## Deep Learning

Profs. Mauro Castelli & Yuriy Perezhohin

# Project Report

## Group 8 Students:


Miguel Cardoso, 20201541
Pedro Coimbras, 20211573
Pedro Bonifácio, 20211652
Petr Terletskiy, 20211580
Tiago Monteiro, 20211641

NOVA IMS Information Management School

# Introduction

Alzheimer's disease, characterized by progressive neurodegeneration, presents a substantial global public health challenge. Given the increasing prevalence of this disorder, the importance of early and precise diagnosis becomes crucial for effective intervention and patient care.

The goal of this project was to design and implement a deep-learning model capable of accurately classifying Alzheimer's progression types, with an emphasis on achieving robust performance on unseen data. The project utilizes a dataset sourced from the *Open Access Series of Imaging Studies*, encompassing a diverse range of images, each corresponding to a distinct stage of Alzheimer's progression.

# Data Preprocessing

## Metadata

The dataset is initially composed of only two columns. Upon closer examination of the 'Image ID' column, important information was identified and, thus, partitioned into four additional columns. These columns contained information regarding '**Patient ID**' ('OAS1_**0001**_MR1_mpr-1_100.jpg'), '**MR Category**' ('OAS1_0001_**MR1**_mpr-1_100.jpg'), '**MPR Type**' ('OAS1_0001_MR1_**mpr-1**_100.jpg') and '**Slice Number**' ('OAS1_0001_MR1_mpr-1_**100**.jpg').

The dataset contains **347** patients, with a total of **84337** images. Out of these nearly eighty five thousand images, **79.7%** are of "*Non Demented"*, **16.3%** of "*Very Mild Dementia"*, **3.6%** of "*Mild Dementia"* and **0.5%** of "*Moderate Dementia"* which shows our dataset is severely imbalanced.

After checking for inconsistencies in the data, nine missing values were found in the *"Label"* column. These missing labels in the images were imputed by the patient's label mode, i.e, with the label of the other images of that patient.

The data was then split into train and test sets and duplicates were checked. Afterwards, the number of images per patient were examined and we concluded that the majority of patients have around 244 MRI images, suggesting a commonality in the number of scans performed for a significant portion of the dataset. However, there is variability in the counts, with some patients having different numbers of MRI images, such as 122, 146, 131, 129, 118, etc.

## Image Preprocessing

**N4 Bias Field Correction** algorithm is a popular method for correcting low frequency intensity non-uniformly, also known as bias, present in Magnetic Resonance Imaging data. N4 Bias helps to correct these bias fields in the MRI images, making them more suitable for further analysis.

**Intensity Normalization** is a process in image processing that changes the range of pixel intensity values. This technique is often used in applications where the contrast in an image is poor due to factors such as glare. Due to the nature of MRI, even images of the same patient on the same scanner at different times can have different intensities. Many MRI segmentation models use an intensity normalization from Nyul et al. (2000) to alleviate this problem. Additionally, as is typical with Convolutional Neural Networks, each input channel (i.e. sequence) is normalized to have zero mean and unit variance within the training set.

**Skull Stripping** or brain extraction, is a process that involves the removal of non-brain tissue signals from magnetic resonance imaging (MRI) data. The package we used - *deepbrain* - employs a CNN model which generates an image mask where the brain tissue is located. Afterwards, we just need to filter the original image using the generated mask. We attached the package's scripts and models under directory "*tools' ' because* we had to port it to *tensorflow* >2.x (the package was built for *tensorflow 1.x*).

In the final preprocessing algorithm, we only used Intensity Normalization and Skull-Stripping. The time taken by the algorithm that used all three techniques took 20 hours, so we decided to remove the N4 Bias Field Correction due to its significant time consumption. Excluding the N4 Bias Field Correction the preprocessing model runtime reduced from 20 hours to just 4 hours.

### MRI Selection Algorithm

The primary objective of this algorithm is to identify the MRI slice where we see the top of the patient's eyes. Images starting from this region and going into the lower regions of the brain provide optimal visibility of the temporal lobe and hypothalamus (fig. 1). These two regions are of paramount importance when identifying Alzheimer's disease using axial planes of MRI scans, due to being the earliest regions to display atrophy [1] [2] [3].

We developed this algorithm because we plan to use convolutional 3D layers in one of our models. This type of model is widely used in Medical Imaging problems, but we encountered a roadblock in our data: each patient has a variable number of MRI slices, a variable number of MPR Types (MRI reconstructions), and slice numbers that don't correspond to the same brain areas between patients. Convolutional 3D layers expect the same number of slices for each patient, so we had to find a way to select the same slice for each patient and while doing so, we also wanted to select the slice that would provide the best visibility of the temporal lobe. This algorithm was our solution to this problem. After finding said slice for each patient, we extracted that slice plus 19 more slices below that one, resulting in a 20-slice MRI for each patient. This way, we give the same structure to our data, so we can use 3D convolutional layers in our model.

Here is a brief explanation of the algorithm's steps:

1. Iteration by each patient;
2. Iteration by each image of said patient;
3. Applies a custom filter to the image, which will replace pixels with values lower than 155 by a 0 (black), and the remaining pixels by a 1 (white);
4. Finds the rightmost pixel with value equal to 1, this is often the nose area;
5. Crops the image to the right eye's location by creating a window considering the location of the rightmost pixel;
6. Calculates the whiteness of the cropped image by summing the binary array's values. Records the whiteness value in a list;
7. After calculating whiteness for each image, iterates over the resulting list from the last element to the first, basically, from the topmost slice to the bottommost slice;
8. Selects the first image for which the whiteness value is higher than 300 or amounts to 80% of the maximum whiteness. The Slice Number of the selected image is recorded, as well as the respective MPR Type used.

## Implemented Models

### Convolutional Neural Network

A Convolutional Neural Network (CNN) is constructed with multiple layers, each containing numerous neurons. Fundamental components of a CNN include convolution layers, detector layers, and pooling layers. In a convolution layer, convolutional kernels are employed to extract low-dimensional features from the input data while maintaining the spatial relationship between the pixels of the input data.

### Base Model - 2D CNN

Before proceeding to more complex models, we will start by building a simple model to serve as a baseline. This model will be a simple 2D CNN with a few convolutional layers and a couple of dense layers.

This model will train on each raw image individually, with complete disregard for information about the patient. After performing the train test split, we trained the model and discovered that it reached an accuracy nearing 100%. The model was suffering from data leakage because the train test split was done randomly, without considering the Patient ID. This means that the model was learning from the same patient in both train and validation sets, which is not a good practice. This problem is further aggravated by the fact that each patient contributes too many MRI images with little variation between them, which means that the model easily learns to identify the patients instead of the disease. The solution to this problem is doing the train test split by Patient ID, stratifying by the target variable.

After implementing these changes to the train test split, the model was nearing 75% accuracy on validation data, but it was clearly betting on the majority class and easily overfitted to the training data, even after employing rather drastic measures against it. This score will serve as a baseline comparison to our other, more complex solutions. We also evaluated the model on the test set, although we know that the results will be biased because of the data leakage between the train and test sets (83% accuracy and 80% weighted f1-score). We will employ the same split strategy in the subsequent models.

### Final Model - 3D CNN

The input data for our 3D CNN model was obtained by extracting sequences of 20 slices per patient. To generate the sequence, the selected scan is retrieved from the results of the MRI Selection Algorithm. Then, the sequence is created by appending the 19 slices that anticipate the selected one. This effectively creates a 3D model of the lower part of the patient's brain, which contains the temporal lobe [4].

With the data in a suitable format, as described above, we trained two models: one with image preprocessing (intensity normalization and skull stripping), and the other without. This allowed us to quantify the effect of preprocessing on our model's performance. These models have an identical architecture, with three 3D Convolutional Layers of 32, 64 and 128 filters, respectively. In between each of these layers we have one Max Pooling Layer of size (2, 2, 2), each followed by a Dropout Layer. After flattening, a Dense Layer with 128 Neurons, using "Relu" as its activation function is followed by another Dropout Layer. The final layer of our model has 4 neurons, one for each class, and "Softmax" as its activation function for outputting the probability distribution. We compiled our model using the optimizer Adam (learning rate of 0.0001) and given our one-hot encoded labels, used categorical cross entropy as the loss function. The metrics used were Recall and Precision. These metrics were chosen due to the nature of the problem and the dataset.

## Results

After training and testing both models, we arrived at the following conclusions:

| Metric | Test. Precision | Test. Recall | Test f1-score |
|---|---|---|---|
| Macro-Averaged | 0.585 | 0.604 | 0.591 |
| Weighted-Averaged | 0.774 | 0.785 | 0.773 |

Due to hardware limitations, we had to limit the batch size to 3, which severely hindered the model's performance and compelled it to overfit. Even so, the chosen evaluation metrics show that we are on the right track, with a fairly high recall, which we deemed the most important metric for this problem. We chose this metric because in the context of dementia classification, false negatives (failing to identify actual cases of dementia) can have severe consequences, as early and accurate detection is crucial for timely intervention and patient care. We should also take into account that we are training the model using only 20 slices per patient, so we consider these good results for the amount of data fed.

## Areas for Improvement

When trying to improve the final model, we encountered two obstacles:

1. Hardware Limitations: if we trained the model on a GPU with higher memory capacity, it certainly would allow us to fine-tune it more freely and train the model with more data (e.g. implementing oversampling on minority classes);
2. Test set: due to the nature of the test set, which contains different slices from the same patients that the model trains on, we couldn't get an accurate quantification of the model's performance on unseen data. The structure of the 3D tensor fed to the model is also compromised due to the test set often not having consecutive slices of a patient's MRI. One solution to this would be to gather all the available data and create our own train and test datasets, but we decided against it, as to preserve the project's structure.

## Conclusion

In conclusion, our approach to this problem was tailored for a real-life use case where the model is used on unseen patients. The processing techniques and tools applied to our dataset and our progression through several CNN models, from an initial 2D CNN to the finished 3D CNN, illustrates how our methodology was tailored in this way. The models showed potential despite early difficulties including data leaking and major hardware constraints affecting batch size and training data size. The meticulous examination of preprocessing methods, such as intensity normalization, and skull-stripping was a key part of our methodology and played a crucial role in shaping our methodology and refining the model's performance.

However, hardware limitations restrained our ability to effectively fine-tune our final model such as by limiting the batch size to no more than 3. Given the preprocessing work done we believe that more competent hardware would allow us to create an effective model for real use case scenarios on unseen patient data.

## Python Libraries

| Python Library | Description and use |
|---|---|
| Pathlib, Pickle & OS | Used to simplify file and directory path handling, file manipulation, and environment variables management |
| Pandas & NumPy | Used for data manipulation |
| Matplotlib & Seaborn | Used for data visualization |
| OpenCV (cv2) | Used for image preprocessing enabling tasks such as object detection, and image enhancement |
| SimpleITK | Used to simplify the complexities of image analysis |
| tqdm | Used to show the progress of operations |
| tools.extractor | Used to build the Skull-Stripping algorithm |
| concurrent.futures | Used to optimize and simplify the execution of tasks, enabling efficient utilization of multi-core processors. |
| Sklearn | Used for train-test split and model evaluation |
| Keras & Tensorflow | Used for building and training our deep learning model |

## References


- [1] https://www.jneurosci.org/content/42/10/2131
- [2] https://www.medicalnewstoday.com/articles/alzheimers-brain-vs-normal-brain#alzheimers-brain
- [3] https://dementia.ie/lessons/mri-scan-of-brain-alzheimers-disease/
- [4] https://www.frontiersin.org/articles/10.3389/fbioe.2020.534592/full


## Annex

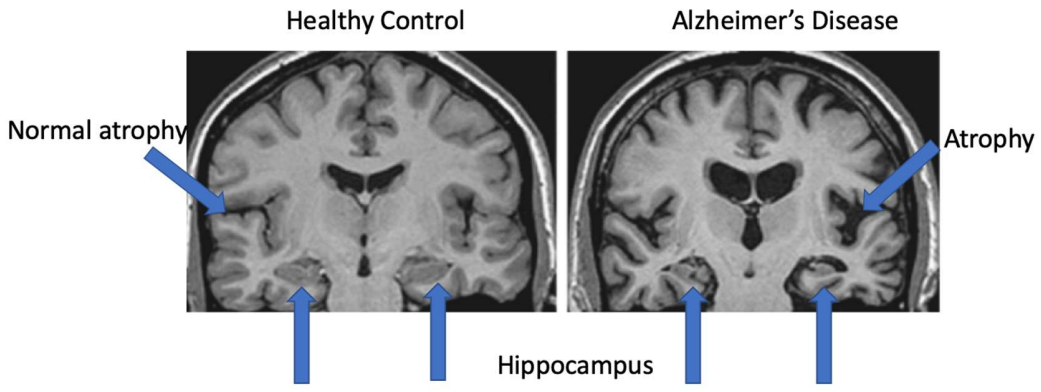


fig. 1 - a healthy brain (on the left) VS a brain with Alzheimer's disease (on the right)