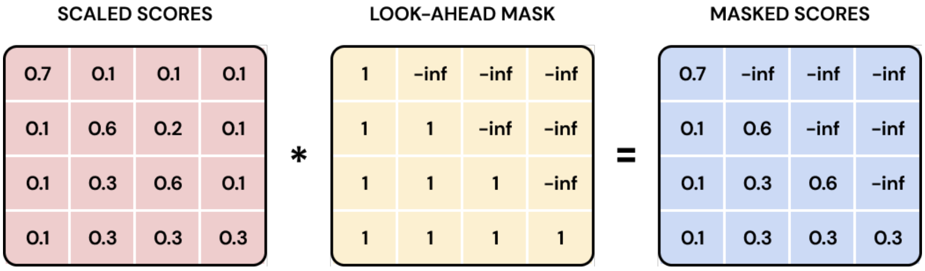
Masked Multi-headed attention

The positional embeddings gets fed into the first multi-headed attention layer which computes the attention scores for the decoder's input. However, the multi-headed attention layer operates slightly differently than during the encoding process. Since the decoder is autoregressive and generates the sequence word by word, we need to prevent it from conditioning to future tokens.

We need a method that prevents computing attention scores for future words, this method is called masking. We must apply a look-ahead mask. This mask is added before calculating the softmax and after scaling the scores. The mask consists in a matrix with the same size as the attention scores filled with values of zeros and negative infinities. When adding this mask to the scaled attention scores, we get a matrix of scores with the top right triangle filled with negative infinities (see Figure 3.10). Taking the softmax of the masked scores the negative infinities get zeroed out leaving a zero attention score for future tokens. This essentially tells the model to put no focus on future words.

The output of the first multi headed attention is a masked output vector with information on how the model should attend on the decoders inputs. For the second multi headed attention layer, the encoder's output are the queries and the keys and the first multi headed attention layer outputs are the values. This process allows the decoder to decide which encoder's input is relevant to put focus on.

The output of the second multi headed attention goes through a point wise feed forward layer for further processing. Then the output of the feed forward later goes through a final linear layer that acts as a classifier. The classifier is as big as the number of classes we have. For instance, if we gave $10,000$ classes for $10,000$ words, the output of that classifier will be $10,000$. The output of the classifier then gets fed into a softmax layer which produces probabilities scores between 0 and 1 for each class. Taking the index of the highest probability score and that equals our predicted word. The decoder then takes the output and adds it to the list of decoder inputs before decoding again until an end token is predicted.



SCALED SCORES

0.7	0.1	0.1	0.1
0.1	0.6	0.2	0.1
0.1	0.3	0.6	0.1
0.1	0.3	0.3	0.3

*

LOOK-AHEAD MASK

1	-inf	-inf	-inf
1	1	-inf	-inf
1	1	1	-inf
1	1	1	1

=

MASKED SCORES

0.7	-inf	-inf	-inf
0.1	0.6	-inf	-inf
0.1	0.3	0.6	-inf
0.1	0.3	0.3	0.3

Figure 3.10: A Look-Ahead Mask turns Multi-Head Attention into Masked Multi-Head Attention [8]

[8] Credit : `https://www.revistek.com/posts/transformer-architecture`

3.4. Vision Transformer

Since its first appearance, the Transformer has become the model of choice for natural language processing tasks. Thanks to its computational efficiency and scalability, it has become possible to train models of unprecedented size (Brown et al., 2020).

Naive application of self-attention to images would require that each pixel attends to every other pixel. With quadratic cost in the number of pixels, this does not scale to realistic input sizes. In 2020, the Google research team introduced the Vision Transformer which interpret an image as a sequence of 16 × 16 patches and process it by a standard Transformer encoder as used in NLP (Dosovitskiy et al., 2020).

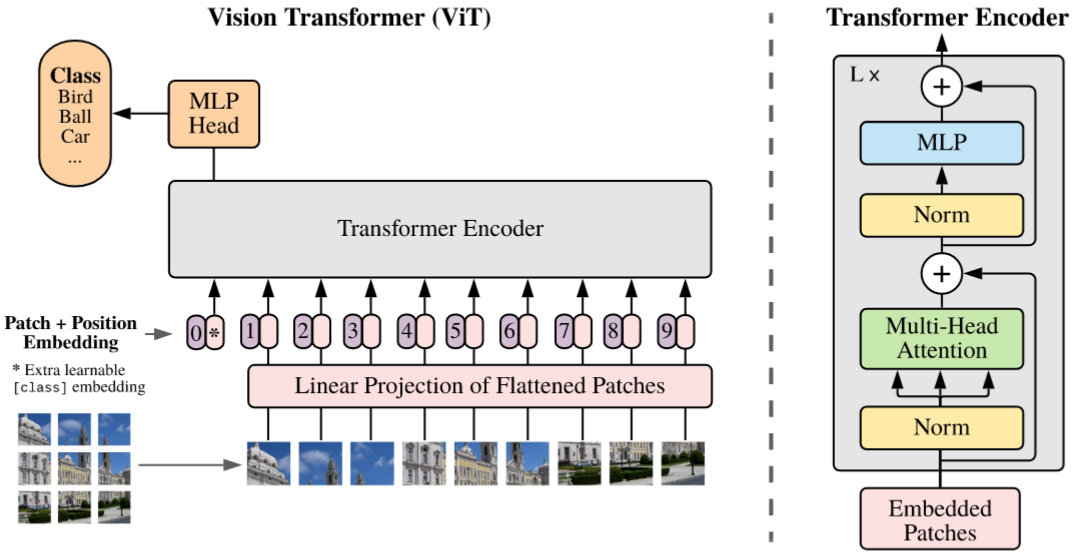


Figure 3.11: Architecture of the Vision Transformer (Dosovitskiy et al., 2020)

Inspired by the Transformer scaling successes in NLP, their work applied a standard transformer directly to images, with the fewest possible changes. To do so, the image is split into fixed-size 2 dimensional patches. The transformer uses constant latent vector size D through all of its layers, so the patches are flattened and mapped to D dimensions with a trainable linear projection. This output of this projection is refered to as the patch embeddings.

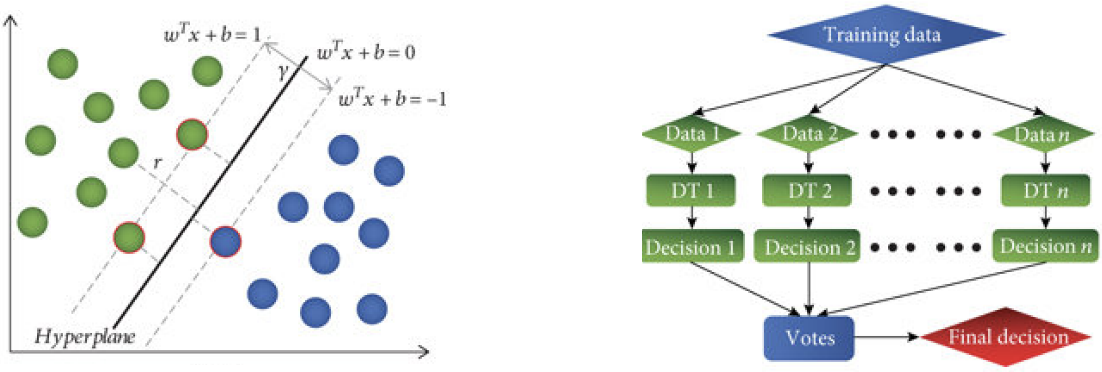
Positional embeddings are added to the patch embeddings to retain positional information using a standard learnable 1 dimensional position embeddings. The resulting sequence of embedding vectors is fed as an input to a standard Transformer encoder. In order to perform classification, we use the classic approach of adding an extra learnable "classification token" to the sequence, whose state at the output of the transformer encoder serves as the image representation. Let $\mathbf{z}_L^0$ be the classification token at the output of transformer encoder repeated L times. Both during pre-training and fine-tuning, a classification head is attached to $\mathbf{z}_L^0$. This classification head is implemented by a MLP with one hidden layer at pre-training time and by a single linear layer at fine-tuning time.

Just like the one described in subsection 3.3.1, the transformer encoder in ViT consists of alternating layers of multihead self attention and MLP blocks. Layer normalization is applied before every block, and residual connections after each block. The MLP contains two layers with a GELU activation function.

3.5. Related Work

3.5.1. Classical Machine Learning

One of the most popular machine learning algorithms used in the neuroimaging field is the Support Vector Machines (SVM) (Hearst et al., 1998). For a binary classifier, SVM will map training examples to points in space so as to maximise the width of the gap between the two categories. Indeed, this intuitive method has proven to be very useful when dealing with problems with high dimension and small sample. The Random Forests method (Breiman, 2001) is another technique renowned for its state-of-the-art results on machine learning problems with huge dimension-sample ratios and has also provided good results (Wehenkel et al., 2018).



(a) SVM

(b) Random Forest

Figure 3.12: Machine learning in neuroimaging (Yin et al., 2020)

Decision Tree is one of the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart-like tree structure, where each internal node denotes a test on an feature, each branch represents an outcome of the test, and each leaf node, called terminal node, holds a class label. Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction.

A classification framework can consist of other elements in addition to the classification algorithm as an entity:

- feature extraction
- feature selection
- dimensionality reduction

Classical methods based on feature vectors directly extracted from the neuroimages are called voxel-as-feature (VAF) based methods. Due to the large number of brain voxels recovered from a single neuroimage (between $50,000$ and $300,000$ features) and the small number of samples (from 50 to a few hundreds), classifiers based on neuroimages frequently require one or more feature reduction approaches. Since there are so few instances and numerous potential explanatory variables, a machine learning system will find it challenging, as efficient as it could be, to reliably identify highly discriminative features. For MRI images, there is no point in working directly with voxels and thus feature extraction is necessary. For instance, information about regional volumes and shapes or tissue densities can be computed from the MRI. Brain atrophy is thus estimated with density maps of grey matter, which are provided by voxel-based morphometry methods (Ashburner and Friston, 2000). Classifiers can thus be learnt either using the map as a whole or by performing feature selection.

To avoid a feature selection procedure, some works directly select brain regions which were identified in previous literature as relevant for the disease, like the hippocampus for Alzheimer disease. However, this prevents the discovery of new regions of interest. We may categorize feature selection procedures into three main groups : the wrapper methods, the filter methods and the embedded methods.

Wrapper methods require a lot of computer power because they calculate classifier accuracy for all possible (or many) feature subsets. With the quantity of variables in neuroimaging data, this is not feasible. Filter methods can immediately highlight features that have no connection to the output by using a correlation criterion or another equivalent measure. Embedded methods include in their machine learning algorithm a feature selection process, like LASSO or CART decision tree (Tibshirani, 2011; Leo Breiman, 1984).

Some unsupervised learning techniques are sometimes used in machine learning with neuroimaging such as Principle component analysis or partial least square. However these feature reduction methods reduce the interpretability of the classifier.

3.5.2. Deep Learning

Deep learning has lately gained popularity and proven to be an effective technique in the field of medical imaging, thanks to the quick advancement of computer science and the accumulation of clinical data. The general applications of deep learning in medical imaging are mainly feature extraction, image classification, object detection, segmentation and registration (Litjens et al., 2017).

Several deep learning approaches have recently been proposed as diagnostic aids for Alzheimer's disease, assisting doctors in making informed medical decision. A simple search for "Alzheimer's disease" and "deep learning" on the website PubMed[9], which is a free resource supporting the search and retrieval of biomedical and life sciences, tells us that an approximate 667 different papers have been release with those keywords (See Figure 3.13).

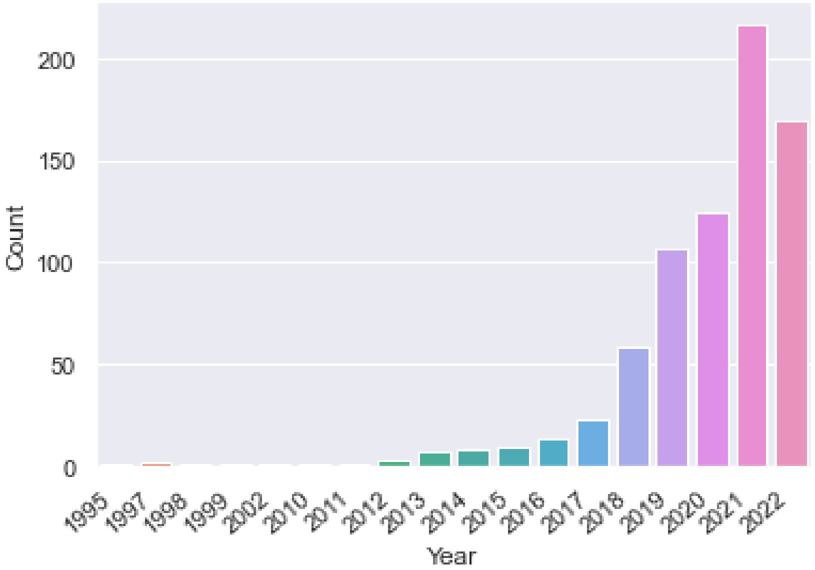


Figure 3.13: Evolution of articles involving "Alzheimer's disease" and "Deep learning" on PubMed along the years.

In the rest of this section we present a subset of the studies that are closely related to this work. The articles discussed are listed along with their respective method and accuracy on Table 3.1 Those studies operated on several databases, some of them with different annotations. In the following, Normal control (NC) and healthy control (HC) both refers to a patient not suffering from dementia.

Lu et al. (2018) proposed a novel multimodal deep neural network with a multistage technique to identify people with dementia. This method provides 82.4% accuracy in Mild Cognitive Impairment (MCI) prediction and those patients later exposed to Alzheimer's disease in three

[9] https://www.ncbi.nlm.nih.gov/

years. The model achieves a sensitivity of 94.23 % for the Alzheimer's disease class and an accuracy of 86.3 % for the non-demented class.

Gupta et al. (2019) proposed a diagnosis method for the classification of AD using the ADNI and National Research Center for Dementia (NRCD) dataset by combined features from cortical, subcortical, and hippocampus region from MRI images which achieve the better accuracy of 96.42% for classification of AD vs Healthy Control (HC).

Ahmed et al. (2019) proposed the ensemble CNN model for feature extractor and SoftMax classifier to diagnose AD diagnosis. This model prevents overfitting and achieves an accuracy of 90.05% by using the left and right hippocampus area in MRI images.

Basher et al. (2021) came up with a method to localize the target regions from large MRI volume to automate the process. Based on the left and right hippocampi, the method achieves the accuracies of 94.82% and 94.02%.

Nawaz et al. (2020) presented a pretrained Alexnet model to classify the stages of AD to address the class imbalance model. The pretrained model is used as the feature extractor and classified using Support Vector Machine (SVM), K-nearest neighbour (KNN) and Random Forest (RF) with the highest accuracy of 99.21%.

Ieracitano et al. (2018) propose a data-driven method for distinguishing subjects with AD, MCI, and HC by analyzing non-invasive recordings of EEG. The Power Spectral Density of the EEG traces of 19 channels reflects their corresponding spectral profiles in 2D Grayscale images. Then the CNN model is used to classify the binary class and multiclass from the 2D images with an accuracy of 89.8% and 83.3%, respectively.

Jain et al. (2019) uses a pretrained VGG16 model for feature extraction, which uses a FreeSurfer for pre-processing, selecting MRI slices using Entropy and classification using transfer learning named PFSECTL mathematical model. The researchers were able to achieve 95.73% accuracy for classifying Normal Control (NC), Early MCI (EMCI), and Late MCI (LMCI) using the ADNI database.

Mehmood et al. (2021) uses tissue segmentation to extract Grey Matter (GM) from each subject. The model attains the classification accuracy of 98.73% for AD vs NC and 83.72% for EMCI vs LMCI patients.

Shi et al. (2018) proposed deep polynomial network which performs well for both small and large dataset to diagnose AD. The model achieves an accuracy of 55.34% for both binary and multi-classification using the ADNI dataset.

Liu et al. (2019) proposed Siamese neural networks to investigate the discriminative capacity of whole-brain volumetric asymmetry. The team used the MRI Cloud process to create low-dimensional volumetric features for pre-defined atlas brain structures, as well as a unique non-linear kernel method to normalize features and eliminate batch effects across datasets and populations. The networks achieve a balanced accuracy of 92.72% for the classification of MCI and AD using the ADNI dataset.

Wang et al. (2019) presented AD and MCI using a 3D ensemble model convolutional networks. 3D-DenseNets optimized by using a probability-based fusion approach. The model achieves the classification accuracy of 97.52% using the ADNI dataset.

K et al. (2019) uses the grey wolf optimization technique with a decision tree, KNN, and CNN model to diagnose AD and achieve a 96.23 % accuracy.

Janghel and Rathore (2021) proposed a pretrained VGG16 to extract the features of AD from the ADNI database. For classification, they used SVM, Linear Discriminate, K means clustering and decision tree algorithm. They reach a 99.95% accuracy in functional MRI images and an average accuracy of 73.46 % for the PET images.

Ge et al. (2019) proposed a 3D multiscale deep learning architecture to learn AD features. On a subject segregated random brain scan-partitioned dataset, the system achieved a test accuracy of about 93.53%, with an average accuracy of 87.24 %.

Bi and Wang (2019) presented a Spike Convolutional Deep Boltzmann Machine model for early AD detection with hybrid feature maps and a multi-task learning technique to prevent overfitting. Sarraf et al. (2016) presented a deep learning pipeline where the CNN model is trained with many training images to perform feature classification on the scale and shift-invariant processes. The model achieves 94.32% and 97.88% for functional MRI and MRI images.

Afzal et al. (2019) address the class imbalance problem in detection of AD by data augmentation framework and achieves the classification accuracy of 98.41% in a single view and 95.11% in 3D view of OASIS dataset.

Table 3.1: Overview of the literature survey

Ref	**Method**	**Classes**	**Accuracy** (%)
(Lu et al., 2018)	Multiscale deep learning	Binary class (AD vs HC)	82.4
(Gupta et al., 2019)	Combined feature technique	Binary class (AD vs HC)	96.42
(Ahmed et al., 2019)	Ensemble model	Binary class (AD vs HC)	94.03
(Ieracitano et al., 2018)	2D CNN	Binary class (AD vs HC)	89.08
(Jain et al., 2019)	CNN	Three class (NC, EMCI, LMCI)	95.73
(Mehmood et al., 2021)	Transfer Learning	Binary class (AD vs HC)	98.73
(Shi et al., 2018)	Deep polynomial network	Binary class (AD vs HC)	55.34
(Liu et al., 2019)	Siamese network	Binary class (MCI vs AD)	94.03
(Wang et al., 2019)	3D CNN	Binary class (AD vs HC)	89.08
(Ge et al., 2019)	3D CNN	Binary class (AD vs HC)	98.73
(Afzal et al., 2019)	3D View model	Binary class (AD vs HC)	55.34

Chapter 4

Methodology

This chapter consists in analysing and detailing the different steps undertaken empirically for this work. First we will talk about the preprocessing pipeline and how an MRI file straight out of ADNI is transformed in order to fit as an input for our networks. Then, we will introduce those networks : A small CNN, a large CNN and a Vision Transformer. We will describe their architecture as well as their implementation [1].

4.1. Preprocessing

Data preprocessing transforms the data into a format that can be processed in data mining, machine learning, and other data science tasks more quickly and efficiently. To ensure reliable findings, the techniques are typically applied at the very beginning of the machine learning and AI development pipeline.

The most effective way to display data for machine learning and deep learning algorithms is to highlight the key information needed to solve a problem. Data transformation, data reduction, feature selection, and feature scaling techniques used in feature engineering help reorganize raw data into a format suitable for specific kinds of algorithms. This can considerably lower the processing power and time needed to train a new machine learning algorithm or perform an inference against it.

Table 4.1: Demographic details of my own ADNI dataset (μ is for mean and σ is for standard deviation)

	Sex			Age		
Diagnosis	#	M	F	μ	σ	Range
Alzheimer Disease (AD)	322	**163**	159	75.62	7.84	55 – 91
Cognitively Normal (CN)	616	301	**315**	78.28	4.86	70 – 94
Mild cognitive impairment (MCI)	589	**402**	186	75.73	7.51	56 – 90

[1] All the codes produced for this work are available in this Github

For this work, the data that will be exploited will consist in only T1-weighted MP-RAGE MRI scans of men and women of different diagnosis. Table 4.1 contains all the data used for this work. Looking closely at this table, one can realize that it does not necessarily represent a good sample of a population. For instance several studies found that women were more likely to develop any dementia as well as Alzheimer disease (Beam et al., 2018). This pattern is consistent with women's survival to older ages compared to men. However, table 4.1 displays a larger number of men both in Alzheimer diagnosis and in mild cognitive impairment. Another important risk factor for Alzheimer's disease is advanced age (Guerreiro and Bras, 2015). Once again our sample does not reflect this correlation since cognitively unimpaired patients are older in this case. This will, however, not impact our inference models since none of those pieces of information are fed into them, only the MRI scan is relevant.

As described in section 2, MCI is an heterogeneous group and it can be furthermore classified according to its various clinical outcomes. In this work, we decided to partition MCI into two categories : progressive MCI (pMCI) and stable MCI (sMCI). Those groups are retrospective diagnostic terms based on the clinical follow-up according to the DSM-5 criteria (American Psychiatric Association, 2013). The term pMCI, refers to MCI patients who develop dementia in a 36-month follow-up, while sMCI is assigned to MCI patients who do not convert to a dementia stage. Distinguishing between pMCI and sMCI plays an important role in the early diagnosis of dementia, which can assist clinicians in proposing effective therapeutic interventions for the disease process. Figure 4.1 shows the number of patients at each follow up visit collected by ADNI.

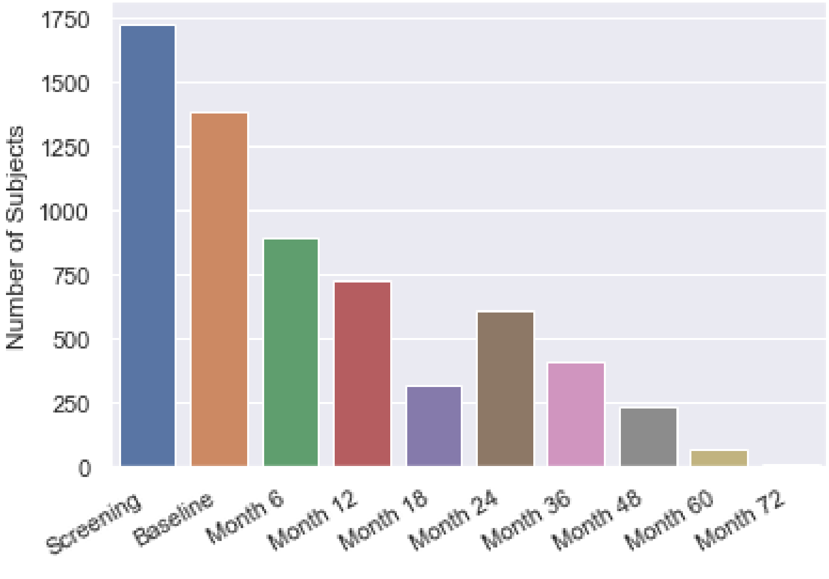


Figure 4.1: Participant visit distribution of MCI patients from ADNI

During each of the processing phase, one has to realize that every choice made will eventually result in a loss of information relative to the original raw data. It is the practitioner's job to evaluate the benefit risk balance of abandoning some information, which according to his hypothesis might be of few relevance, in order to save computing power, resources and time.

All the medical images for this work were downloaded in the NIfTI format (Neuroimaging Informatics Technology Initiative). [2] `SimpleITK` was used for transforming the `.nii` images provided by ADNI into numpy arrays. In order to understand the data preparation steps, let us follow patient number 24602 throughout his preprocessing pipeline.

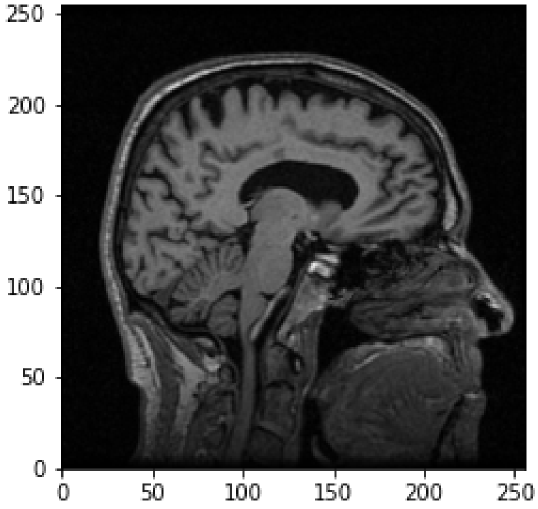


Figure 4.2: Sagittal view of an ADNI patient
Original size : (256, 256, 180) voxels

This image has a resolution of $(256, 256, 180)$ This means it has $65,536$ voxels for each one of the 180 sagittal slices, cf. Figure 4.2, and a total input of $11,796,480$ pixels for this single image. This is of course way too big to be digested by a neural network.

The original neuroimages downloaded from ADNI were in wildly different shape but fell into the same order of magnitude as Figure 4.2. The main goal of condensing information along the preprocessing step was to resize the images in order for them to be inputs for our networks. Both networks, either CNN based or transformer based, will have a small and a little instance.

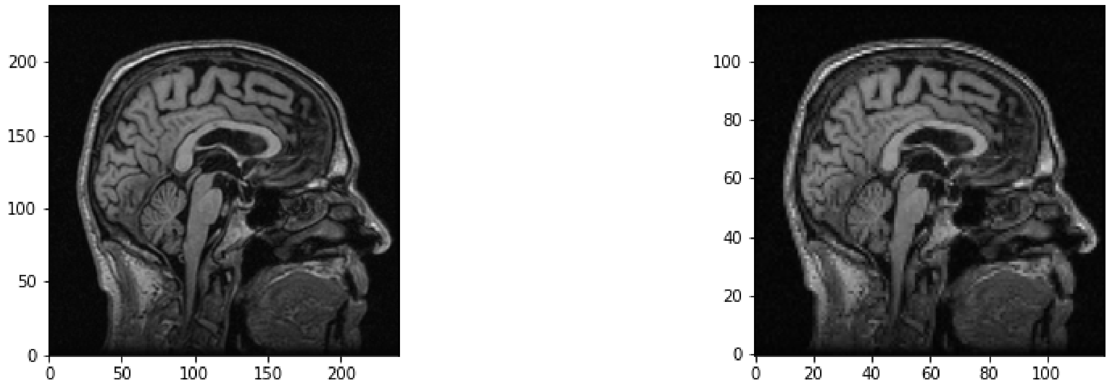
4.1.1. Spatial normalization

The goal of the spatial normalization preprocessing phase is to make the spatial structure of each image in the collection as similar as possible. The original ADNI scans had a wide range of shapes, and several sets from the database had been preprocessed earlier using various methods. Images had to be resampled to a standard isotropic resolution as a result.

Thus, in a first step, images were resampled to an isotropic resolution of 1 mm, meaning that every voxel would represent 1 mm^3 of space in the "real" world. Human brains differ in size

[2] `https://nifti.nimh.nih.gov/`

and shape, and one of the main goal of spatial normalization is to deform human brain scans so one location in one subject's brain scan corresponds to approximately the same location in another subject's brain scan.



(a) Voxel size = 1mm

(b) Voxel size = 2mm

Figure 4.3: Sagittal cut of a resampled MRI for different isotropic spacing
Grid sizes : (a) (240, 240, 216) and (b) (120, 120, 108)

On Figure 4.3 the patient have been resampled with two different isotropic sizes. In terms of content, one can see the the brains looks exactly the same. However, the 2 mm resampling has a resolution half as good as the 1 mm. Meaning that relevant information might have been lost. The upside of having a stronger resampling size resides in the size of the image. Indeed, Figure 4.3 (b) is twice smaller than Figure 4.3 (a). The resampling were done with the `sitk` tool from the `SimpleITK` library, an open-source multi-dimensional image analysis in Python. This data processing step was carried out in almost every work when dealing with neuroimages.

4.1.2. Skull-stripping

After spatial normalization, a skull-stripping step has been carried out to remove from the images anything outside the intracranial volume and keep only the relevant brain information to help us with the diagnosis. Skull stripping is the most important preprocessing step as it defines which voxels will be available for further processing.

Finding the right value for the primary hyper-parameter, the fractional intensity threshold, was crucial in this situation. In essence, this hyper-parameter assesses how aggressively the algorithm removes visual elements that do not correspond to brain tissue. A very small value would leave some irrelevant information in the image making the segmented brain larger, while a very big value would remove some brain tissue and make the segmented brain smaller.

Finding the perfect fractional intensity threshold for the whole dataset would not be possible because not all MRI data is acquired and preprocessed in the same manner before being put in the ADNI database. At the same time, keeping things straightforward was important to this endeavor. This threshold must lie between 0 and 1 and thus 10 values between that interval were considered. Each sample were then displayed as a sagittal view of the middle plane in order to visually assess the skull stripping performance. The threshold of 0.5 was chosen since it was able to maintain the right balance for the majority of MRI. On top of that, this value was advocated by the the `FSL BET` (Smith, 2002) community. Figure 4.4 show the performance of the skull stripping method on patient 24602.

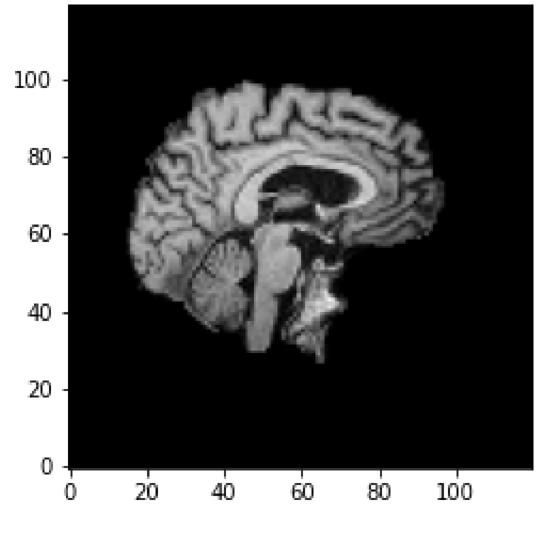


Figure 4.4: Sagittal cut of a skull-stripped image of a 2mm isotropic resampled MRI scan. Size : (120, 120, 108)

the `FSL BET` tools implemented into Python by the `Nipype` (Gorgolewski K, 2011) (Neuroimaging in Python Pipelines and Interfaces) library were used. Although FSL BET by itself works as a command line tool for UNIX systems, Nipype methods allow to execute it from a Python environment.

4.1.3. Cropping or padding

Finally, the last step is to make the size of the image 'digestible' for our neural networks. From Figure 4.4 one can still observe a big part of masked out image which is obviously irrelevant for the diagnosis of patient 24602. This part was carried out by sequentially performing two transformations: trimming then cropping or padding.

Trimming consists in finding the best fitting 3-dimensional bounding box that contains the whole brain while being the most compact. A trim function was implemented with two arguments : The original image and a mask to apply on it. Since the values of black pixels are null the mask was set to be different than 0.

From that point on, we will need to crop or pad the result into a fixed dimension size depending on the network. The big networks take as input images of sizes $(128, 128, 96)$ while the small networks will deal with $(96, 96, 48)$.

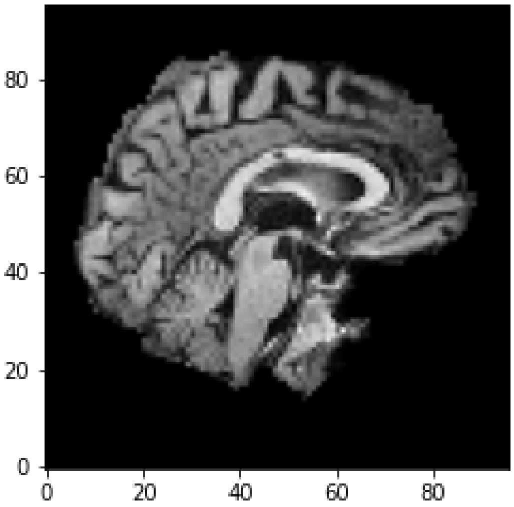


Figure 4.5: Sagittal cut of a cropped image from a 2mm isotropic resampled MRI scan. Final size : (96, 96, 48)

From Figure 4.5, one can see that the information is much more condensed comparing to Figure 4.2.

4.1.4. Intensity normalization

This final preprocessing phase was optional and its effect on performance will be studied and discussed during the results. The pixels in the neuroimages have a lot of different values ranging from 0 to 255. Several packages propose various methods to normalize the intensity of several modalities of magnetic resonance images whether it is T1-weighted, T2-weighted, or FLAIR. Among those individual normalization processes we find the fuzzy C-means (FCM) tissue-based normalization method [3].

FCM-based normalization uses fuzzy c-means to calculate a white matter (WM) mask of the image I. This WM mask is then used to normalize the entire image to the mean of the WM. The procedure is as follows : Let $W \subset B$ be the WM mask for the image I, i.e., W is the set of indices corresponding to the location of the WM in the image I. Then the WM mean is $\mu_{fcm} = \frac{1}{|W|}\sum_{\mathbf{w}\in\mathbf{W}} I(\mathbf{w})$ and the FCM-based normalized image is

$$I_{fcm}(\mathbf{x}) = \frac{c_1 \cdot I(\mathbf{x})}{\mu_{fcm}} \tag{4.1}$$

where $c_1 \in \mathcal{R}_{>0}$ is a constant that determines the WM mean after normalization. In this experiment, we use three-class fuzzy c-means to get a segmentation of the WM over the brain mask B for the T1-w image and we arbitrarily set $c_1 = 1$ (Reinhold et al., 2019).

4.2. Datasets

For the rest of this work, we will be dealing with three distinct datasets, each of them being differently preprocessed. In order not to be confused for the following we will assign names to each one.

- *HighRes Full* will correspond to a 1 mm isotropic resampled dataset of MRI scans containing the entire skull-stripped brain (size : $(128, 128, 96)$).
- *HighRes ROI* will correspond to a 1 mm isotropic resampled dataset of MRI scans cropped to a Region Of Interest focus on the Hippocampus (size : $(96, 96, 48)$).
- *LowRes* will correspond to a 2 mm isotropic resampled dataset of MRI scans containing the entire skull-striped brain (size : $(128, 128, 96)$).

[3] `https://github.com/jcreinhold/intensity-normalization`

In order to obtain the *HighRes ROI* dataset, the images were trimmed and then cropped, just like explained in subsection 4.1.3. The cropping resulted in a 3 dimensional centered cube of the brain images. On Figure 4.6, one can clearly notice the hippocampus (see section 2.4 for a comparison).

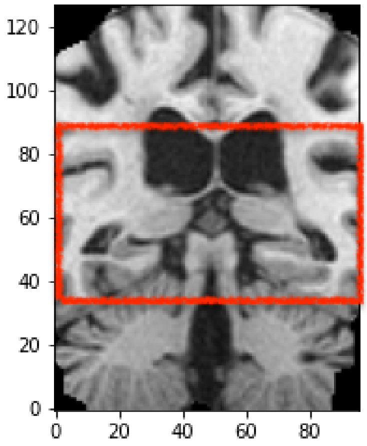


Figure 4.6: Coronal image of the Region of interest (red rectangle) focus on the Hippocampus

In every datasets, one can find 'AD, 'CN' and 'MCI' samples directly from the labels of ADNI. the 'MCI' images were then subdivided into two groups : progressive and stable like explained during section 4.1. This will enable a third classification task during experiments. There were 108 pMCI and 481 sMCI which means this classification will be the most imbalanced of the entire work.

4.3. Architectures and implementation details

At this point, the images are up and ready to be given as inputs into our networks. On the one hand, the big networks will be dealing with images of size $(128, 128, 96)$ from *HighRes Full*. On the other hand, the small networks will perform on inputs of size $(96, 96, 48)$. Those images can either be a from the *HighRes Roi* or the *LowRes* dataset.

Before going into the networks, all the input images will be standardized. Meaning constant of normalization such as the means and the standard deviation were computed on the training dataset and will be used to normalized each image. Indeed, computing the means and standard deviation across the entire dataset would have produce over-optimistic results. Let p be a random pixel from an image, μ_p be the mean value of this pixel along all the images in the training set and σ_p be its standard deviation. The standardized pixel $p*$ would have the value :

$$p* = \frac{p - \mu_p}{\sigma_p} \tag{4.2}$$

All the networks described here under were implemented using the `PyTorch` library (Paszke et al., 2017). The training procedures were carried out on the Alan GPU cluster at the University of Liège using several GPUs and CPUs.

4.3.1. CNN based models

This first network will take as input a full brain MRI scan which has been resampled to an isotropic size of 1 mm. To accelerate the training process and to avoid local minima, we used an ADAM optimizer (Kingma and Ba, 2014) to train.

Figure 4.7 displays the big Convolutionnal neural network dealing with MRI scans of shape $(128, 128, 96)$. Each block consists of a sequence of convolutional layers which are followed by a max-pooling layer. The same kernel size $(3 \times 3 \times 3)$ is applied over all convolutional layers. We also used a padding size of 1 to keep the size of the output after each convolutional layer. A max-pooling of size $(2 \times 2 \times 2)$ with strides of 2 is also applied to halve the resolution after each block.

BigCNN is composed of 8 convolutional layers, 5 max-pooling layers, and 4 fully connected layers. Therefore, the number of layers having tunable parameters is 12 (8 convolutional layers and 4 fully connected layers). The number of filters in the first block is 32, then this number is doubled in the later blocks until it reaches 512. This model is finished by four fully connected hidden layers and one output layer. The four fully connected layers go from 12288 to 10 gradually. The output layer consists of 1 neuron since this is a binary classification.

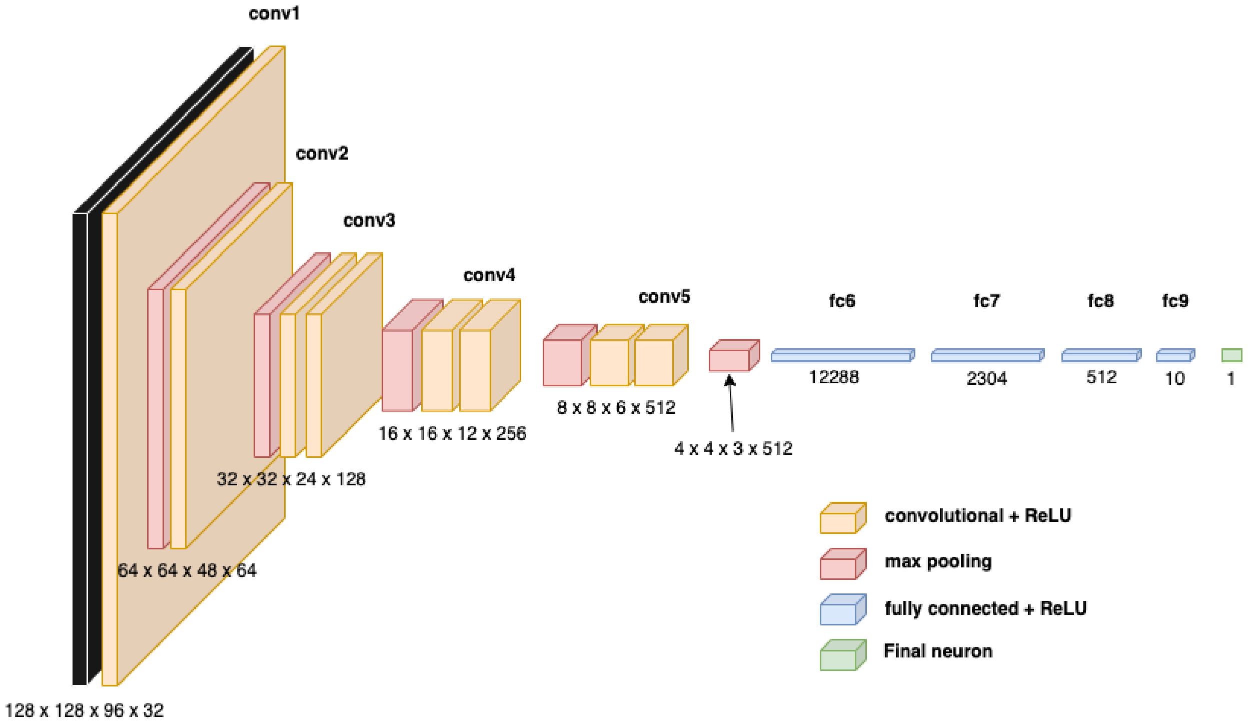


Figure 4.7: Convolutionnal Neural Network for a full brain scan

The last output neuron should represent the probability of the input being an MRI scan of an AD patient. However in this case the number in the last output neuron is not a probability lying between 0 and 1 but a logit. This decision was made in order to use the Binary Cross entropy with logit loss function from `torch` [4] which is a version more numerically stable than using a plain Sigmoid followed by a BCELoss. Since the operations are combined into one layer, the implementation take advantage of the log-sum-exp trick for numerical stability.

Figure 4.8 displays the smaller convolutional neural network. Another difference between them is the initial size of the output dimension of the convolution. In Figure 4.7 it is set at 32 while in Figure 4.8 it is set at 16.

[4] `https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html`

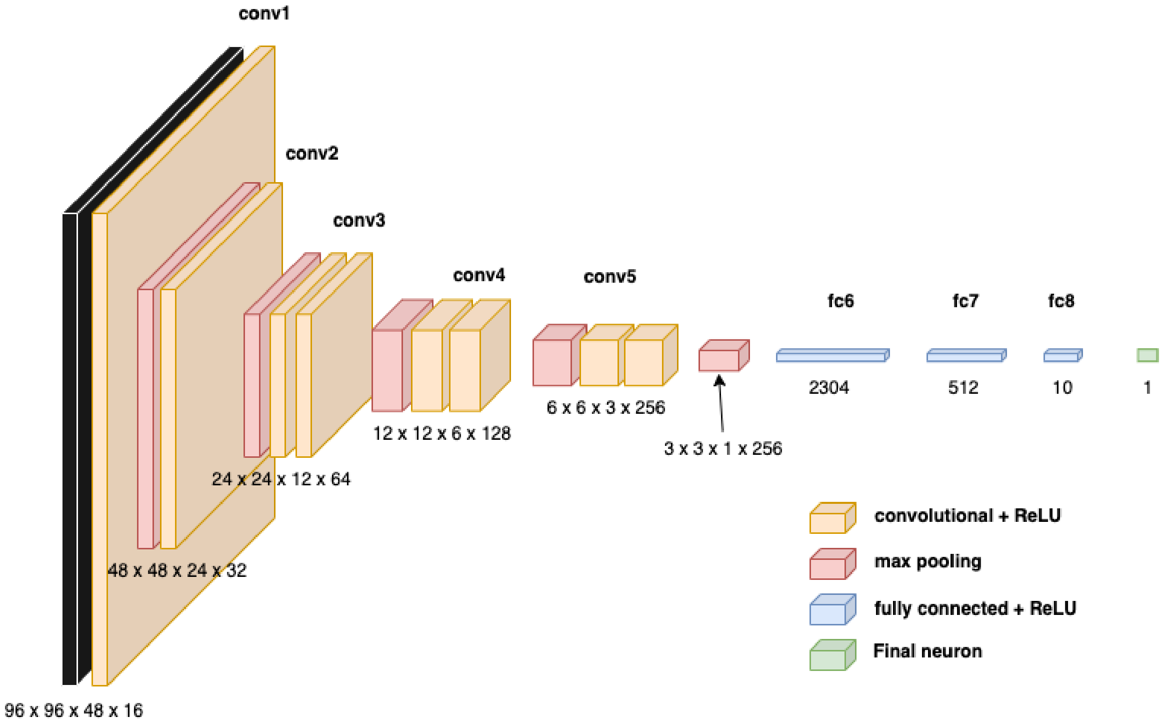


Figure 4.8: Small Convolutionnal Neural Network

4.3.2. Transformer based model

After the success of the Vision Transformer (ViT) in computer vision tasks (Dosovitskiy et al., 2020), such as demonstrating state of the art performance in image classification by large scale pre-training and fine tuning of a pure transformer, transformer based models have gained traction outside of the NLP framework.

The implementation we propose is inspired by the UNETR (Hatamizadeh et al., 2021), A vision transformer capable of dealing with 3 dimensional images such as MRI scans (see Figure 4.9). Their model was used for semantic segmentation of volumetric medical images. The transformer encoder was proposed to increase the model's capability for learning long-range dependencies and effectively capturing global contextual representation at multiple scales.

Just like in natural language processing, the transformer operate on 1 dimensional sequence of input embeddings. Similarly, we create a 1 dimensional sequence of a 3 dimensional input volume by dividing it into flattened uniform non-overlapping cubic patches. Subsequently, we use a linear layer to project the patches into a K dimensional embedding space, which remains constant throughout the transformer layers. In order to preserve the spatial information of the extracted patches, a 1 dimensional learning is added to the projected patch embedding.

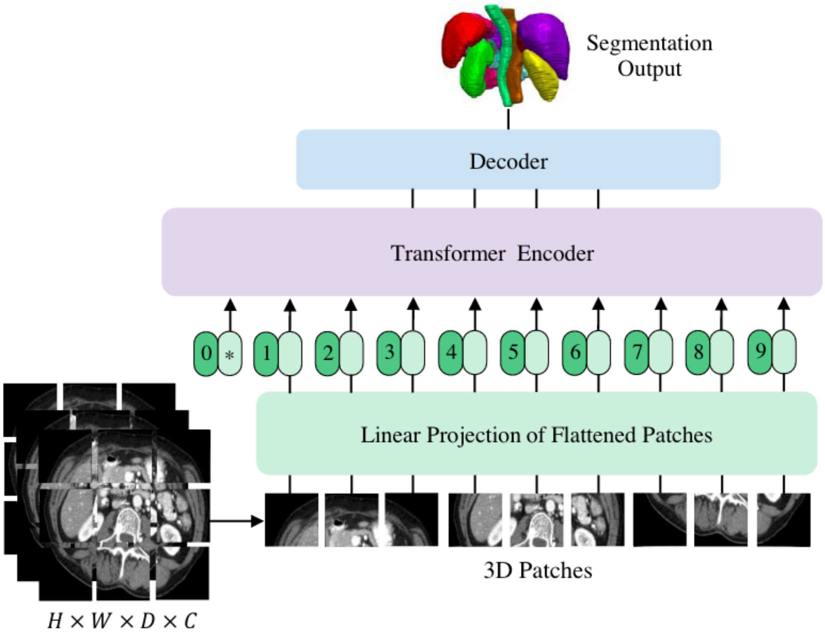

Segmentation Output
Decoder
Transformer Encoder
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches
3D Patches
$H \times W \times D \times C$

Figure 4.9: Architecture of the UNETR

In this instance, the 3D patches resolution is $(16, 16, 16)$ and the embedding size K is set to 768. The dimension of the feedforward layer is 3072 and there are a total of 12 attention heads. Only the input dimensions will change from the large ViT to the small ViT, the patch sizes will remain the same.

Chapter 5

Experiments

The generalization performance of a learning method relates to its prediction capability on independent unseen data. Assessment of this performance is extremely important in practise, since it guides the choice of learning method or model and gives us a qualitative measure of the ultimately chosen model. On top of that, a protocol must be carefully established in order to ensure the performances are relevant. In this chapter, protocol and evaluation metrics are discussed before reporting and discussing the final results.

5.1. Protocol

Any machine learning procedure will require tuning of hyperparameters. This tuning must be carefully undertaken in order not to "cheat" and therefore bias the results. The protocol for this work will follow the standard training - validation - test framework (Figure 5.1).

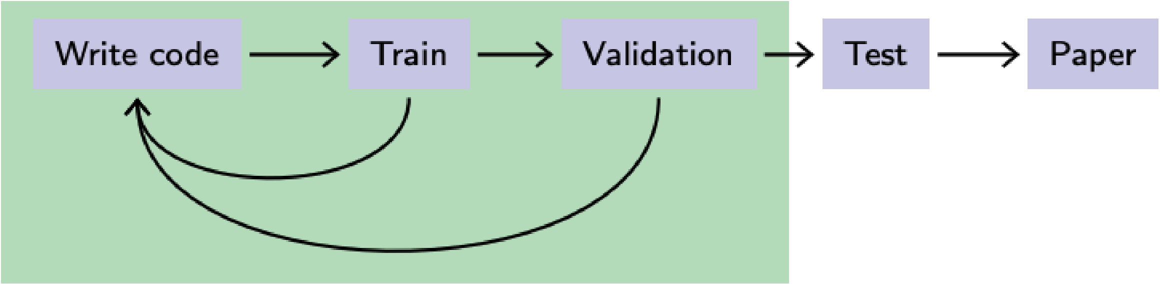


Figure 5.1: Classical machine learning protocol

The dataset for each classification will be separated into three parts : Training (70%), validation (15%) and Testing (15%). Across those three distinct datasets, each class will have the same proportion. Figure 2.4 informs us that there are roughly twice as much healthy patients (CN) than patient with Alzheimer's disease (AD). This means that each dataset will contain this ratio. To tackle the issue of unbalanced classification we add a weight w to the loss function, therefore penalizing more the error made on the minority class. Equation 3.1 from section 3 then becomes:

$$\mathcal{L} = w \cdot y \cdot \log f(\boldsymbol{x}) + (1 - y) \cdot \log(1 - f(\boldsymbol{x})) \tag{5.1}$$

The value of the weight w is initialized as the ratio between the two class. This means a prediction of 0 made by the model of a sample labeled 1 will be twice as penalized. This will be also enforce on the other classifications depending on the ratio between the majority and minority class.

To better visualize and capture the effects of hyperparameters, the training loss and the validation loss will be plotted side by side. When overfitting starts to occur, i.e. when the model becomes too familiar with training data and therefore is doing a worst job at generalizing, the model is saved. The latter will then perform on the testing set and the final evaluation metric will be reported.

5.2. Evaluation metrics

Let us denote P the total number of samples belonging to the positive class and N the total number of samples belonging to the negative class, TP the total number of true positives detected by the method and TN the total number of true negatives detected by the method. Results of a classification method can be summarized in a contingency table, or confusion matrix (see Table 5.1).

Table 5.1: Confusion Matrix

	Predicted Class		
Actual Class	**Positive**	**Negative**	Total
Positive	**T**rue **P**ositive	**F**alse **N**egative	P
Negative	**F**alse **P**ositive	**T**rue **N**egative	N

From those values one can derivate numerous ratios and metrics each evaluating in a different way a classification model performance. The first one will be the accuracy which is the rate of samples that have been correctly classified, i.e.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{P} + \text{N}}$$

While the accuracy might be the most intuitive evaluation metric anyone can think of, in a context of an unbalanced dataset it could be misleading. Consider a case where the model always predicts 0, the accuracy will be of the proportion of negative samples (in our case 66%). We may have to look for different metrics in that case. The precision is the proportion of positive samples correctly classified among all the samples that have been classified as positive. The recall (also called true positive rate or sensitivity) is the rate of positive samples that have been correctly classified while the specificity (or true negative rate) is the rate of negative samples that have been correctly classified.

$$Recall = \frac{TP}{P} = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{N} = \frac{TN}{TN + FP}$$

$$Precision = \frac{TP}{TP + FP}$$

Each of those metrics have their advantages and disadvantages. Determining our goals, in terms of which error metric to use, is a necessary first step because our error metric will guide all our future actions and decisions. Those metrics will all tell a different story about the performance of our model when taken individually, perhaps we need them to work in pairs.

5.2.1. Receiver Operating Characteristic

The receiver operating characteristic curve (ROC) is a commonly used summary for assessing the tradeoff between sensitivity and specificity. It is a plot of the sensitivity versus the false positive rate (1 - specificity) as we vary the parameters of a classification rule. In our case the parameter will be the classification treshold. The overall assessment of the model is then carried out by calculating the area under the curve (AUC).

Similarly to the accuracy, the ROC curve is usually more relevant when dealing with balanced dataset. Another curve which is mostly used in imbalanced framework and particularly in medical context classification, meaning the positive label refers to an illness or disease, is the precision-recall curve (PR-curve).

To evaluate our networks, we will use three different metrics specifically chosen to entirely grasp the performance as relevant as possible. The first one will be the accuracy and the second one will be the area under a precision recall curve. Given that the former is a straightforward ratio and the latter is a closer look at the positive predictions we would like to have some information about the negative prediction as well. The third metric will then be the specificity.

[1] Sofaer, H.R., Hoeting, J.A. Jarnevich, C.S. (2019)

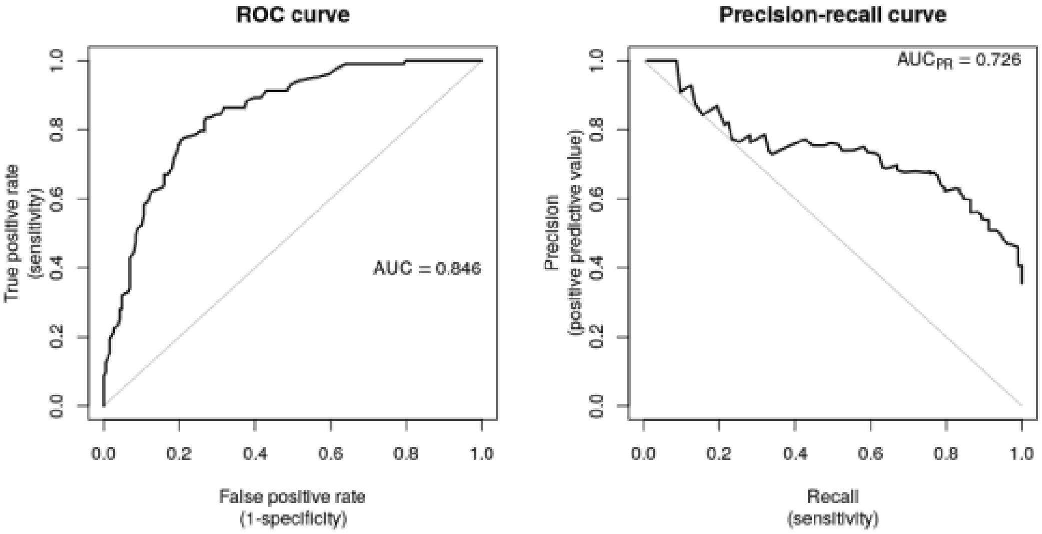


Figure 5.2: Comparison between the ROC and the Precision-Recall Curve [1]

5.2.2. Hyperparameters tuning

Every deep learning models come with several hyperparameters that control many aspects of the algorithm's behavior. Some of these hyperparameters affect the time and memory cost of running the algorithm. Some of these hyperparameters affect the quality of the model recovered by the training process and its ability to infer correct results when deployed on new inputs. Several tuning tools are available in deep learning libraries in order to visualize and quantify the effect of hyperparameters whether together or individually. In this work the tensorboard visualization kit implemented by tensorflow was used [2].

Each hyperparameter were chosen according to the framework described in section 5.1. Meaning their respective tuning were done in order to stabilize and optimize the performance in the training and validation process only. The first one is the `batch size`, i.e. the size of each batch of data going through the network during the training process before readjusting the weights with gradient descent. In this work, it was important to have large batches for two reasons. First, the training loss had a more stable convergence with a lower computing time and cost. Second, while larger batches might raise problems in generalizing to new data, in this case the generalization was actually improved in regards to smaller batches. A final key point concerning batch size is the fact that they must be as close as a whole divisor of the training set size for them to be equal along the entire training set. If not, this will cause problems by assigning a greater influence on the last smaller batch and possibly damage the learning process. Depending on the classifications the batch size were between 85 and 103.

[2] https://www.tensorflow.org/tensorboard/

The next hyperparameter is the `learning rate` γ. An improper learning rate, whether too high or too low, will result in a model with low effective capacity. Maintaining the same learning rate during the entire training process did not stabilize well. In order to take advantage of a relatively large γ in the start of the training and smaller ones at the end, we used scheduling to avoid sudden jumps if a local minima is reached. A scheduler is able to anneal the learning rate over time in different manners. In our case, the step decay method was used meaning γ was reduced by some constant factor every few steps. After considerations, the learning rate was initialized at 10^{-4} and is reduced by half every 100 steps. Note that the step in this case do not correspond the the epochs but the total number of batches that went into the network.

For the convolutionnal networks, an important hyperparameter is the kernel width. Increasing the kernel size increases the number of parameters in the model therefore its capacity. However a wider kernel causes a narrower output dimension, which will in contrast reduce model capacity unless we use implicit zero padding to reduce this effect. Wider kernel require more memory for parameter storage and increase runtime, but a narrower output reduces memory cost. As explained during subsection 4.3.1, the kernel is of size $(3 \times 3 \times 3)$ which is often a popular choice among practitioner.

5.3. Results and discussion

On Figure 5.4 is plotted the evolution of the loss value for the training and the validation set in respect to the steps. Indeed, the training loss is plotted after each batch while the validation loss after each epoch. We notice a nice convergence in regards to the training loss as well as for the validation. The accuracy and the specificity are expressed in % while the AUC_{pr} simply represents an area between 0 and 1[3]. During the discussions, we will refer 'AD' and 'pMCI' as the true samples given that they represent the worst case of the disease and they belong to the minority class. While 'CN', 'MCI' and 'sMCI' will be refered to as false samples.

As a baseline comparison we tested the 3D CNN model of Huang et al. (2019) on our unpreprocessed data [4]. This model was chosen because they also focused on a Hippocampus ROI with ADNI data, their model contained 3D CNN and finally they were one of the few to test pMCI vs sMCI.

Table 5.2: Overall Results on MRI scans from the *LowRes* dataset

	AD vs CN			AD vs MCI			pMCI vs sMCI		
	acc	AUC_{pr}	**spec**	**acc**	AUC_{pr}	**spec**	**acc**	AUC_{pr}	**spec**
Huang et al. (2019)	87.32	86.27	92.16	80.29	75.85	84.62	82.02	51.16	86.84

[3] will be multiplied by 100 for convenience

[4] Since they were dealing with ROI, the only difference was that they did not work with intensity normalized images (see section 4.1.4)

Table 5.3: Overall Results on MRI scans from the *HighRes* database

		AD vs CN			AD vs MCI		
		acc	AUC_{pr}	**spec**	**acc**	AUC_{pr}	**spec**
3D CNN	ROI	94.53	98.08	95.4	85.27	89.83	82.80
	Full Brain	87.86	92.94	90	85.29	87.52	88.50
Vision Transformer	ROI	79.69	68.28	85.71	71.87	60.24	89.41
	Full Brain	75.54	68.69	76.59	74.26	63.30	81.82

Baseline versus our work

Our preprocessing as well as our custom 3D CNN seem to have better results than the baseline in general in identifying AD using a region of interest. For the first classification (AD vs CN) our model scored 94.53 in accuracy compared to 87.32 for the baseline. On Figure 5.3 there is the evolution of the validation accuracy for this classification, not to be mistaken with the testing accuracy listed in the tables. It also important to notice that the AUC_{pr} is more than 10 points higher meaning our model is very good as distinguishing samples with dementia. The last metric also shows that healthy patients were well classified in both case with a slight advantage for our model (92.16 for them and 95.4 for us).

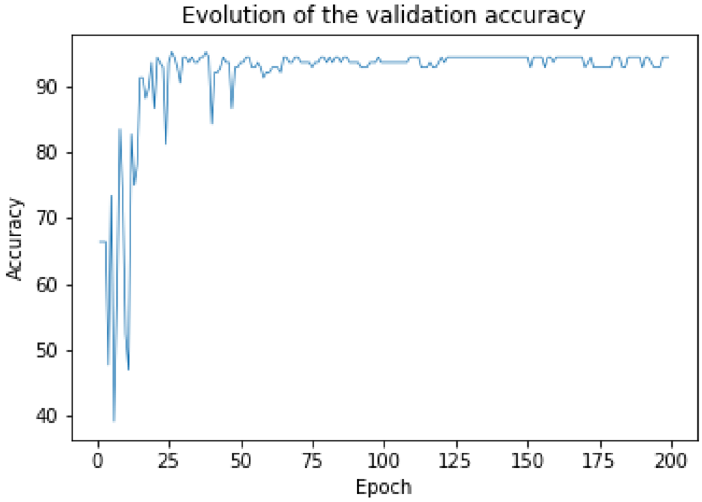


Figure 5.3: Evolution of the validation accuracy for the 3D CNN on the *HighRes* Roi database

Looking at the second classification (AD vs MCI) we also see stronger results both in accuracy and AUC_{pr} in our benefit, 85.27 versus 80.29 in accuracy and 89.83 versus 75.85 in AUC_{pr}. Their model achieved a higher specificity (84.62) than ours (82.8). This could come from the fact that they do not use a weight for positive samples (minority class) in their loss function which could explain why we have better prediction concerning true samples (with AUC_{pr}) but we have similar or sometimes worse predictions concerning false samples. The precision-recall curves for the first two classifications can be observed on Figure 5.5.

Table 5.4: Results for the third classification for the *HighRes* dataset

		pMCI vs sMCI		
		acc	AUC_{pr}	**spec**
3D CNN	ROI	85.39	70.80	97.06
	Full Brain	75	53.27	98.48
Vision Transformer	ROI	82.02	41.19	95.83
	Full Brain	87.34	47.25	95.31

Finally for the third classification (Table 5.4), we are glad to see that our CNN model performs better in accuracy (85.39 against 82.02) in AUC_{pr} (70.80 against 51.16) and in specificity (97.06 against 86.84). The rest of the results are not very good for one main reason : the AUC_{pr}. Indeed, one might believe that the performance of the vision transformer and the CNN on the *HighRes* ROI database are more or less similar (only a 3% difference between the accuracies and less than 2% difference between specificity) however the AUC_{pr} drops from 70.80 for CNNs to 41.19 for ViT. This means the ViT is doing a very bad job at identifying true samples. In fact, when given a true sample, the ViT will be more wrong then right. When taking a closer look at the test results we see that out of the 88 samples used to test the model, 17 were pMCI. Only 4 of them were correctly classified. This is the main reason why AUC_{pr} is a very important metric when dealing with relevant minority class. The ROC curve, which covers both classes, would have told a completely different story.

HighRes versus *LowRes*

On Table 5.5 are displayed the performance of the models on *LowRes* dataset. The effect on the metrics are clear. Indeed, resampling to a lower resolution implied loosing perhaps relevant information on the MRI scans. If we must compare them to Table 5.3, one must keep in mind that the *LowRes* dataset contains whole brain scan of dimensions $(96, 96, 48)$. We will then compare them to Full brain results of *HighRes*.

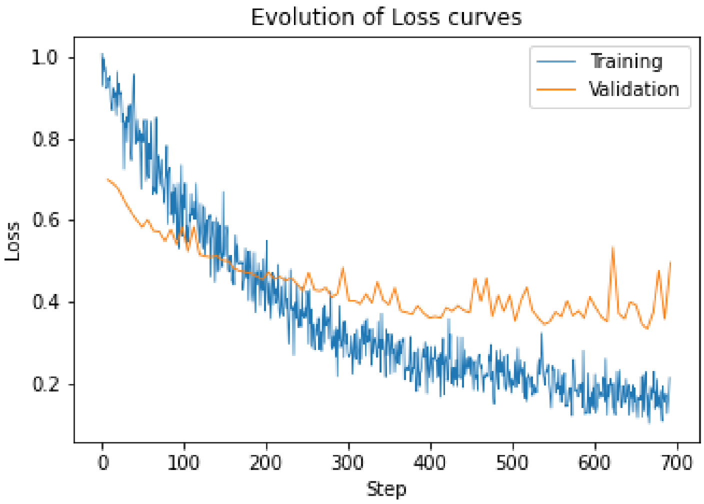


Figure 5.4: Evolution of the loss curve for the training (blue) and validation (orange) of the CNN on the *HighRes* Full brain dataset

The classification of AD vs MCI was really affected by the stronger resampling. Dropping from 87.52 to 61.38 in AUC_{pr} and 85.29 to 71.53 in accuracy. It is important to notice that the metrics concerning the AD vs CN classification are not far from each other. It could be interesting to also crop a region of interest focus on the hippocampus for this dataset. The data will be way smaller and therefore we might need less parameters and computing power to achieve similar results as Table 5.3.

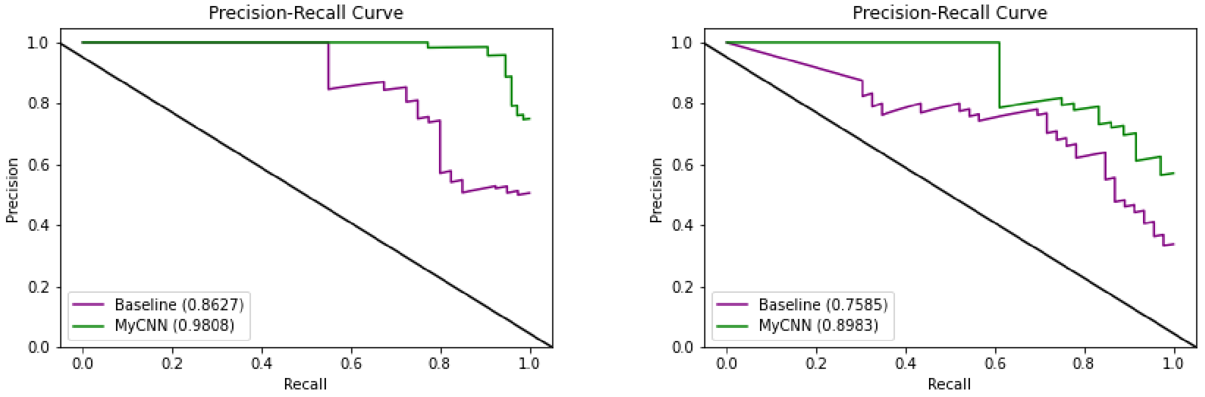
Table 5.5: Overall Results on MRI scans from the `LowRes` dataset

	AD vs CN			AD vs MCI			pMCI vs sMCI		
	acc	AUC_{pr}	**spec**	**acc**	AUC_{pr}	**spec**	**acc**	AUC_{pr}	**spec**
3D CNN	88.57	87.60	96.63	71.53	61.38	86.17	80.88	77.61	100
ViT	77.14	77.49	88.24	71.32	64.04	77.27	75	41.74	92

Roi versus Full brain

In two of of three classifications, the region of interest appears to have a more discriminative power than a full brain scan. This confirms that the hippocampus is in fact a brain region strongly affected by the disease. It is especially the case for the AD vs CN task where the full scan is automatically beaten by the ROI (In *HighRes*, 94.53 to 87.86 for CNNs, 79.69 to 75.54 for vision transformer).

However, it is interesting to see that for the AD vs MCI classification task the networks have the same kind of performance whether they are dealing with full brain scan or ROIs. This could be a consequence of the fact that the hippocampus is more vulnerable at the early stages



(a) AD vs CN

(b) AD vs MCI

Figure 5.5: Precision-recall curves of the baseline (purple) and our own CNN (green) along with their respective AUC

of AD, therefore it may not have a sufficient discriminative power between advanced stages of congnitive impairment.

CNN versus Vision transformer

Without any surprises, the CNN outperforms the ViT in every classification tasks. A very good result from ViT was on the *HighRes* ROI database reaching an accuracy of 79.69, a AUC_{pr} of 68.28 and a specificity of 85.71. Another fine result was on the *LowRes* dataset where the ViT had a 77.14 accuracy, a 77.49 AUC_{pr} (the highest for the ViT) and a 88.24 specificity (also its highest).

In the *LowRes* dataset, the difference between the two models in the AD vs MCI task is not big. The accuracies are less than 1% from each other, the ViT has a higher AUC_{pr} (64.04) then CNN (61.38) but the CNN has a higher specificity (86.17) than ViT (77.27).

During training, the vision transformer converged faster than the CNN. However, its generalization abilities were not as great.

A word on specificity for the third classification

The framework of the last classification operates with more than 4 times more false samples than true samples. As we discussed some models have a hard time predicting true samples but most of them can recognize false ones. This is why the specificity for the pMCI vs sMCI task is never lower than 95 (even reaching 100 in the *LowRes* dataset), but in this context it is not very relevant.

Chapter 6

Conclusion and future perspective

To finalize this work, we discuss the prospective and limitations of machine learning and medical diagnosis for the near future as well as the ethics of such work.

6.1. The future of AI assisted diagnosis

In natural language processing, researchers run and test their findings on an unified and agreed upon corpus called Workshop on Statistical Machine Translation (WMT). The latter is a machine translation dataset composed from a collection of various sources, including news commentaries and parliament proceedings. The corpus file has around 4M sentences. With the same evaluation metrics, i.e. BLEU scores or sacreBLEU scores, practitioners can easily assess the performance of their models or algorithms in fair comparison to other work. This drives research centers to pursue the perhaps new state-of-the-art result in a worldwide frame of reference. Whether it is in translation between english and german [1] or english and french [2] (see Figure 6.1)

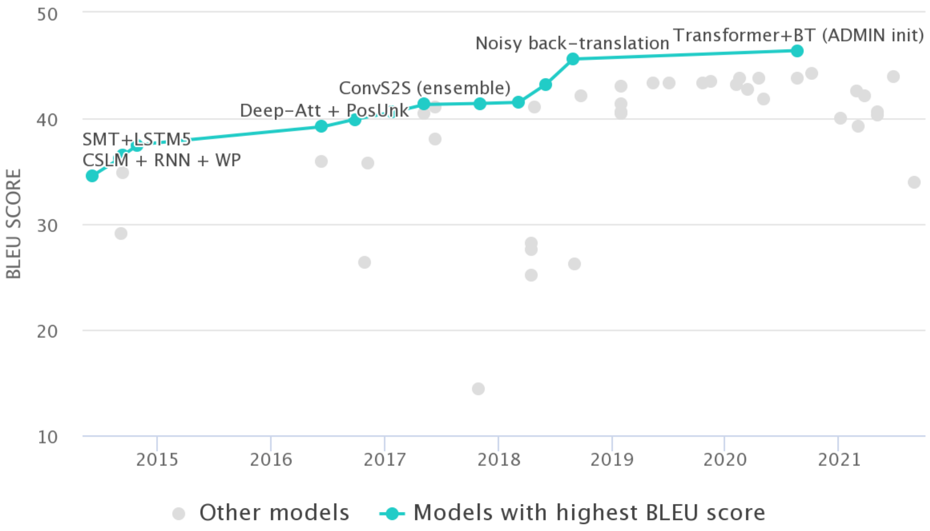


Figure 6.1: Machine Translation on WMT2014 English-French

[1] https://paperswithcode.com/sota/machine-translation-on-wmt2014-english-german
[2] https://paperswithcode.com/sota/machine-translation-on-wmt2014-english-french

Even if the NLP framework is inherently different from a medical binary classification framework, one can still argue that such unified frame of reference does not exist in medical contexts. There are plenty of databases available such as ADNI, AlzBiomarker [3] or OASIS [4] but none of them offers a huge uniform dataset of brain images.

Deep learning models operate best when given the maximum amount of data. While the ADNI database is one of the most significant in its field, the total amount of Alzheimer patients (see Figure 2.4) is astonishingly low. Especially considering the fact that an estimated 6.5 million Americans age 65 and older are currently living with the disease as of today (Alzheimers Dement, 2022). This number could grow to 13.8 million by 2060 barring the development of medical breakthroughs to prevent, slow or cure AD.

The distinction between stable and progressive MCI had already been made several times and have been the target of a lot of classification models. But not nearly as much as the more classical 'CN vs AD' or 'MCI vs AD' tasks. However as discussed in subsection 2.1 the importance in terms of medical relevance between them is clear. The main reason of this issue stems from the fact that the 'sMCI vs pMCI' dataset has to be handmade by the practitioner whereas dementia and healthy control are hardcoded labels in every neurodegenerative diseases database.

6.2. Ethics

Maybe someday, a model will be capable of predicting with impeccable accuracy the progression and possibly the potential state of patients in months to come way before medical staff can. With the right data and the right algorithm, this future might be closer than we think. Indeed, machine learning is rapidly expending and already taking over parts of our lives at a pace nobody can fully grasp. It is only a matter of time before diagnostic and prognostic are handed over to machines. While this prospect sounds utopian and safe, one must remember that the privacy of one's data have never been so important. The data flood in which we are currently drowning leverages the best machine assisted prediction while, at the same time, exposing information with a never seen before transparency. A world where an employer is capable of finding out which of his employees will develop dementia or any kind of illness is not a sane world.

[3] https://www.alzforum.org/alzbiomarker

[4] https://www.oasis-brains.org/

6.3. Conclusion

During this master thesis, we investigated the potential of some deep learning models to assess and predict Alzheimer's disease. With a different, more elaborated, preprocessing pipeline than most studies we were able to achieve strong classification results. Using a 3D convolutionnal neural network, we were able to reach 94.53% in identifying patients with Alzheimer's disease on a region of interest focused on the hippocampus and 85.39% in predicting patients who might develop dementia within 36 months. Both of those results are higher than the reference model. Experiments were also carried out on the entire brain image, mild cognitive impairment patients and on a lower resolution database with interesting outcomes. To the best of our knowledge this was the first time a Vision Transformer was applied to a MRI classification framework and while it cannot yet compete with the state if the art CNN architecture it shows some promising results.

Bibliography

Sitara Afzal, Muazzam Maqsood, Faria Nazir, Umair Khan, Farhan Aadil, Khalid M Awan, Irfan Mehmood, and Oh-Young Song. A data augmentation-based framework to handle class imbalance problem for alzheimer's stage detection. *IEEE Access*, 7:115528–115539, 2019. doi: 10.1109/ACCESS.2019.2932786.

Samsuddin Ahmed, Kyu Yeong Choi, Jang Jae Lee, Byeong C. Kim, Goo-Rak Kwon, Kun Ho Lee, and Ho Yub Jung. Ensembles of patch-based classifiers for diagnosis of alzheimer diseases. *IEEE Access*, 7:73373–73383, 2019. doi: 10.1109/ACCESS.2019.2920011.

Rufus Akinyemi, Michael Firbank, Godwin Ogbole, Louise Allan, Mayowa Owolabi, Joshua Akinyemi, Bolutife Yusuf, Oluremi Ogunseyinde, Adesola Ogunniyi, and Raj Kalaria. Medial temporal lobe atrophy, white matter hyperintensities and cognitive impairment among nigerian african stroke survivors. *BMC Research Notes*, 8, 10 2015. doi: 10.1186/s13104-015-1552-7.

Alzheimers Dement. 2022 alzheimer's disease facts and figures. *Alzheimers Dement*, 55:700–789, 4 2022. doi: 0.1002/alz.12638.

American Psychiatric Association. *Diagnostic and statistical manual of mental disorders (5th ed.)*, volume 55. 10 2013.

John Ashburner and Karl Friston. Voxel-based morphometry—the methods. *NeuroImage*, 11: 805–821, 07 2000. doi: 10.1006/nimg.2000.0582.

Abol Basher, Byeong C. Kim, Kun Ho Lee, and Ho Yub Jung. Volumetric feature-based alzheimer's disease diagnosis from smri data using a convolutional neural network and a deep neural network. *IEEE Access*, 9:29870–29882, 2021. doi: 10.1109/ACCESS.2021.3059658.

Christopher Beam, Cody Kaneshiro, Jung Jang, Chandra Reynolds, Nancy Pedersen, and Margaret Gatz. Differences between women and men in incidence rates of dementia and alzheimer's disease. *Journal of Alzheimer's Disease*, 64:1–7, 06 2018. doi: 10.3233/JAD-180141.

Y. Bengio and Yann Lecun. Convolutional networks for images, speech, and time-series. 11 1997.

Xiaojun Bi and Haibo Wang. Early alzheimer's disease diagnosis based on eeg spectral images using deep learning. *Neural Networks*, 114:119–135, 2019. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2019.02.005. URL `https://www.sciencedirect.com/science/article/pii/S0893608019300486`.

L Breiman. Random forests. *Machine Learning*, 45:5–32, 10 2001. doi: 10.1023/A:1010950718922.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL `https://arxiv.org/abs/2005.14165`.

Avinash Chandra, George Dervenoulas, and Marios Politis. Magnetic resonance imaging in alzheimer's disease and mild cognitive impairment. *Journal of Neurology*, 266, 06 2019. doi: 10.1007/s00415-018-9016-3.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. 10 2020.

Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning, 2016. URL `https://arxiv.org/abs/1603.07285`.

Giovanni Frisoni, Nick Fox, Clifford Jack, Ph Scheltens, and Paul Thompson. The clinical use of structural mri in alzheimer's disease. *Nature reviews. Neurology*, 6:67–77, 02 2010. doi: 10.1038/nrneurol.2009.215.

Chenjie Ge, Qixun Qu, Irene Yu-Hua Gu, and Asgeir Store Jakola. Multiscale deep convolutional networks for characterization and detection of alzheimer's disease using mr images. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 789–793, 2019. doi: 10.1109/ICIP.2019.8803731.

Madison C Clark D Halchenko YO Waskom ML Ghosh SS. Gorgolewski K, Burns CD. Nipype: a flexible, lightweight and extensible neuroimaging data processing framework in python. *Frontiers in neuroinformatics*, vol. 5, p. 13, 2011. doi: 10.3389/fninf.2011.00013.

Rita Guerreiro and Jose Bras. The age factor in alzheimer's disease. *Genome Medicine*, 7, 10 2015. doi: 10.1186/s13073-015-0232-5.

Yubraj Gupta, Kun Ho Lee, Kyu Yeong Choi, Jang Jae Lee, Byeong Chae Kim, Goo Rak Kwon, the National Research Center for Dementia, and Alzheimer's Disease Neuroimaging Initiative. Early diagnosis of alzheimer's disease using combined features from voxel-based morphometry and cortical, subcortical, and hippocampus regions of mri t1 brain images. *PLOS ONE*, 14(10):1–30, 10 2019. doi: 10.1371/journal.pone.0222446. URL `https://doi.org/10.1371/journal.pone.0222446`.

Ali Hatamizadeh, Dong Yang, Holger Roth, and Daguang Xu. Unetr: Transformers for 3d medical image segmentation. 03 2021.

Marti Hearst, S.T. Dumais, E. Osman, John Platt, and B. Scholkopf. Support vector machines. *Intelligent Systems and their Applications, IEEE*, 13:18 – 28, 08 1998. doi: 10.1109/5254.708428.

Yechong Huang, Jiahang Xu, Yuncheng Zhou, Tong Tong, Xiahai Zhuang, and the Alzheimer's Disease Neuroimaging Initiative (ADNI) . Diagnosis of alzheimer's disease via multi-modality 3d convolutional neural network. *Frontiers in Neuroscience*, 13, 2019. ISSN 1662-453X. doi: 10.3389/fnins.2019.00509. URL `https://www.frontiersin.org/articles/10.3389/fnins.2019.00509`.

Cosimo Ieracitano, Nadia Mammone, Alessia Bramanti, Amir Hussain, and Francesco Morabito. A convolutional neural network approach for classification of dementia stages based on 2d-spectral representation of eeg recordings. *Neurocomputing*, 323, 10 2018. doi: 10.1016/j.neucom.2018.09.071.

Rachna Jain, Nikita Jain, Akshay Aggarwal, and Jude D. Convolutional neural network based alzheimer's disease classification from magnetic resonance brain images. *Cognitive Systems Research*, 57, 10 2019. doi: 10.1016/j.cogsys.2018.12.015.

R.R. Janghel and Y.K. Rathore. Deep convolution neural network based system for early diagnosis of alzheimer's disease. *IRBM*, 42(4):258–267, 2021. ISSN 1959-0318. doi: https://doi.org/10.1016/j.irbm.2020.06.006. URL `https://www.sciencedirect.com/science/article/pii/S195903182030110X`.

Shankar K, Lakshmanaprabu S.K., Ashish Khanna, Sudeep Tanwar, Joel J.P.C. Rodrigues, and Nihar Ranjan Roy. Alzheimer detection using group grey wolf optimization based features with convolutional classifier. *Computers Electrical Engineering*, 77:230–243, 2019. ISSN 0045-7906. doi: https://doi.org/10.1016/j.compeleceng.2019.06.001. URL `https://www.sciencedirect.com/science/article/pii/S0045790618325448`.

Brendan J. Kelley and Ronald C. Petersen. Alzheimer's disease and mild cognitive impairment. *Neurologic Clinics*, 25(3):577–609, 2007. ISSN 0733-8619. doi: https://doi.org/10.1016/j.ncl.2007.03.008. URL `https://www.sciencedirect.com/science/article/pii/S0733861907000540`. Dementia.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

Charles J. Stone R.A. Olshen Leo Breiman, Jerome Friedman. *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen van der Laak, Bram Ginneken, and Clara Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42, 02 2017. doi: 10.1016/j.media.2017.07.005.

Chin-Fu Liu, Shreyas Padhy, Sandhya Ramachandran, Victor X. Wang, Andrew Efimov, Alonso Bernal, Linyuan Shi, Marc Vaillant, J. Tilak Ratnanather, Andreia V. Faria, Brian Caffo, Marilyn Albert, and Michael I. Miller. Using deep siamese neural networks for detection of brain asymmetries associated with alzheimer's disease and mild cognitive impairment. *Magnetic Resonance Imaging*, 64:190–199, 2019. ISSN 0730-725X. doi: https://doi.org/10.1016/j.mri.2019.07.003. URL `https://www.sciencedirect.com/science/article/pii/S0730725X19300086`. Artificial Intelligence in MRI.

Donghuan Lu, Karteek Popuri, Weiguang Ding, Rakesh Balachandar, and Mirza Faisal Beg. Multimodal and multiscale deep neural networks for the early diagnosis of alzheimer's disease using structural mr and fdg-pet images. *Scientific reports*, 8, 04 2018. doi: 10.1038/s41598-018-22871-z.

Atif Mehmood, Shuyuan Yang, Zhixi Feng, Min Wang, AL Smadi Ahmad, Rizwan Khan, Muazzam Maqsood, and Muhammad Yaqub. A transfer learning approach for early diagnosis of alzheimer's disease on mri images. *Neuroscience*, 460:43–52, 2021. ISSN 0306-4522. doi: https://doi.org/10.1016/j.neuroscience.2021.01.002. URL `https://www.sciencedirect.com/science/article/pii/S0306452221000075`.

Hina Nawaz, Muazzam Maqsood, Sitara Afzal, Farhan Aadil, Irfan Mehmood, and Seungmin Rho. A deep feature-based real-time system for alzheimer disease stage detection. *Multimedia Tools and Applications*, pages 1–19, 2020.

Sriramakrishnan Padmanaban, Kalaiselvi Thiruvenkadam, Padmapriya T., M. Thirumalaiselvi, and RAM KUMAR. A role of medical imaging techniques in human brain tumor treatment. 8:565–568, 01 2020. doi: 10.35940/ijrte.D1105.1284S219.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Ronald Petersen, Glenn Smith, Steve Waring, Robert Ivnik, Eric Tangalos, and Emre Kokmen. Mild cognitive impairment: Clinical characterization and outcome. *Archives of neurology*, 56:303–8, 04 1999. doi: 10.1001/archneur.56.6.760.

Jacob C Reinhold, Blake E Dewey, Aaron Carass, and Jerry L Prince. Evaluating the impact of intensity normalization on MR image synthesis. In *Medical Imaging 2019: Image Processing*, volume 10949, page 109493H. International Society for Optics and Photonics, 2019.

Saman Sarraf, Danielle DeSouza, John Anderson, and Ghassem Tofighi. Deepad: Alzheimer's disease classification via deep convolutional neural networks using mri and fmri. *bioRxiv*, 08 2016. doi: 10.1101/070441.

Jun Shi, Xiao Zheng, Yan Li, Qi Zhang, and Shihui Ying. Multimodal neuroimaging feature learning with multimodal stacked deep polynomial networks for diagnosis of alzheimer's disease. *IEEE Journal of Biomedical and Health Informatics*, 22(1):173–183, 2018. doi: 10.1109/JBHI.2017.2655720.

Dan Silverman, Gary Small, Yinhla Carol, Carolyn Lu, Michelle Aburto, Wei Chen, Johannes Czernin, Stanley Rapoport, Pietro Pietrini, Gene Alexander, Mark Schapiro, William Jagust, John Hoffman, Kathleen Welsh-Bohmer, Abass Alavi, Christopher Clark, Eric Salmon, Mony Leon, Renee Mielke, and Michael Phelps. Positron emission tomography in evaluation of dementia regional brain metabolism and long-term outcome. *JAMA*, 286:2120–2127, 11 2001. doi: 10.1001/jama.286.17.2120.

Stephen Smith. Fast robust automated brain extraction. *Human brain mapping*, 17:143–55, 11 2002. doi: 10.1002/hbm.10062.

Robert Tibshirani. Regression shrinkage selection via the lasso. *Journal of the Royal Statistical Society Series B*, 73:273–282, 06 2011. doi: 10.2307/41262671.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. URL `https://arxiv.org/abs/1706.03762`.

Hongfei Wang, Yanyan Shen, Shuqiang Wang, Tengfei Xiao, Liming Deng, Xiangyu Wang, and Xinyan Zhao. Ensemble of 3d densely connected convolutional network for diagnosis of mild cognitive impairment and alzheimer's disease. *Neurocomputing*, 333:145–156, 2019. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2018.12.018. URL `https://www.sciencedirect.com/science/article/pii/S0925231218314723`.

Marie Wehenkel, Antonio Sutera, Christine Bastin, Pierre Geurts, and Christophe Phillips. Random forests based group importance scores and their statistical interpretation: Application for alzheimer's disease. *Frontiers in Neuroscience*, 12, 2018. ISSN 1662-453X. doi: 10.3389/fnins.2018.00411. URL `https://www.frontiersin.org/articles/10.3389/fnins.2018.00411`.

Tao Yin, Peihong Ma, Zilei Tian, Kunnan Xie, Zhaoxuan He, Ruirui Sun, and Fang Zeng. Machine learning in neuroimaging: A new approach to understand acupuncture for neuroplasticity. *Neural Plasticity*, 2020:1–14, 08 2020. doi: 10.1155/2020/8871712.

Y. T. Zhou and Rama Chellappa. Computation of optical flow using a neural network. *IEEE 1988 International Conference on Neural Networks*, pages 71–78 vol.2, 1988.

Appendix A

Your appendix