PREDICTING BRAIN CONNECTIVITY MAPPING USING RADIOMICS FEATURES IN ANATOMICAL MRI

LEVENTE ZSOLT NAGY

Thesis supervisor
ALFREDO VELLIDO ALCACENA (Department of Computer Science)

Thesis co-supervisor
ESTELA CAMARA MANCHA (Hospital Universitari de Bellvitge)

Degree
Master's Degree in Artificial Intelligence

Master's thesis

School of Engineering
Universitat Rovira i Virgili (URV)

Faculty of Mathematics
Universitat de Barcelona (UB)

Barcelona School of Informatics (FIB)
Universitat Politècnica de Catalunya (UPC) - BarcelonaTech

Abstract

This study explores alternatives to diffusion MRI for mapping brain connectivity, predicting fractional anisotropy and mean diffusivity, using radiomic features derived from T1 and T2 structural MRI images. This approach aims to significantly enhance the cost and time efficiency of data acquisition, eliminating the need for diffusion MRI and tractography. The research is centered on the basal ganglia, a region primarily affected by neurodegeneration in Huntington's disease, comparing its characteristics between control subjects and patients with the condition.

Contents

List of Notations & Abbreviations

MRI magnetic resonance imaging 7

dMRI diffusion magnetic resonance imaging 7

FA fractional anisotropy 7

MD mean diffusivity 7

RD radial diffusivity 7

ROI region of interest 7

NN neural network 14

FNN feedforward neural network 22

CNN convolutional neural network 14

FCNN fully convolutional neural network 13

NIfTI neuroimaging informatics technology initiative 10

FMRIB functional magnetic resonance imaging of the brain

FSL FMRIB software library 43

FNIRT FMRIB's nonlinear image registration tool 7

GLCM gray level co-occurrence matrix 15

GLSZM gray level size zone matrix 15

GLRLM gray level run length matrix 15

NGTDM neighbouring gray tone difference matrix 15

GLDM gray level dependence matrix 15

cUHDRS composite Unified Huntington's Disease Rating Scale 18

CAP CAG Age Product 18

UML Unified Modeling Language 43

List of Figures

List of Tables

Introduction

Basal ganglia is a part of the human brain which is a group of subcortical nuclei responsible primarily for motor control, as well as other roles such as motor learning, executive functions and behaviors, and emotions. [1] Huntington's disease is a disorder that causes the progressive degeneration of the basal nuclei. [2]

Hospital de Bellvitge provided an excellent dataset of magnetic resonance imaging (MRI) and diffusion magnetic resonance imaging (dMRI) records of 32 control and 38 Huntington patient records of T1 and T1/T2 MRI images with isotropic voxels of 1 millimeter resolution and dMRI fractional anisotropy (FA), mean diffusivity (MD) and radial diffusivity (RD) images with isotropic voxels of 2 millimeter resolution. Furthermore this dataset also contains the mask for the basal ganglia, which will also be referenced as the region of interest (ROI). Masks for the 7 main cortical regions of the brain, which will also be referenced as the target regions: Limbic, Executive, Rostral-Motor, Caudal-Motor, Parietal, Occipital and Temporal are also included in the dataset. Tractography was performed on the dMRI images to figure out which parts of the ROI are connected to which cortical target, in a similar manner to how it was done in this paper [3]; where the relative connectivity maps are representing the ratio of the number of streamlines to each cortical target. Furthermore, the raw streamline images are also available, where there are a maximum of 5000 streamlines from each voxel in the ROI. The subcortical segmentation of the Basal Ganglia is also available, for the Caudate, Putamen and Accumbens on the control records. And lastly FMRIB's nonlinear image registration tool (FNIRT) warp fields were also provided for converting the records into normalized space.

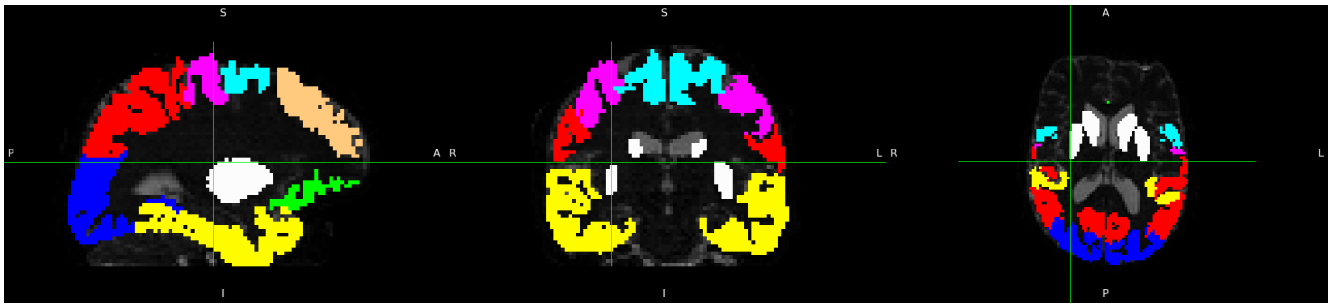

Figure 1.1: Basal Ganglia (ROI) & Cortical Targets

Color	Region
White	Basal Ganglia (ROI)
Green	Limbic
Brown	Executive
Light Blue	Rostral-Motor
Purple	Caudal-Motor
Red	Parietal
Blue	Occipital
Yellow	Temporal

Table 1.1: Regions Legend

Furthermore, for both the ROI and cortical targets, the dataset distinguishes between the right and left hemispheres of the brain. Thus there are actually 2 ROIs and $2 \cdot 7 = 14$ target regions.

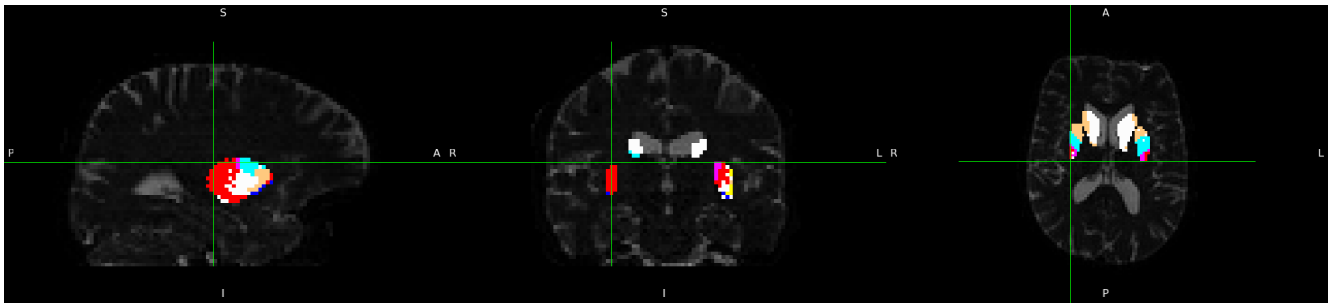

Figure 1.2: Connectivity Maps

1.1 Objectives

The end goal is to predict the relative connectivity of the Basal Ganglia to the cortical targets, from the radiomics features of the T1 and T1/T2 images.

This being a very complex problem, there is the possibility that the correlation between the connectivity of the brain and the T1, T1/T2 images are too weak to be mapped on this dataset. As from a datascience perspective, 70 records are not much. But from a medical perspective it is substantial as it is very hard to collect uniform, clean data, with permissions to use it for research.

A simpler task leading up to the complex end goal, is a model for the simple segmentation of the Basal Ganglia for the subcortical regions Caudate, Putamen and Accumbens. In order to confirm that the radiomics texture of the T1 and T1/T2 images of this dataset are correlated to the segmentation of the Basal Ganglia. This problem is inherently connected to the main goal, as the relative connectivity does obey certain anatomical restrictions, and the subcortical segmentation of the Basal Ganglia is confirmed to be related to the relative connectivity. Thus if this simpler prediction fails, there is a good chance that the complex end goal will fail as well.

Another intermediate task, is a model for predicting FA and MD images. This is also related to the main goal, as these images are computed from the dMRI images, the same image that the relative connectivity is computed from. But it is inherently simpler, not needing to perform complex algorithms like tractography.

The biggest obstacle of this project is the preprocessing of the data, as there are many variations and hyperparameters that can be tuned. An exhaustive search definitely will not be viable, thus

the preprocessing and model will needed to be tuned in a waterfall like manner, making educated guesses and comparing model performances across different tries. The main metric to measure model performance, will be the accuracy of the label prediction across voxels, as it should be comparable between all approaches. And pearson correlation will be used as the metric to evaluate the FA and MD regression predictions.

1.2 Motivation

The motivation for predicting the connectivity maps from the T1 and T1/T2 MRI images, is skipping the time and resource consuming process performing dMRI and tractorgrapy.

1.3 State of the Art

Design

In order to understand some of the following design choices, it makes sense to establish it early that the model will be operating on extracted voxel based features and non-voxel based features, and will predict on a voxel by voxel level.

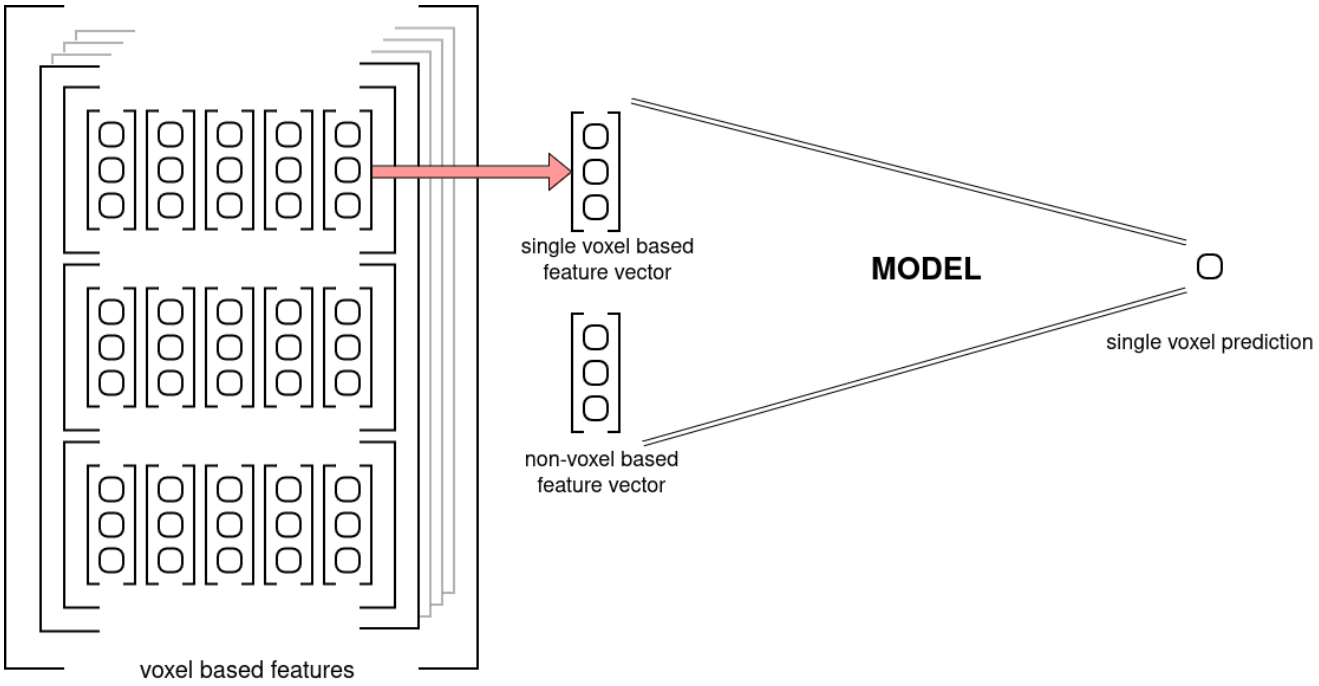

Figure 2.1: Simple Model Overview

This report will reference to the control/patient spatial data as 'Record' *("voxel based features" in Figure 2.1)* **and will reference to the individual feature vectors as 'Datapoint'** *("single voxel based feature vector" and "non-voxel based feature vector" in Figure 2.1)***.** This logical differentiation is needed, as the model only operates on Datapoints and has no global context available, while some preprocessing and evaluation logic should happen on a Record level.

2.1 Preprocessing

2.1.1 Raw Data

All provided records are in the neuroimaging informatics technology initiative (NIfTI) format, first these are need to be understood and parsed. This format stores the raw output of the MRI record, and additionally an affine transformation matrix used for aligning different spaces.

2.1.1.a Available Data

The following records will be preprocessed and read, even if not all of them are going to be used later on it helps providing the largest possible flexibility.

Data	Shape	Range	Type	Space	Reference
dMRI	(118, 118, 60, 74)	$[0, 4096]$	uint	diffusion	diffusion
Diffusion FA	(118, 118, 60)	$[0,\ 2]$	float	diffusion	diffusion_fa
Diffusion MD	(118, 118, 60)	$[0,\ 0.01]$	float	diffusion	diffusion_md
Diffusion RD	(118, 118, 60)	$[0,\ 0.01]$	float	diffusion	diffusion_rd
T1	(208, 256, 256)	$[0,\ 1000]$	float	t1	t1
T1/T2	(208, 256, 256)	$[0, 1]$	float	d_aligned	t1t2
Cortical Targets	(118, 118, 60, 14)	$\{0, 1\}$	bool	diffusion	targets
Relative Connectivity	(118, 118, 60, 14)	$[0, 1]$	float	diffusion	connectivity
Streamline Image	(118, 118, 60, 14)	$[0, 5000]$	uint	diffusion	streamline
ROI Mask (Basal Ganglia)	(118, 118, 60, 2)	$\{0, 1\}$	bool	diffusion	mask_basal & roi
Brain Mask	(208, 256, 256)	$\{0, 1\}$	bool	t1	mask_brain
Basal Ganglia Segmentation	(208, 256, 256)	$[0, 58]$	uint	t1	basal_seg

Table 2.1: Raw Data

2.1.1.b Brain Mask

The provided dataset did not apply the brain masks for the T1 images out of the box so it can be done with a simple element wise multiplication of the T1 image and T1 mask.

2.1.1.c Registration

The process of aligning different records into the same native space is called "registration". The provided dataset comes with with 2 (3) different spaces, earlier referenced to as t1 and diffusion (and d_aligned). Most of the data are in diffusion space, thus it is logical to register the rest into the same space. After manual inspection, only 15 records required registration. Out of which 3 only required a tiny translation, and the rest 12 needed a complete affine registration.

The image T1/T2 is the odd one out, as it is inherently in a different space from diffusion (due to them being different resolution). But they are aligned into diffusion space. Although they do not need to be registered, this has to be taken into account later on.

2.1.1.d Normalization

The process of warping each brain into a common space is called "normalization". Applying the FNIRT warp fields are more or less straight forward, as two warp fields are provided, one for the diffusion space and one for the T1 space. Note that this process inherently contains the benefits of registration, as it is warping the different images into a common brain shape and space. This also paves the direction of future experiments, as it opens the door to working in either native and normalized space.

The only encountered obstacle was with the T1/T2 image. As it is aligned in diffusion space, but FNIRT convention ignores the affine transformation of the NIfTI format, thus making it's registration useless as the raw data of the t1t2 has nothing to do with the raw diffusion data

(due to them being different resolution). The solution is to apply an affine matrix to t1t2's raw data which transforms it into t1's raw data space, after which the t1's FNIRT warp field can be applied to the t1t2 image. This affine transformation matrix can be easily calculated from the already given matrices. Let A denote T1/T2's affine matrix and B denote T1's affine matrix (after registration), thus the matrix which transforms the T1/T2 into T1 space is $M = A \cdot B^{-1}$.

2.1.1.e Basal Ganglia Segmentation

As the tractography of the brain is performed on the diffusion image, it inherently means that the connectivity maps and the roi are in diffusion space. But the basal ganglia's subcortical segmentation is in T1 space. This means that even if they are registered in the same space, they will not have a pixel perfect union due to the different resolutions.

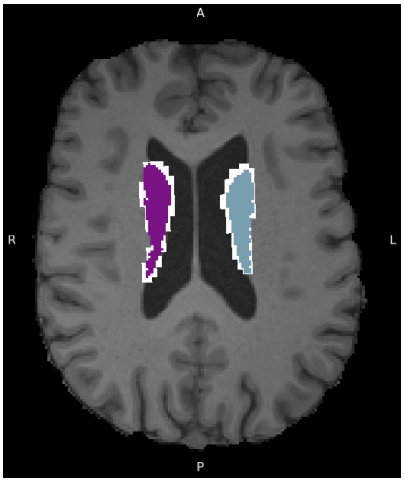

Figure 2.2: Basal Ganglia Subcortical Segmentation

The figure above visualizes the alignment of the Caudate subcortical region, where the white (larger) region is the Basal Ganglia mask from the diffusion space and the colored (smaller) regions are the Basal Ganglia segmentation from T1 space.

In order to keep the data consistent, mapping the segmentation to the Basal Ganglia mask can be done by assigning the same label for each voxel in the basal ganglia as the label of the closest voxel in the subcortical segmentation.

2.1.1.f N-Dim Array

The used NIfTI format stores the raw voxel space and the affine transformation matrix separately, in order to not loose data in the process of interpolating voxels when applying the transformation. But in order to consistently compare voxel data across different spaces (even if they are registered in the same space), the transformation needs to be applied, computing the interpolated voxels in the common space, bringing them into the same raw format of matching X, Y and Z dimensions, and discarding the stored affine matrices.

By default the native anatomical space's origin is near the center of mass of the brain, between the ears. This makes sense for medical professionals, when working with MRI records, but datastructure wise an array is indexed from 0. Meaning after applying the transformation to the voxel space, the yielded array will only contain one quadrant of the record as the rest are clipped in the

negative regions. Thus the space is also needed to be translated with the negative vector of the transformed space's bounding box's lower end.

The translation value can be calculated by calculating the boundaries of the transformed space's bounding box. Get all 8 corners of the voxel space and apply the transformation matrix to all of them. Then get the min-max coordinates along X, Y and Z from the 8 transformed vectors, yielding the lower and upper bounds of the transformed space's bounding box.

It is very important to use the same translation value across different spaces to properly align them in the native space. For example let D and T denote a diffusion and t1 records and M_D and M_T denote their respective transformation matrices. Let T_D and T_T denote their respective translation values. In order to properly align them we need to apply $A_D = (M_D \cdot T_D)$ matrix and $A_T = (M_T \cdot T_D)$ matrix to D and T respectively, with matching T_D translation values.

The last issue is the missaligned length of the dimensions of the T1 and diffusion records. This can be simply fixed by truncating the excess along each dimension.

2.1.1.g Uniform Shape

After aligning the data into the same space per record, it is still very likely that the individual records do not have a uniform shape. This is due to them being in native space, some records will contain a smaller volume brain, some will contain a larger, they will not be the same.

Due to the per-voxel based prediction model architecture this is not a problem, but fixing this for being able to use the data in a spatial model like a fully convolutional neural network (FCNN) can be simply solved by adding padding to the records in order to match their shapes.

Data	Volumes	Range	Type
diffusion	74	$[0, 4096]$	float16
diffusion_fa	1	$[0, 2]$	float16
diffusion_md	1	$[0, 0.01]$	float16
diffusion_rd	1	$[0, 0.01]$	float16
t1	1	$[0, 1000]$	float16
t1t1	1	$[0, 1]$	float16
targets	14	$\{0, 1\}$	bool
connectivity	14	$[0, 1]$	float16
streamline	14	$[0, 5000]$	float16
mask_basal	2	$\{0, 1\}$	bool
mask_brain	1	$\{0, 1\}$	bool
basal_seg	6	$\{0, 1\}$	bool

Table 2.2: Uniform Data

2.1.2 Quality Control

Having a low count of records means that if there are even just a few outliers, it can heavily affect the end result. Thus all data were manually inspected to make sure they are as clean as possible.

2.1.2.a Mismatched Data

Looking through the diffusion, diffusion_fa, diffusion_md and diffusion_rd images, 2 records' FA, MD and RD images were seemingly from completely different patients. Thus the FA, MD and RD images were omitted for 2 records.

2.1.2.b Garbled Data

Looking through the subcortical segmentation of the Basal Ganglia revealed that 1 record had a garbled segmentation. Thus, said basal_seg image was omitted for 1 record.

And one record had a garbled T1 FNIRT warp field. Said record was entirely omitted from the normalized set of records.

2.1.2.c Missing Data

Looking through the relative connectivity and streamline images, 3 records were missing these images, said 3 records were completely omitted, as these records are effectively missing the labels.

And the t1t2 images were missing for 10 records, but these were not omitted completely as the t1 images were present for these records, thus experiments only concerning the t1 can have a bit more available data.

2.1.3 Radiomics Features

Although the term is not strictly defined, radiomics generally aims to extract quantitative, and ideally reproducible, information from diagnostic images, including complex patterns that are difficult to recognize or quantify by the human eye. [4] Using these features is key, as there are not nearly enough data for neural network (NN) based features extraction such as a convolutional neural network (CNN).

Extracting the voxel based radiomic features has two main parameters to tune, the bin width and the kernel width. Where the binning parameter(s) influence how the intensity values of the image are binned, and the kernel size influences the size of the 'sliding window' similar to a convolution.

The two approaches for binning are absolute discretization and relative discretization. Where in the prior one, a fixed bin width is chosen and in the latter one, a fixed number of bins are chosen and the bin width scales relatively according to the min-max voxel values. This study found that "The absolute discretization consistently provided statistically significantly more reproducible features than the relative discretization." [5] Relying on this information, the obvious choice to start with is the absolute discretization.

The bin width and the kernel width will be tuned in later experiments. And possibly features calculated with different setting will be concatenated and used simultaneously for better results. The used default values will be 25 and 5 for the bin and kernel widths respectively.

The following types of radiomic features will be used:

Feature Type	Number of Features
first order	18
gray level co-occurrence matrix (GLCM)	23
gray level size zone matrix (GLSZM)	16
gray level run length matrix (GLRLM)	16
neighbouring gray tone difference matrix (NGTDM)	5
gray level dependence matrix (GLDM)	14
3D shape	17

Table 2.3: Radiomic Feature Types

2.1.3.a Voxel Based

The 92 features in Table **??** will be calculated voxel based. Shape features do not makes sense to calculate voxel based as it would just describe the shape of the used kernel, which is constant and independent from the input image.

2.1.3.b Non-Voxel Based

However, the additional shape features in Table **??** do make sense for the non-voxel based features. As it can be computed for each target region, both hemispheres of the ROI and the entire brain.

2.1.4 Coordinates

One additional input that can be included in the experiments is the coordinates. Although this approach only makes sense in normalized space, where the images from different records are aligned. This theoretically would allow the model to learn certain anatomical markers based on the location of the voxel, adding a type of global context to the input of the model.

Furthermore, this approach can be adopted to the native space, by constructing the normalized coordinate map and then 'de-normalizing' them with an inverse FNIRT warp field.

2.1.5 Data Augmentation

The only data augmentation that makes sense involves applying small rotation values to the input images in their native space before calculating radiomic features. Applying transformations to the already extracted features is illogical, as interpolating between voxels in feature space is unlikely to yield the same results as computing features after transforming the input images. In summary, any spatial data transformations should be performed upstream. Furthermore, data augmentation only makes sense in native space, as by definition such transformations would make the normalized image pointless.

2.1.6 Scaling and Normalization

As the extracted features have very different ranges, it makes sense to follow the standard practice of scaling the data to a fixed range. Inspecting the histogram of some of the radiomic features reveals that most of them follow a bell curve with moderate standard deviation, such as Figure 2.3 (Firstorder Energy).

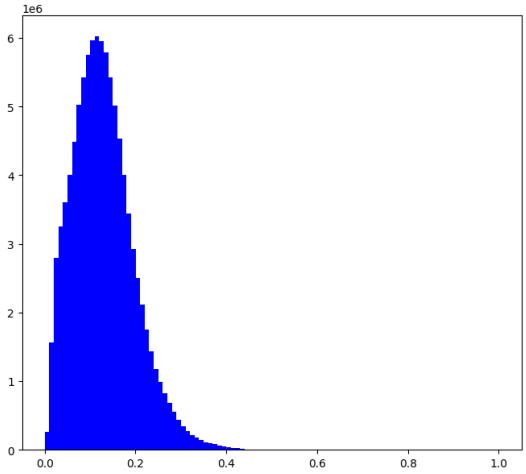

Figure 2.3: Histogram: Firstorder Energy

However, some other features like Figure 2.4 (GLDM Small Dependence High Gray Level Emphasis) and Figure 2.5 (NGTDM Busyness) have a very skewed distribution, the latter one being the most extreme case. This skewing can be mitigated by applying logarithm to the offending features.

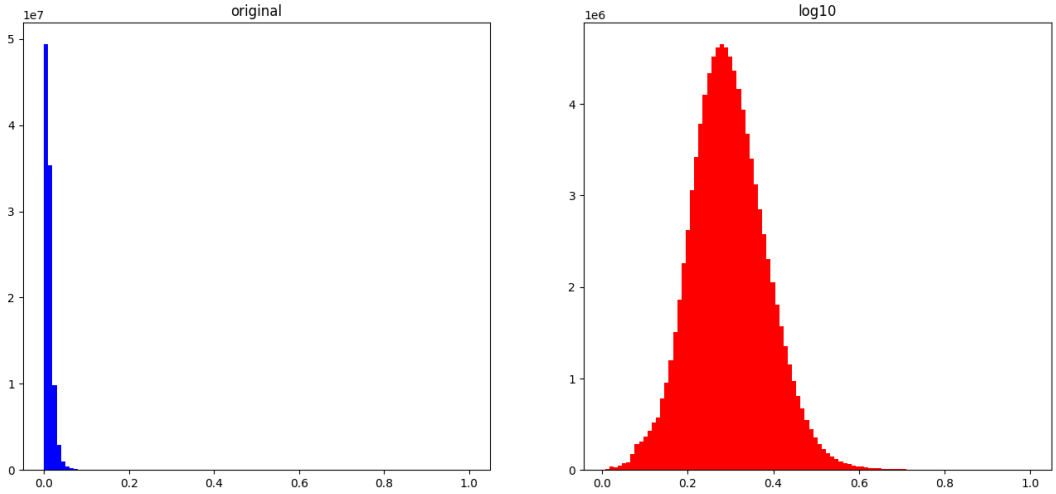

Figure 2.4: Histogram: GLDM Small Dependence High Gray Level Emphasis

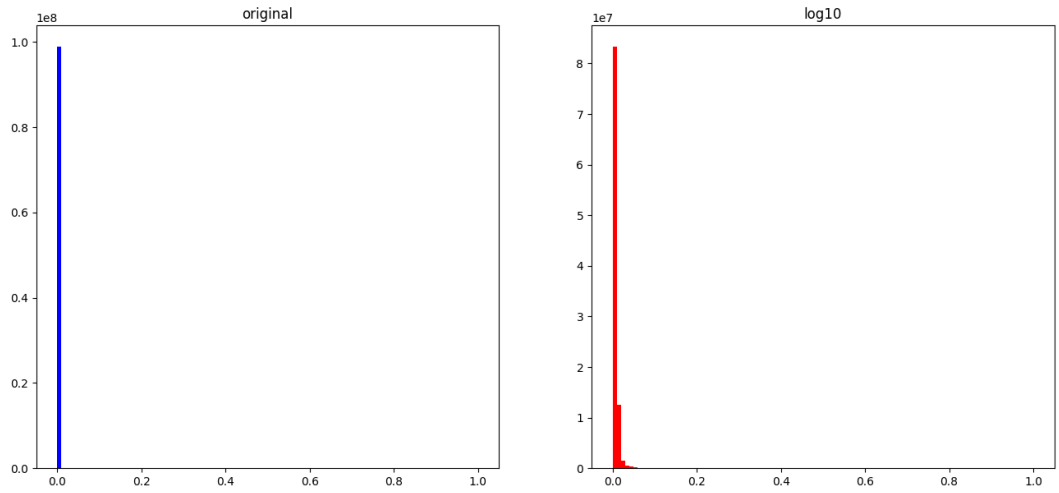

Figure 2.5: Histogram: NGTDM Busyness

Besides the standard benefits of making the optimization process more stable and efficient, and reducing the sensitivity to outliers. It also have some less evident benefits.

Although it is very subtle, but storing these records in float16 inherently looses some information. This loss is not a problem for the features that have a healthy distribution, but in the more extreme cases it can cause compression artifacts visible even to the naked eye, such as the very subtle loss of detail in Figure **??**. And in the most extreme case it can even render the entire feature useless like in Figure **??**. While the normalized features have no problem storing this fine detail in float16.

This makes the system much more robust from a practical perspective; as depending on the hardware, some GPUs are much more efficient at computing in float16. And it also halves the memory and storage requirements, as in float32 a sinlge MRI image of 92 volumes (for the 92 features) takes up around 1GB of space.

Selecting which features need normalization is done programmatically, and the exact selection criteria is detailed in Appendix **??**.

2.1.7 Data Balancing

Working with highly unbalanced data can be challenging, and balancing it does not necessarily going to help the model's generalization capability. Thus, a method for partially balancing the data will be used, where the bins of the unbalanced data will be up-sampled by a ratio of the difference of the number of datapoints in the bin (compared to the bin with the maximum number of datapoints). Figure 2.6 demonstrates how a ratio 1 means perfectly balanced data, 0 means unbalanced data. And how the ratios in between are approximately preserving the shape of the distribution and partially balance the data.

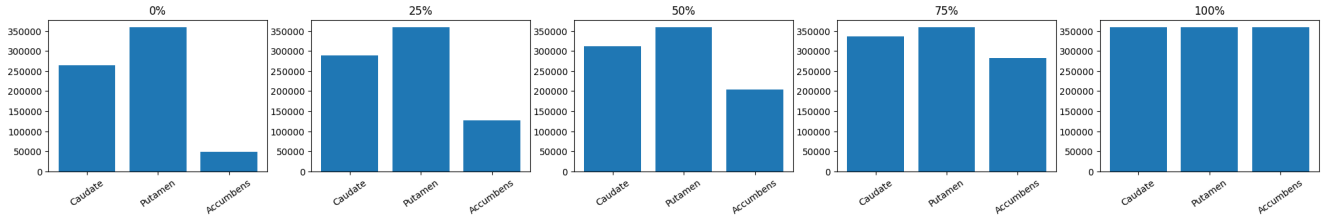

Figure 2.6: Balance: Subcortical

For the diffusion_md and diffusion_fa, which are regression problems and have continuous labels, binning can be used to create artificial groups which can be balanced.

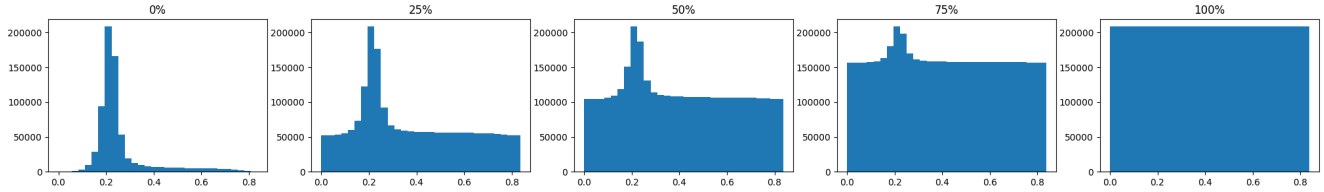

Figure 2.7: Balance: Diffusion MD

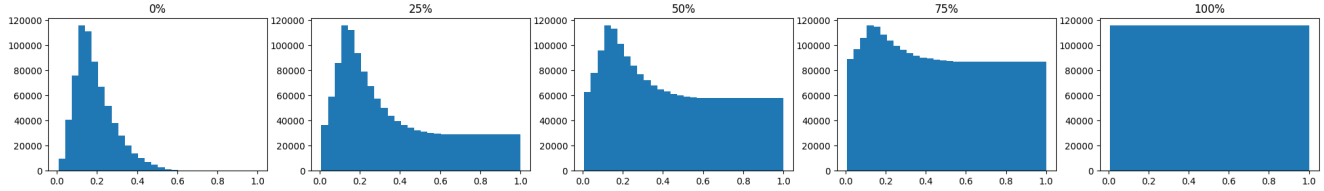

Figure 2.8: Balance: Diffusion FA

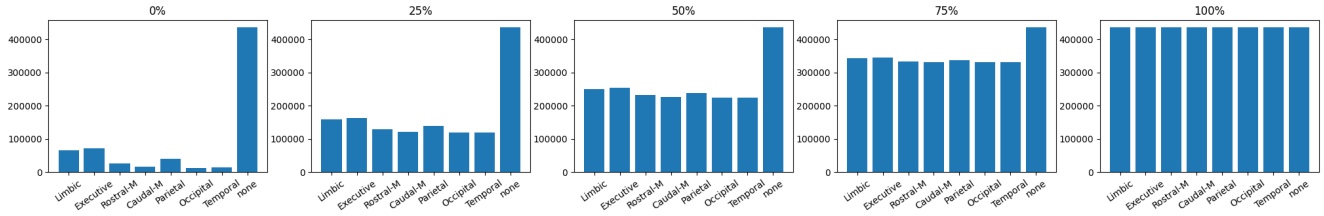

Figure 2.9: Balance: Relative Connectivity (thresholded at 0.6 & binarized)

2.1.8 Clinical Data

There are additional clinical data available for the Patient records. Disease severity can be characterized in terms of CAG Age Product (CAP) score. Providing a measure of cumulative exposure to the mutant HTT gene. [6] This widely accepted and used CAP score is available for all patients.

Another, newer metric for characterizing disease severity is the composite Unified Huntington's Disease Rating Scale (cUHDRS) [7]. This is calculated from 4 other basic metrics: Total Functional Capacity, Total Motor Score, Symbol Digit Modalities Test and Stroop Word Reading. These are available for most patients, with a handful exceptions.

And there are a total of 91 available clinical features, with relatively a lot of missing data on some of these features. There are 8 additional patients available in the clinical data. These can be

used to aid the data imputation for the missing values, and can be omitted afterwards, as these have no corresponding MRI records.

All clinical features were scaled the range of 0-1 with min-max scaling (per feature). And euclidean distance was used for the following imputation process. The imputation strategy itself consisted of 2 steps, first the few missing cUHDRS values were imputed from the CAP score. And then the remaining features were imputed from the combined CAP and cUHDRS values.

2.1.9 Relative Connectivity

Relative Connectivity describes the ratio of the number of streamlines going into each cortical target. This means that the Streamline record can be converted into Relative Connectivity by simply dividing by the total number of streamlines in each voxel. With the additional inbetween step of filtering some noise, by thresholding the streamlines at 250 (5% of the total 5000 streamlines), meaning any voxel which has less than 250 streamlines to a target region are set to zero.

Then the relative connectivity could be converted into a label, by picking the label of the cortical target to the highest connection per voxel. However this would yield very noisy labels, as these ratios can be quite balanced between the multiple cortical targets, for example voxels with ratios of 0.31/0.29/0.3/0.1. To mitigate this, the relative connectivity can be thresholded at a value higher than 0.5, meaning a label can only be picked for a voxel if at least half of the connections are going to a single target.

But this also means that there can be voxels without labels. This can be dealt with introducing an artificial 'Not Connected' label for these voxels.

To achieve the best results and filtering, 0.6 was chosen for the thresholding value, as it also filters potential 50-50 situations and only allows labeling strong connections.

2.2 Evaluation

2.2.1 Train, Validation and Test Splits

There are 2 important aspects when splitting the data into Train/Validation/Test groups. In order to truly validate the model's generalization capability, the split must happen on a record level and not on a datapoint level. This means that our model can only learn on certain records, and it can be validated on records that it never seen before, not even partially. This has the consequence of that the split will not follow the defined ratio on a datapoint level, as it could happen that by pure chance the train split contains records with larger volumes, resulting in having a bit more datapoints than the validation split.

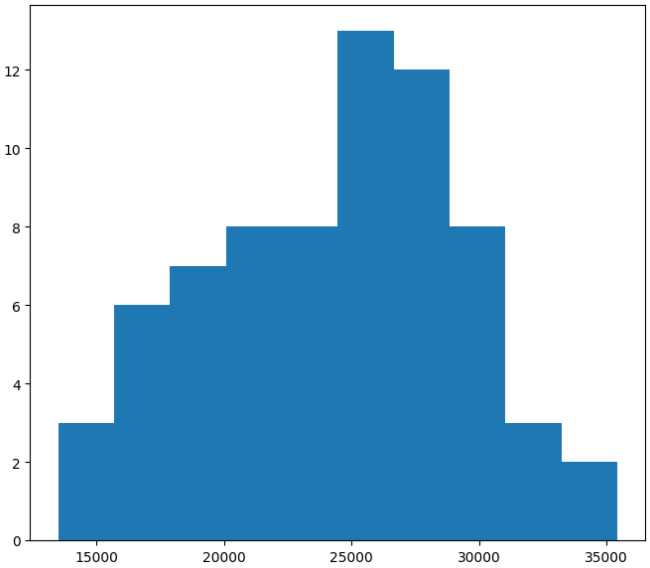

Figure 2.10: Distribution of Records in relation to Datapoints

In practice the lower end of datapoint count per record is around half of the higher end. Figure 2.10 shows the distribution of both Control and Patient records and both Left and Right datapoints. During experimentation, with 0.8 Train split and 0.5 Validation/Test split, the datapoint ratios stayed in the range of 0.8 ± 0.02 and 0.5 ± 0.06 for the two split ratios.

Furthermore to avoid introducing bias in the case of experiments with mixed Control and Patient records, the ratio of Controls/Patients must be constant across the different splits. This is required as Controls and Patients can have vast differences due to neurodegeneration. An extra caveat is having another ratio that must also be kept constant for the same reason, which is the symptomatic and asymptomatic patients, as they can also have vast differences due to different stages of neurodegeneration.

2.2.2 Accuracy and Pearson Correlation

There will be 3 groups of metrics for evaluating each model. First is the 'raw' (will also be referenced as 'train') metric group (train/val/test), which is computed on the datapoints that were extracted with the same hyperparameters as the datapoints during the training process. Meaning that this metric group best reflects how the model performs on the different splits, such as if the model was trained with a balancing of 0.5, all splits will be balanced the same way and the metrics will be computed on a datapoint level (the same way as how it is naturally computed in the loss function).

The second and third groups are for comparing model performances in between models and are for practical evaluation. The difference is that the metrics in this case are computed for each record, and then averaged out inbetween records. This means that it is computed on a record level instead of a datapoint level, resulting in the elimination of potential bias coming from the deviation from the number of datapoints per records. It also means that these metric groups will inherently ignore data balancing, as it operates on a record level.

And the 2nd metric group is computed in native space, while the 3rd is computed in normalized space. This means that if the model operates in native space, the normalized metrics will be computed by predicting the datapoints for each record, then the spaital record is reconstucted

from the datapoints and warped to normalized space, and then the datapoints are extracted from the normalized spaital prediction, and compared against the normalized labels (This process would be computationally quite expensive, so the implementation does not follows this exact logic, but numerically it is doing the same; more information on this in Appendix **??**). This way the models can have comparable metrics even if they operate in different spaces.

Experiments

The following hyperparameters were constant during all of the experiments:

Hyperparameter	**Value**
Train Split	0.8
Validation/Test Split	0.5
Model Type	feedforward neural network (FNN)
Optimizer	Adam

Table 3.1: Hyperparameters: Common

3.1 Subcortical Segmentation

This simple problem did not need a lot of tuning, as it was working very well almost from the start. The following set of hyperparameters were constant during these experiments:

Hyperparameter	**Value**
Control/Huntington Datapoints	Control Only
Left/Right Hemisphere Datapoints	Both
Space	Native
Image	T1
Scaling/Normalization	Normalized Voxel Based Features
Hidden Layers	$1024 \rightarrow 512 \rightarrow 256 \rightarrow 128$
Loss	Categorical Crossentropy
Activation	Sigmoid (softmax for the output layer)
Learning Rate	0.001
Batch Size	10000
Early Stopping Patience	7

Table 3.2: Hyperparameters: Subcortical

The reasoning behind the initial choices of these parameters are straight forward. The T1 image and native space were chosen, because those are the simplest to acquire in practice. Thus if the model is doing great on those, there is no need for more complicated inputs. Including both hemispheres would hopefully result in a model which can generalize better. Only using control datapoints should translate into less variance between the general characteristics of the datapoints, as it does not contain patients with neurodegeneration. The number and sizes of the hidden layers were choosen based on the potential size of the input layer, which should range from 92 (single set of voxel based features) up to $\sim 1000 - 2000$ including many different kernel sizes and non-voxel based features as well. Categorical crossentropy loss function and the output layer's softmax activation function are standard practice for a classification problem. The Sigmoid activation function should work fine without having to deal with exploding gradients and dieing

relu problems. Learning rate is the default learning rate of the Adam optimizer in TensorFlow. And a batch size of $10,000$ seems appropriate for a train split size of $1,000,000$ datapoints. And the early stopping patience of 7 epochs should also be good enough to prevent overfitting and stop the training in time, but it will be evaluated based on the learning curves and the accuracy of the model.

The used metric for evaluating model performance on the Train/Validation/Test splits is Accuracy. The 'k' and 'b' notations stand for kernel and bin, where k5 means a kernel width of 5mm, b25 means an absolute bin size of 25, and b10r means relative binning with 10 bins. And in the case of multiple kernel sizes denoted by a dash, it naturally only means odd kernel sizes. Tuning the rest of the hyperparameters were done in the following experiments:

	Experiment	**Train**	**Val**	**Test**	**Input Layer**
1.	**Voxel Features k5_b25**	68.9	69.1	72.4	92
2.	*Voxel Features k5_b25* **Non-Voxel Features of Target Regions b25**	73.2	68.9	72.5	1576
3.	*Voxel Features k5_b25* **Non-Voxel Features of ROI b25**	75	74.3	78.5	304
4.	*Voxel Features k5_b25* *Non-Voxel Features of ROI b25* **Non-Voxel Features of Brain b25**	74.5	70.4	70.3	410
5.	*Voxel Features k5_b25* **Non-Voxel Features of ROI b10 b25 b50 b75**	71.2	70.6	74.3	856
6.	**Voxel Features k5_b25 - k21_b25** *Non-Voxel Features of ROI b25*	94.5	94.1	95.1	1040
7.	**Voxel Features k5_b25 - k21_b25**	95	94.6	93.7	828
8.	*Voxel Features k5_b25 - k21_b25* *Non-Voxel Features of ROI b25* **Balance Ratio 0.5**	94.9	94.4	94.9	1040
9.	*Voxel Features k5_b25 - k21_b25* *Non-Voxel Features of ROI b25* **Balance Ratio 1**	95.9	95.5	95.9	1040

Table 3.3: Hyperparameter Tuning: Subcortical

The best performing model was in experiment number 9, where it achieved a 96% accuracy with practically no overfitting. The biggest improvement during the experiments was to include many different kernel sizes for the voxel based features. The additional non-voxel based features of the ROI yielded a small improvement. And balancing the data yielded a marginal improvement, by reducing overfitting.

Examples of the true/predicted records can be found in Figures **??** **??** **??**. And the loss training curves can be found in Figure **??**.

3.2 Methodology

The experimentation from this point on, will be divided into 4 main groups:

- Native - T1

- Native - T1/T2
- Normalized - T1
- Normalized - T1/T2

The same set of core experiments will be run for all 4 groups, and some additional experiments will be run per group, depending on how they perform. The experiments will consider the following aspects:

- Single/Many Different Kernel Sizes for Voxel Based Features
- Additional Non-Voxel Based Features
  - Single/Many Different Bin Sizes
- Control/Patient/Both Records
- Left/Right/Both Hemisphere Datapoints
- Additional Clinical Features for Patient Records
- Additional Coordinate Map Features
- Scaled Voxel Based Features (not normalized)
- Different Bin Sizes for Voxel Based Features
- Different Balance Ratios

3.2.1 Missing Records

In order to be completely fair when comparing model performances, only records should be used which are available for all 4 groups of experiments. In practice the following records were missing:

Record	**Missing Amount**
Normalized	1
T1/T2	10
Diffusion FA & MD	2

Table 3.4: Missing Records

This meant that for the Diffusion FA & MD experiments there were a total of 13 records omitted, yielding 57 records in total, out of which 29 are Control and 28 are Patient records. And for the Relative Connectivity experiment, 11 records were omitted, yielding 59 records in total, out of which 30 are Control and 29 are Patient records.

As additional experiments for the groups with more available data (such as T1, where 10 more records could be included), these records can be appended to the train split on the best performing model, feasibly increasing model performance.

3.2.2 Architecture Tuning

For the best performing model, the architecture will be further tuned, considering the following aspects:

- Number of Layers and Layer Sizes
- Activation Function
- Batch Size
- Learning Rate
- Dropout Normalization
- Early Stopping Patience

3.3 Diffusion Fractional Anisotropy Regression

All numerical results of the experiments can be found in Tables **??** - **??**. The baseline starting experiment is trying to predict the FA from a single set of voxel based radiomic features, with a kernel size of 5. And with the same starting hyperparameters (Table 3.2) that were also used in the subcortical segmentation (with exception of the used loss function, which is Mean Squared Error instead of the Categorical Crossentropy).

The next few experiments were trying to determine how does each set (target regions, roi, and entire brain) of non-voxel based features affect the model performance. Between the 4 different group of experiments (Native-Normalized & T1-T1/T2) the observations were more or less consistent, with the final consensus being that the inclusion of the entire brain's non-voxel based features are yielding the best results, with an improvement of 5-10% better correlation compared to the baseline.

Including many different bin sized non-voxel based features worsened the model performance by 0-3%.

The biggest improvement was the inclusion of many different kernel sized voxel-based features, with an improvement of 10-15%. And surprisingly after removing the non-voxel based features, it further improved the performance of the T1 experiments by 1-2%, while worsening the T1/T2 experiments by 0-1%. Running this experiment on the Patient records, resulted in the models performing even worse with the additional non-voxel based features by 8-10%.

The experiments consistently showed the model performing much better on the Control records, compared to the Patient records, with much less overfitting and better correlation by 5-10%.

The inclusion of the clinical features were behaving inconsistently between the 4 groups of experiments. For the native T1, including the CAP and cUHDRS features marginally improved the model performance, and for the normalized T1/T2 it improved model performance by 4-5%. While for the native T1/T2 and normalized T1, it worsened the model performance by 5-10%. The overall Patient records even with the best performing clinical features, were still performing worse than the Control records.

Mixing Control and Patient records still performed worse than Control records only, but only with 1-5% correlation.

Including coordinates, did not affect the T1 models' performance, but it did marginally increase the T1/T2 models' performance by 1-2%.

Only using min-max scaling, and not normalizing the datapoints, resulted in marginally worse performance.

Increasing the bin size for the voxel based radiomic features marginally decreased the model performance.

Balancing the data was a bit inconsistent between the groups of experiments, but the balance ratio of 1 usually resulted in a marginally worse, and a balance ratio of 0.5 resulted in a marginally better performance.

Re-including the 10 extra T1 records as part of the training split for the T1 experiment, only resulted in a marginal improvement for the native space, and a 2% improvement for the normalized space.

After combining all of the best configurations, the best performing model was the T1 normalized model, with Control records only, and re-included T1 records, without any additional non-voxel based features. It reached a final correlation of **84.4/84.6/82.8** for the train/val/test splits in native space, and **84.6/84.9/82.9** in normalized space.

After tuning the model architecture, by searching different layer sizes and numbers, activation functions, dropout normalization, adjusting learning rate and batch size, it only increased the model's overfitting, without any actual benefits.

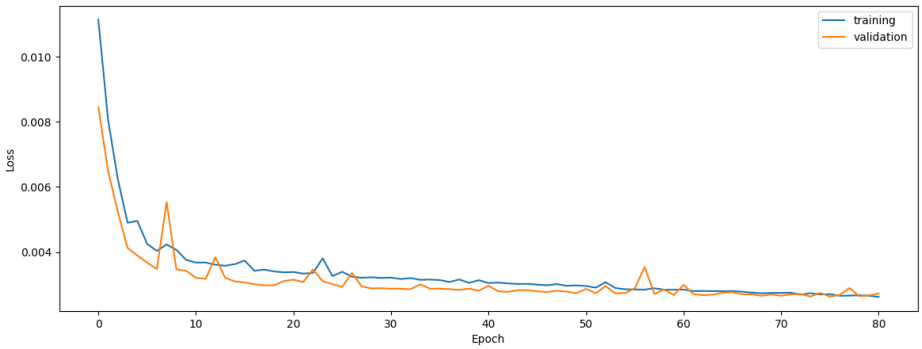

Figure 3.1: Training Curve: Diffusion Fractional Anisotropy

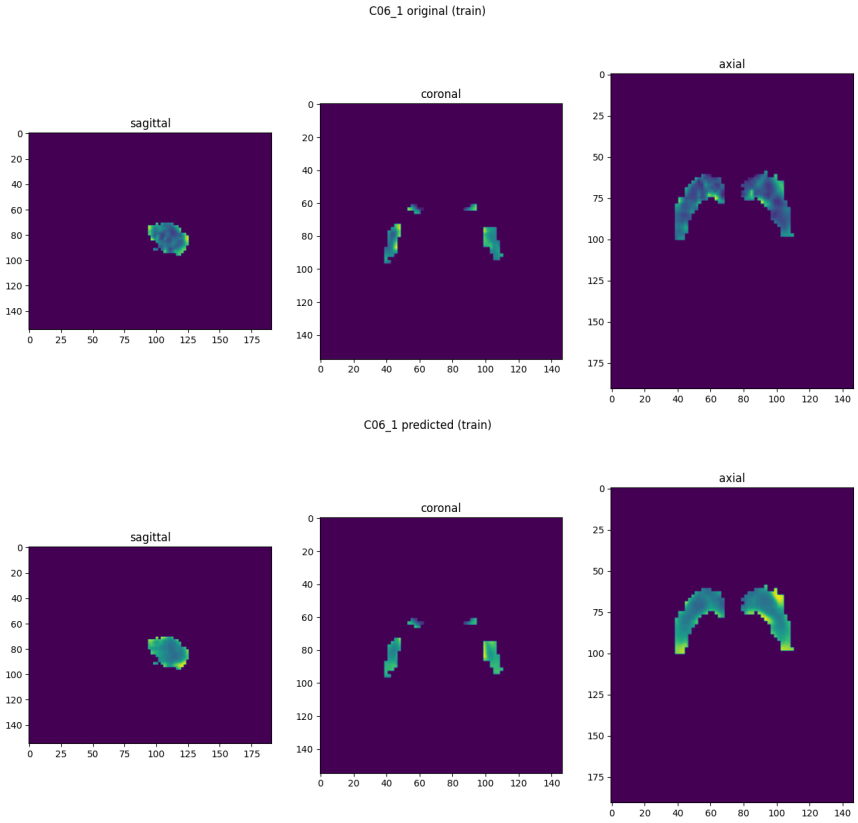

Figure 3.2: Train Predictions: Diffusion Fractional Anisotropy

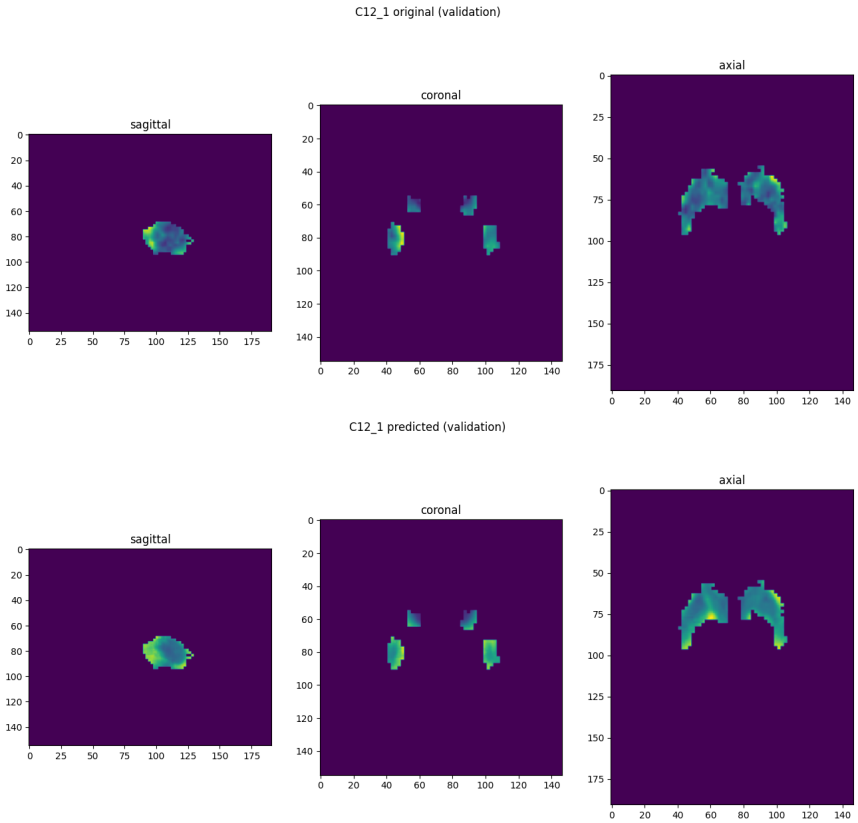

Figure 3.3: Validation Predictions: Diffusion Fractional Anisotropy

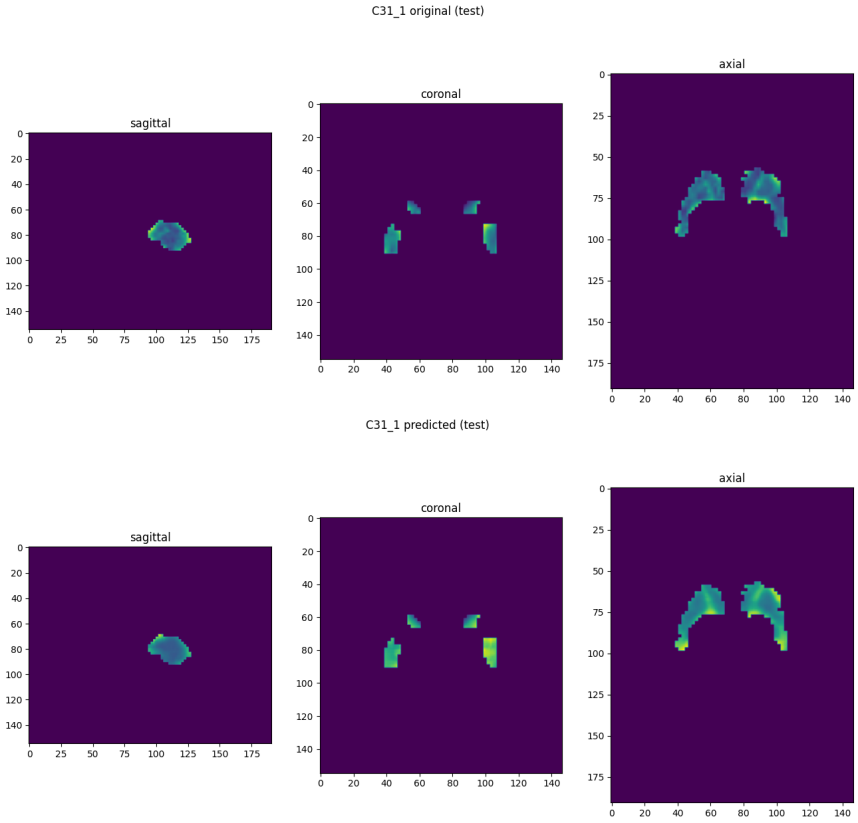

Figure 3.4: Test Predictions: Diffusion Fractional Anisotropy

3.4 Mean Diffusivity Regression

A similar set of experiments were run for predicting MD with the numerical results available in Tables **??** - **??**. But these experiments were performing very well out of the box, even the baseline Native T1 experiment with a single set of voxel based features resulted in a correlation of 94% without any overfitting.

No significant observation can be made here, besides the Patient records performing marginally worse.

The best performing model was Native T1, on the Control records only, with the additional non-voxel based features of the entire brain, and many different voxel based kernel sizes. It reached a final correlation of **94.7/95.5/95.1** for the train/val/test splits in native space, and **95.4/95.7/96.3** in normalized space.

After tuning the model architecture, by searching different layer sizes and numbers, activation functions, dropout normalization, adjusting learning rate and batch size, it could not increase the model performance, not even on the train split. Indicating that this is the absolute best this model can do.

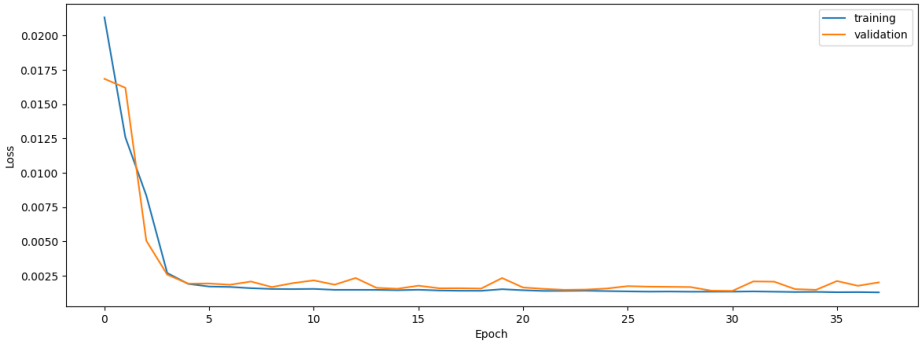

Figure 3.5: Training Curve: Mean Diffusivity

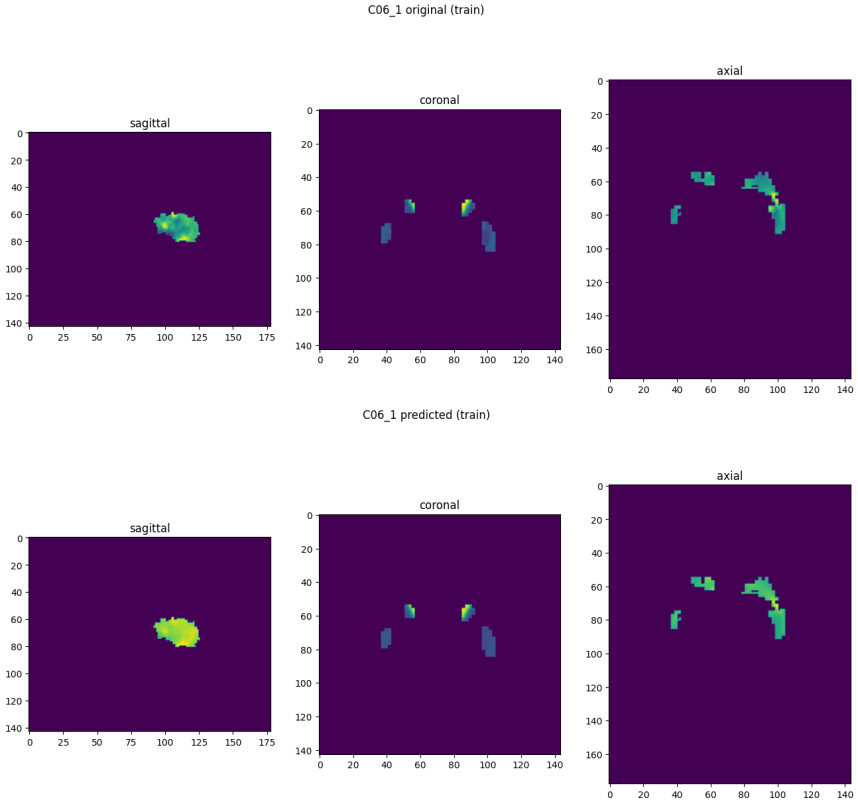

Figure 3.6: Train Predictions: Mean Diffusivity

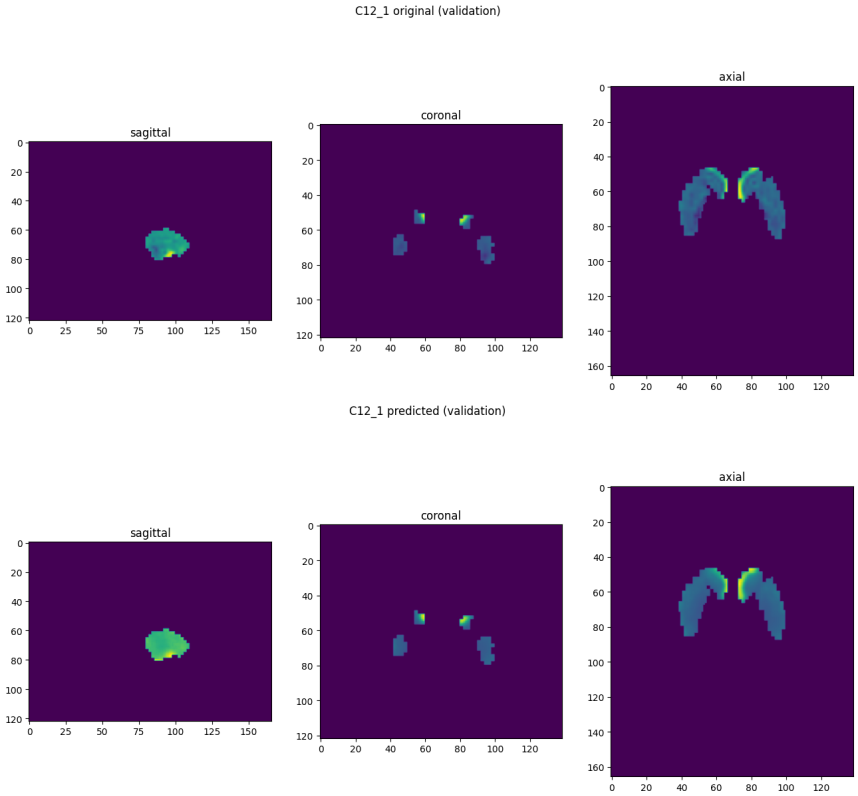

Figure 3.7: Validation Predictions: Mean Diffusivity

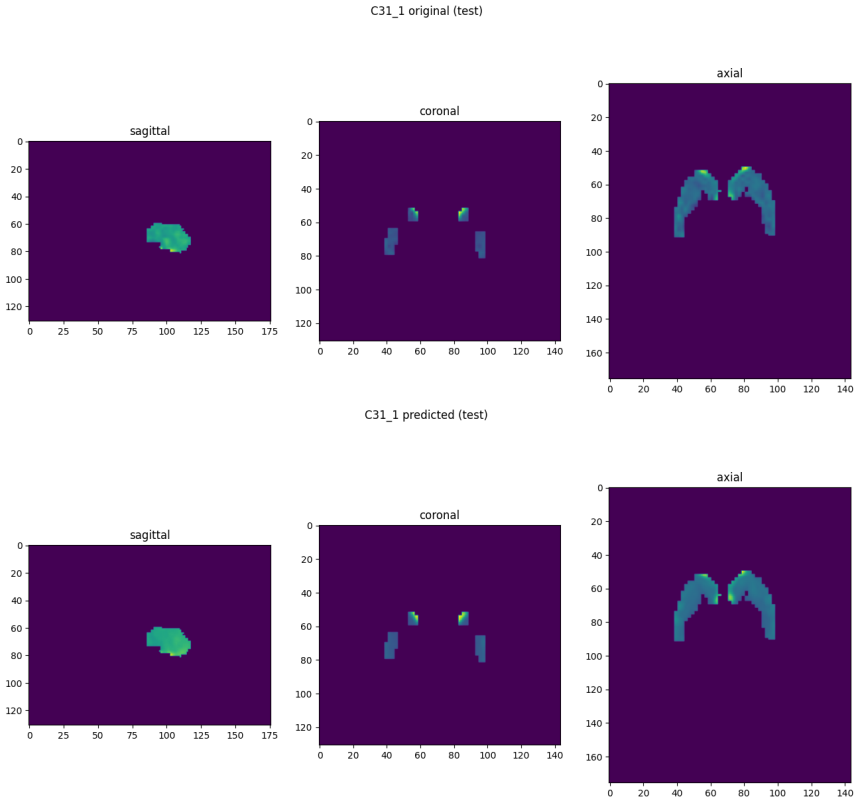

Figure 3.8: Test Predictions: Mean Diffusivity

3.5 Relative Connectivity Segmentation

All numerical results of the experiments can be found in Tables **??** - **??**. The baseline starting experiment is trying to predict the Relative Connectivity (preprocessed with the method described in subsection 2.1.9) from a single set of voxel based radiomic features, with a kernel size of 5. And with the same starting hyperparameters (Table 3.2) that were also used in the subcortical segmentation.

The next few experiments were trying to determine how does each set (target regions, roi, and entire brain) of non-voxel based features affect the model performance. Between the 4 different group of experiments (Native-Normalized & T1-T1/T2) the observations were more or less consistent, with the final consensus being that the inclusion of non-voxel based features does not increase model performance.

The biggest improvement was the inclusion of many different kernel sized voxel-based features, with an improvement of 5-10%.

The experiments consistently showed the model performing much better on the Control records, compared to the Patient records, with much less overfitting and better correlation by 2-5%.

The inclusion of the clinical features were behaving inconsistently between the 4 groups of experiments. For the native T1, including the CAP and cUHDRS features marginally improved the model performance, and for the normalized T1/T2 it improved model performance by 4-5%. While for the native T1/T2 and normalized T1, it worsened the model performance by 5-10%. The overall Patient records even with the best performing clinical features, were still performing worse than the Control records.

Mixing Control and Patient records still performed worse than Control records only, but only with 1-5% correlation.

Including coordinates, did not affect the T1 models' performance, but it did marginally increase the T1/T2 models' performance by 1-2%.

Only using min-max scaling, and not normalizing the datapoints, resulted in marginally worse performance.

Increasing the bin size for the voxel based radiomic features marginally decreased the model performance.

Balancing the data was a bit inconsistent between the groups of experiments, but the balance ratio of 1 usually resulted in a marginally worse, and a balance ratio of 0.5 resulted in a marginally better performance.

Re-including the 10 extra T1 records as part of the training split for the T1 experiment, only resulted in a marginal improvement for the native space, and a 2% improvement for the normalized space.

After combining all of the best configurations, the best performing model was the T1 normalized model, with Control records only, and re-included T1 records, without any additional non-voxel based features. It reached a final correlation of **84.4/84.6/82.8** for the train/val/test splits in native space, and **84.6/84.9/82.9** in normalized space.

After tuning the model architecture, by searching different layer sizes and numbers, activation functions, dropout normalization, adjusting learning rate and batch size, it only increased the model's overfitting, without any actual benefits.

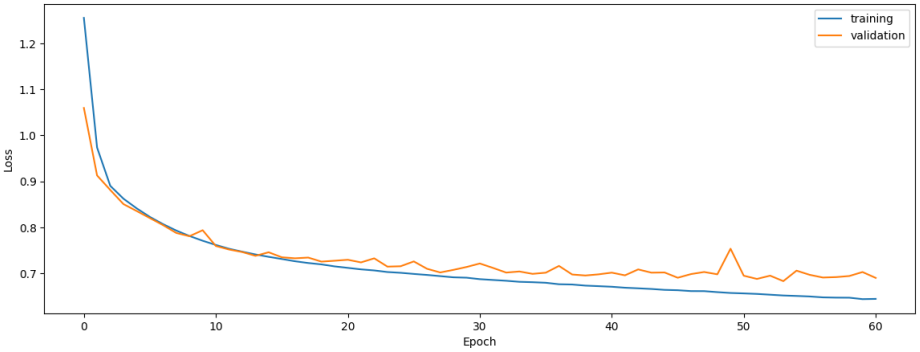

Figure 3.9: Training Curve: Relative Connectivity

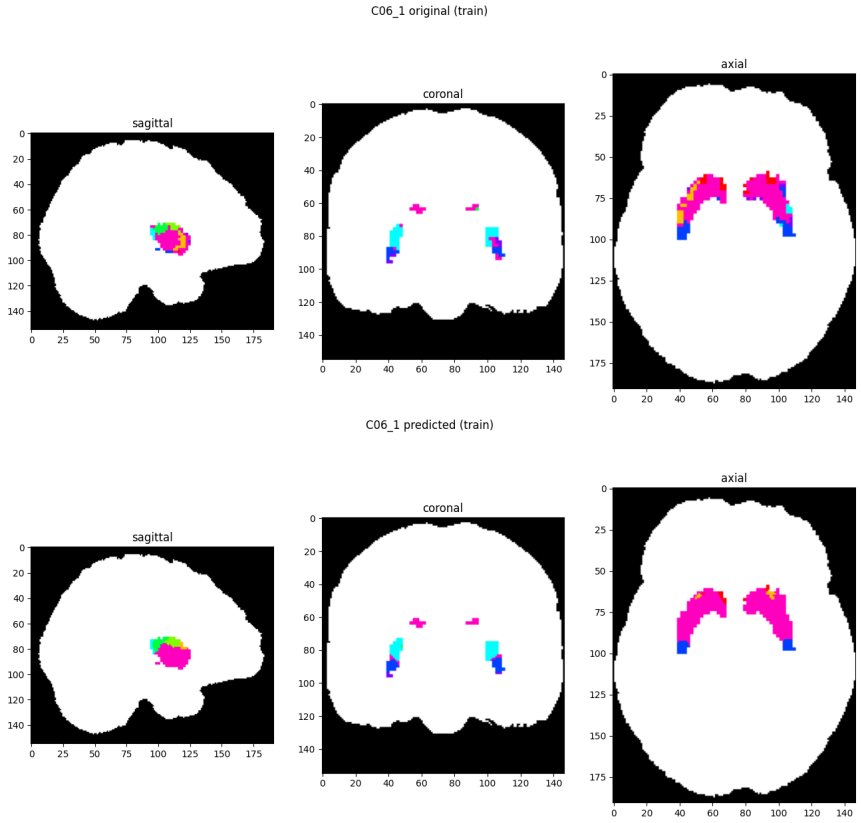

Figure 3.10: Train Predictions: Relative Connectivity

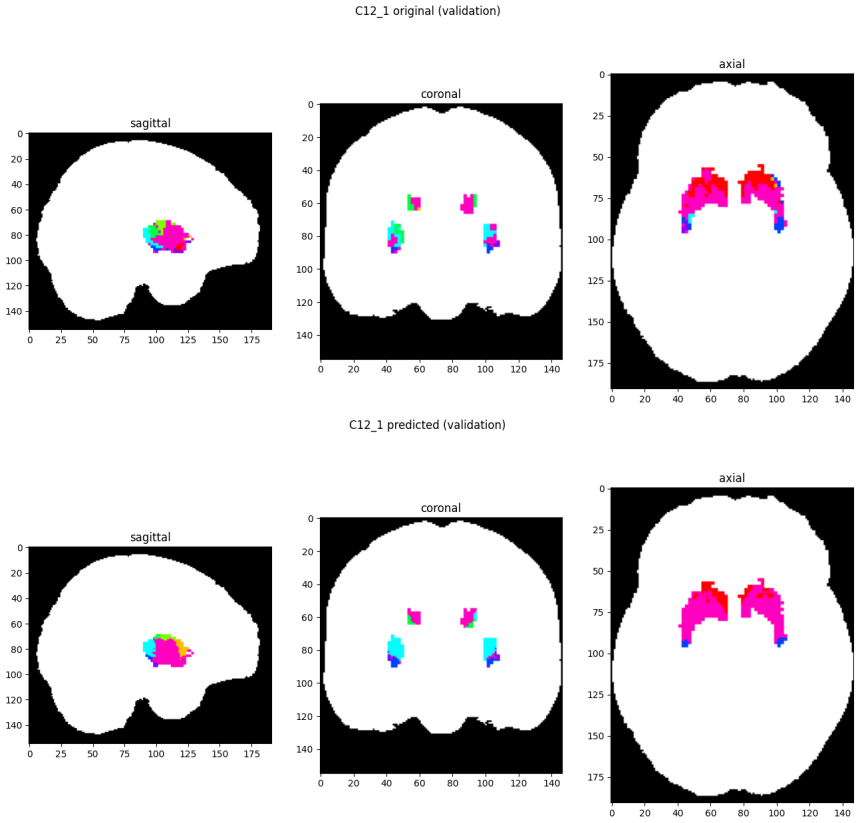

Figure 3.11: Validation Predictions: Relative Connectivity

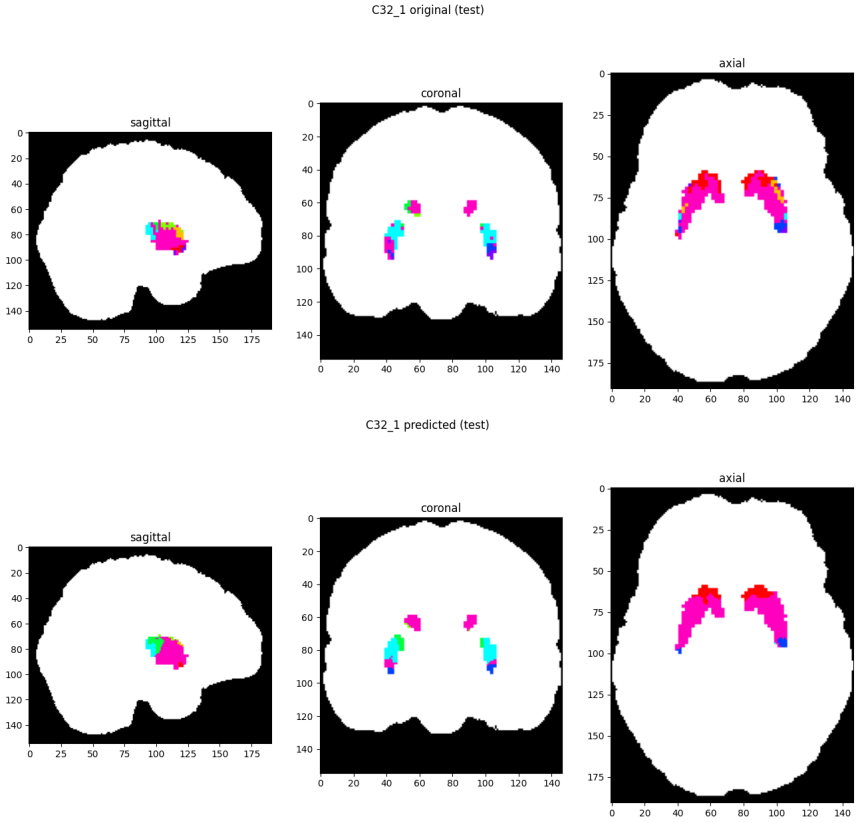

Figure 3.12: Test Predictions: Relative Connectivity

3.5.1 Exhaustive Sequential Backwards Feature Selection

Exhaustive sequential backwards feature selection, is a about training the model iteratively by removing a single feature at a time, going through all features. After an iteration is done, the best performing model is chosen, and the corresponding feature is permanently removed before the next iteration, where the model goes through the remaining $n - 1$ features.

And the stopping criteria can be varied from setup to setup, but in this case the evaluation metric was chosen to be the validation raw accuracy, and a stopping point of where all runs in the iteration performed worse than the baseline (the model with all features included) by more than 2%.

Feature selection was only ran for the Relative Connectivity Segmentation problem, due to it being time consuming and computationally intensive. This was executed on a network cluster, where the cluster head would give out tasks (a task being a model to train with a set of features to exclude), and the workers returning the validation accuracy.

There were 2 logistical oversights that were not mitigated in time. The first mistake being that the execution of the feature selection was rushed due to the longevity of the task, and the rest of the hyperparameters (besides the excluded features) were chosen before finishing the basic experimentation. And the second being an indirect consequence to it being rushed, and it was the usage of a bugy code that was responsible for balancing the data, as the bug was found after already executing the feature selection.

Due to these mistakes, the selected features turned out not to be very efficient with the final hyperparameters. The feature selection was not executed again due to time and resource limitations. More information detailing this mistake and oversight can be found in Future Improvements 5.1 subsection.

Nevertheless the expected result of re-running the feature selection would be something similar to the current sub-optimal result. Where around 1/3rd of the features could be excluded before stopping and a maximum increase of 2% for the validation accuracy. The first features to be excluded was predominantly from the firstorder feature class, followed by the GLCM feature class.

Iteration	**Excluded Feature**	**Accuracy**
0.	BASELINE	59.0
1.	firstorder_MeanAbsoluteDeviation	59.6
2.	firstorder_Entropy	59.9
3.	firstorder_Energy	58.7
4.	glszm_SmallAreaLowGrayLevelEmphasis	58.5
5.	firstorder_90Percentile	59.3
6.	glcm_Autocorrelation	59.5
7.	firstorder_Mean	59.4
8.	firstorder_Maximum	59.1
9.	glszm_LowGrayLevelZoneEmphasis	58.9
10.	glcm_Imc1	60.0
11.	firstorder_Median	60.1
12.	glszm_ZoneVariance	59.3
13.	firstorder_TotalEnergy	58.9
14.	firstorder_10Percentile	60.4
15.	firstorder_Minimum	59.1

16.	glszm_SmallAreaHighGrayLevelEmphasis	59.9
17.	firstorder_InterquartileRange	60.0
18.	firstorder_Kurtosis	59.6
19.	firstorder_RobustMeanAbsoluteDeviation	59.3
20.	firstorder_RootMeanSquared	59.8
21.	gldm_LargeDependenceEmphasis	58.9
22.	firstorder_Variance	59.1
23.	glcm_DifferenceAverage	59.4
24.	firstorder_Uniformity	58.9
25.	firstorder_Skewness	59.0
26.	firstorder_Range	58.6
27.	glcm_JointAverage	58.7
28.	glcm_ClusterProminence	59.2
29.	glcm_ClusterTendency	58.5
30.	glcm_ClusterShade	60.2
31.	glcm_Correlation	60.0
32.	glcm_DifferenceEntropy	58.1
33.	glrlm_RunVariance	59.3
34.	glcm_JointEnergy	59.6
35.	glcm_Contrast	60.5
36.	glcm_Idm	59.3
37.	glcm_Imc2	58.7
38.	glcm_DifferenceVariance	59.2
39.	glcm_Idmn	58.0
40.	glcm_MCC	57.9
41.	glcm_JointEntropy	56.6

Table 3.5: Feature Selection

Sustainability

4.1 Environmental Aspect

4.1.1 Developement

The biggest environmental impact of this project was the required computational power to train the models. During the entire duration of the project there approximately a total of 5000 models were trained.

Out of which the most demanding part was the exhaustive sequential backward feature selection, which in itself required the training of 3000 models. The return value of this computation is close to none, as it did not increase the model performance. The only marginal benefit is the model being able to reach the same performance on 2/3rd of the original feature count. This is marginal because the omitted features belong to different feature groups, and the biggest overhead for most feature groups is calculating the voxel-based matrix volume; meaning if not all features are omitted from a feature group, it only decreases the computational requirements marginally. In conclusion, this was not worth computing.

The 3000 models trained for the features selection were trained on a wide variety of GPUs in a network cluster. These GPUs were H100, P40, RTX3060, RTX3090, RTX2080, multiple T4s (Google Collab's), multiple P100s (Kaggle's). Mining the cluster head's log, reveals that around half of these models were trained on the H100 and the other half on the rest. The H100 was consuming around 300 Watts of power while training with a speed of 5s/epoch and taking 30-60 epochs on average to stop. The H100 consumed between $1500 \cdot 0.3\text{kW} \cdot 30 \cdot 5\text{s} \div 60 \div 60 = 18.75\text{kWh}$ on the lower and $2 \cdot 18.75 = 37.5\text{kWh}$ on the upper end. The other half is harder to estimate due to the many different GPUs used in the cluster, but a crude estimate is $1500 \cdot 0.2\text{kW} \cdot 30 \cdot 8\text{s} \div 60 \div 60 = 20\text{kWh}$ on the lower and $2 \cdot 20 = 40\text{kWh}$ on the upper end. Rounding the sum of them up to 80kWh and adding an extra 30% overhead for running the rest of the components of the servers/PCs, results in an upper boundary of 104kWh consumed during the feature selection.

The rest of the 2000 models were trained on the P40 and RTX3060, with preferring the P40 and probably training 3/4th of them. Doing the same estimations for the P40 yields $1500 \cdot 0.2\text{kW} \cdot 30 \cdot 7\text{s} \div 60 \div 60 = 17.5\text{kWh}$ on the lower and $2 \cdot 17.5 = 35\text{kWh}$ on the upper end. And for the RTX3060 yields $500 \cdot 0.15\text{kW} \cdot 30 \cdot 7\text{s} \div 60 \div 60 = 4.375\text{kWh}$ on the lower and $2 \cdot 4.375 = 8.75\text{kWh}$ on the upper end. Rounding the sum of them up to 45kWh and adding an extra 30% overhead for running the rest of the components of the server, results in an upper boundary of 58.5kWh consumed during the the rest of the trainings.

Feature extraction was running on a CPU, with the entire server consuming 200 Watts during the process. The entire duration while feature extraction was running is around 2 weeks. The total power consumption of the feature extraction was $0.2\text{kW} \cdot 14\text{d} \cdot 24\text{h} = 67.2\text{kWh}$.

In summary this project had the following environmental impact, calculating with the average of 0.25€/kWh for the price of the electricity [8] and $0.2\text{kg}CO_2\text{e}$/kWh for the emission [9]:

Part	Consumption	Price	Emission
feature selection	104kWh	26€	20.8kgCO_2e
rest of the experiments	58.5kWh	14.6€	11.7kgCO_2e
feature extraction	67.2kWh	16.8€	13.4kgCO_2e
Total (+10% overhead; rounded up)	260kWh	65€	52kgCO_2e

Table 4.1: Sustainability

For reference, this number is a bit less than the average Spanish household's monthly power consumption of 324kWh. [10]

4.1.2 Production

This project, whether implemented in a production, clinical or research setting, has the potential to significantly enhance sustainability in its respective fields. Acquiring dMRI data typically requires 10-40 minutes, depending on the specific parameters, whereas obtaining a T1 image takes only about 8 minutes, making data acquisition a considerable time-saving step. Additionally, dMRI data demands extensive and labor-intensive preprocessing. **Accelerating clinical or research workflows by a factor of 5** not only saves time but also increases the capacity for data collection or patient processing by the same factor.

Synthesizing a single record of relative connectivity or diffusion FA/MD takes only a few seconds on a GPU and the radiomic feature extraction takes around 20 minutes beforehand. This translates into the approximate energy demand of 200W $\cdot$(5s $\div$60 + 20m) $\div$ 60 = 67Wh of synthetizing a record. It can be argued that the time saved with the data acquisition is lost with the time it takes to extract the radiomic features, but neither the doctor (or MRI operator) and patinet needs to be present during the computation of the feature extraction. Assuming an MRI machine has a consumption of 25-70kW [11] and calculating with an average of 50kW, and an average of 25m for the data acquisition of a dMRI record, adds up to the power requirement of 50kW $\cdot$25m $\div$60 = 21kWh per record. Adding up the total energy requirements of T1 data acquisition and record synthesis is 50kW $\cdot$8m $\div$60 + 67Wh $\div$1000 = 7kWh per record. This means that **this method could be 3 times more sustainable than the traditional approach**. And it would return the energy consumption of the development phase in the data acquisition of 260 $\div$ (21 $-$ 7) = 19 records, this is naturally a miniscule number compared to how many MRI records are being done in the world.

4.2 Economic Aspect

4.2.1 Cost

The electricity cost during the development phase amounted to 65€. Approximately half of this cost was covered by colleagues and their networks, who contributed computational resources during the feature selection process. Additional support came from free services provided by Google and Kaggle.

Realistically, billing this project would consist of three components: the cost of labor, the price of the server used for development, and the electricity consumed. With the electricity cost already detailed, the server cost and labor are interconnected. The project could have been executed on

a less powerful machine, though this would have required more time and more careful focus on software design. Throughout the development, the primary limiting factors were RAM and disk storage. Significant effort was invested in developing efficient data structures and optimizing the memory footprint to address these constraints.

In conclusion, the minimum hardware which this project could be feasibly executed on is 32GBs of RAM, Intel Core i5-12600K (or similar), Nvidia RTX3060 (or similar), and 256GBs of storage. The current cost (new parts ordered from Amazon) of such server (with 'cheap' consumer grade hardware) would be around 300€ for the GPU, 150€ for the CPU, 50€ for the RAM, 25€ for the SSD, and 150€ for the motherboard, adding up to 675€. The main limiting factors in this configuration are the RAM, which was utilized up to 128GBs during development; and disk storage, which was utilized up to 300GBs. The extra RAM and storage would cost an additional 200€ and 25€ totalling at a new sum of 900€.

The development of this project required approximately 500 hours of work. Calculating with the average hourly rate of a software engineer in Spain [12] of 19€, leads to a total labor cost of 19€ ·500h = 9500€. Therefore, the total billing for the project would be:

Part	**Price**
Cumulative Salary	9500€
Server Components	900€
Electricity	65€
Total	10465€

Table 4.2: Billing

4.2.2 Return

This project by all means, is a proof of concept, checking the viability of this approach. More information on the exact conclusions are in chapter 5, but the potential return value of this project is huge.

As stated in the previous subsection 4.1.2, this project has the potential to accelerate the clinical and/or research workflows by a factor of 5. The economic implications are applicable for both clinical and research workflows. In the clinical workflow, **this could increase the number of patients processed by a factor of 5**, with minimal extra cost. Especially compared to the alternative of buying 4 additional MRI machines, and hiring operators for them. In a research workflow it probably has less of an economic impact, but more of a logistical impact. As it could simplify the data acquisition, thus **allowing researchers to have up to 5 times more data acquired** with minimal extra resources invested. But even more importantly it opens the door of processing past anatomical MRI records, as they are much more common than dMRI records, virtually **increasing the available data for researchers by several magnitudes** (depending on the exact application of course).

4.3 Social Aspect

4.3.1 Development and Collaborations

The development of the project mainly required collaboration from Estela Camara Mancha (external thesis supervisor and project originator) and Alfredo Vellido Alcacena (internal thesis supervisor). Estela being the project originator, this project demanded her time the most, in the neighborhood of 40-50 hours TODO. Alfredo mostly helping with the formalities and miscellaneous nuances, had a demand of 10-15 hours TODO of his time. Additionally Estella's colleague Vasiliki Bikou also spent a good chunk of her time on catching me up to speed with the contextual information of the project. And lastly my good friends Botond Lovasz, Andris Gyori and Daniel Csepregi-Horvath donated computational power, and required a few hours of their time to set up the worker nodes on their hardware. I especially want to thank Botond for using his connections and giving me access to the H100 state of the art tensor core GPU.

4.3.2 Inclusivity

The most obvious consideration from age, gender, sex, and cultural diversity; is sex, as males and females have slightly different brain structures [13]. For the used control records, there are 17 male and 15 female records, which is a relatively even split; and for the patient records, there are 25 male and 13 female records, which is a bit unbalanced. This theoretically could negatively impact the under represented group if there are truly any differences between the two sexes that matter from the model's perspective. However in the case of the project due to the very limited number of records, it was not an option to discard 1/3rd of the patients, to have the sexes completely balanced. And it would also need further experimentation to determine if this truly impacts this project or not.

This potential difference was overlooked during most of the project's lifetime, but it should have been included as another constant ratio to be kept during the train/validation/test splitting, which was covered in subsection 2.2.1. Due to limited time and resources, the models will not be re-trained with this new rule in mind, and this is something that will be included in the Future Improvements 5.1 subsection.

The rest of the considerations are less pressing and hard to take into account, like the brain structure of different ethnicities is an under researched area, plus this dataset does not contain any information regarding this aspect. The only other consideration which can be related is age, but this is partially accounted for as part of the constant symptomatic/asymptomatic ratio. Because being symptomatic is closely related to the CAP score (covered in subsection 2.1.8), which is directly related to age. Thus symptomatic/asymptomatic is indirectly related to age.

4.4 Risks

Environmental, Economic and Social risks are not really applicable to this project, as it is a foundational proof of concept research project.

Conclusions

5.1 Future Improvements

Further investigating the sex imbalance issue discovered in subsection 4.3.2, reveals that with the used seed for the experimentation is not terribly imbalanced for the most part, but there are definitely room for improvement by enforcing a constant ratio. The male/female ratio for the control record splits are 0.49/0.67/0.67 (train/validation/test), and for the patient records are 0.62/0.5/1.

Sources of Information

[1] José L Lanciego, Natasha Luquin, and José Obeso. "Functional neuroanatomy of the basal ganglia". In: *Cold Spring Harbor perspectives in medicine* (2012). URL: https://doi.org/10.1101/cshperspect.a009621.

[2] Olivia C Matz and Muhammad Spocter. "The Effect of Huntington's Disease on the Basal Nuclei". In: *Cureus* (2022). URL: https://doi.org/10.7759/cureus.24473.

[3] Hyungyou Park et al. "Aberrant cortico-striatal white matter connectivity and associated subregional microstructure of the striatum in obsessive-compulsive disorder". In: *Molecular Psychiatry* (2022). URL: https://doi.org/10.1038/s41380-022-01588-6.

[4] Marius E Mayerhoefer et al. "Introduction to radiomics". In: *Journal of Nuclear Medicine* 61.4 (2020), pp. 488–495. URL: https://jnm.snmjournals.org/content/jnumed/61/4/488.full.pdf.

[5] Loïc Duron et al. "Gray-level discretization impacts reproducible MRI radiomics texture features". In: *PLoS One* (2019). URL: https://doi.org/10.1371/journal.pone.0213459.

[6] *Special Characteristics of HD Data*. 2022. URL: https://enroll-hd.org/for-researchers/analyzing-data/special-characteristics-of-hd-data-2.

[7] Dylan Trundell et al. "Defining Clinically Meaningful Change on the Composite Unified Huntington's Disease Rating Scale". In: *Neurology* 92.15_supplement (2019), P1.8–043. URL: https://doi.org/10.1212/WNL.92.15_supplement.P1.8-043.

[8] *Spain - Household electricity prices*. URL: https://countryeconomy.com/energy-and-environment/electricity-price-household/spain (visited on 12/28/2024).

[9] B.W. Ang and Bin Su. "Carbon emission intensity in electricity production: A global analysis". In: *Energy Policy* 94 (2016), pp. 56–63. ISSN: 0301-4215. URL: https://doi.org/10.1016/j.enpol.2016.03.038.

[10] *Electricity Consumption per Dwelling*. URL: https://www.odyssee-mure.eu/publications/efficiency-by-sector/households/electricity-consumption-dwelling.html (visited on 12/28/2024).

[11] *MRI and Sustainability*. URL: https://www.siemens-healthineers.com/perspectives/MRI-reducing-energy-consumption (visited on 12/28/2024).

[12] *Average Software Engineering Salaries by Country*. URL: https://codesubmit.io/blog/software-engineer-salary-by-country (visited on 12/28/2024).

[13] Daphna Joel. "Male or Female? Brains are Intersex". In: *Frontiers in Integrative Neuroscience* 5 (2011). ISSN: 1662-5145. URL: https://doi.org/10.3389/fnint.2011.00057.

Software Design

The software design was mainly guided by the provided format of the raw data. Which was from a few different sources, as different people were working with different data at the Hospital, and they did not have a unified collection. Thus the following documentation of the software design will be structured going from the raw data, to the preprocessed data, and then the model itself.

A.1 Raw Data

The raw data is scattered amongst many files, are registered in different spaces, does not have masks applied, along with other challenges detailed in Section 2.1.

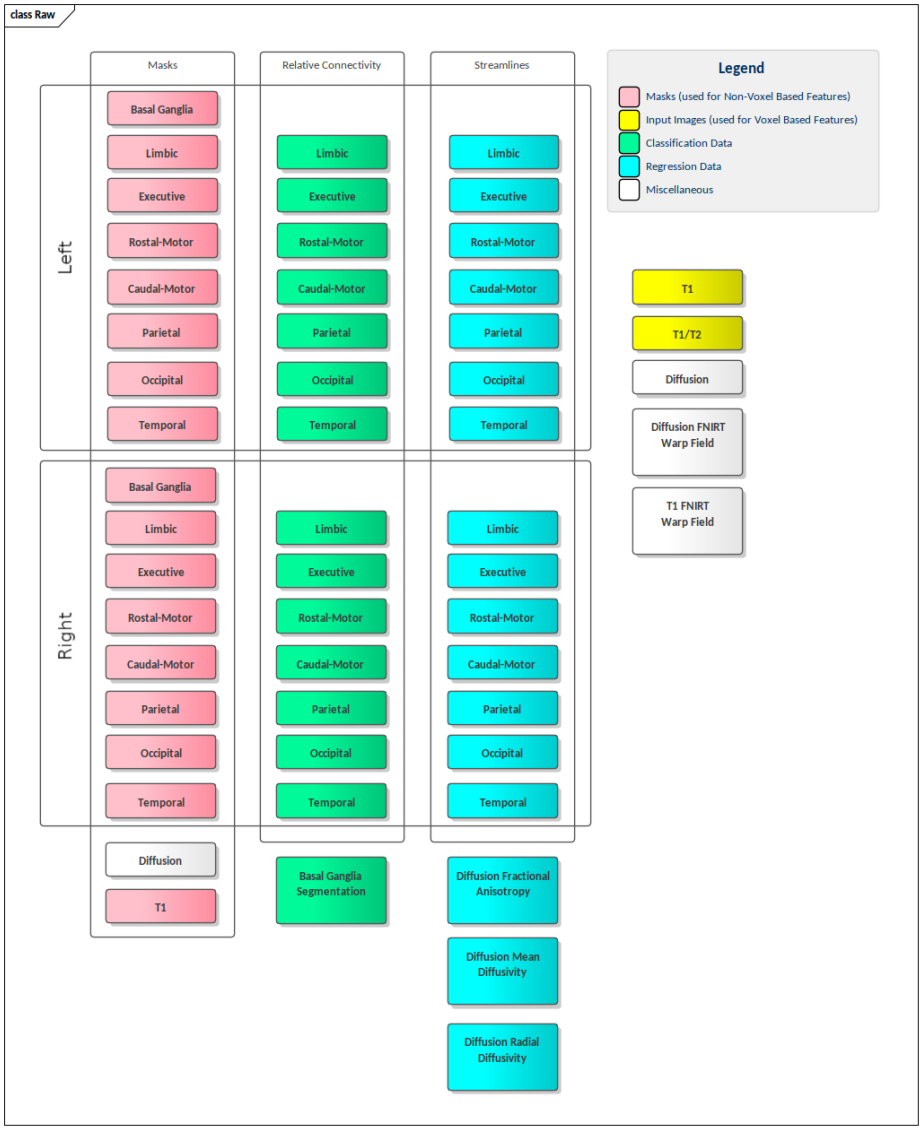

Figure A.1: Files: Raw

The following class diagram documentations are detailed abstractions of the actual source code, as Unified Modeling Language (UML) is not completely Python compliant. Before moving on, the following simple data types are used in the UML diagrams:

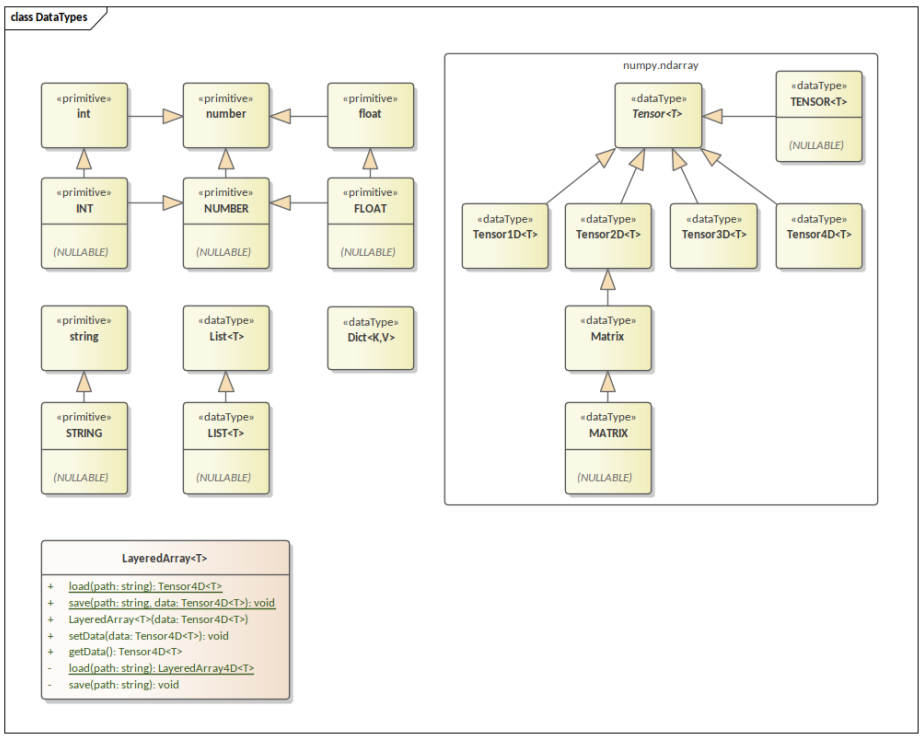

Figure A.2: Class Diagram: Data Types

There is one odd one out in this diagram, that being the LayeredArray, which is more than a simple data type or primitive. This class is a simple space efficient data structure of a 4D tensor.

Storing some of the data provided can be highly inefficient by storing the raw tensor. Ultimately the design aims to collect the scattered data from multiple files, into clean logical groups, for example all the cortical target masks into a single 4D tensor where the 4th dimension is reserved for the multiple targets. But this poses a greatly inefficient way of storing the data, as each target region only takes up a small portion of the entire space. The idea behind the LayeredArray class, it stores the 4D tensor as a list of 3D tensors, each cropped down to their effective space where there are non-zero voxels. This also requires an additional list of 3D vectors to store each layer's origin, and where to paste it in the original space.

This solution offers a very efficient way of storing data for this use case, as the raw storage solution for the cortical targets were around 50MBs, and this cuts it down to 3MBs. This is even more drastic for the relative connectivity, which is more than a simple boolean mask and proportionally takes up even less space of the entire brain; in which's case it was cut down from 110MBs to 0.5MBs, reducing the disk requirement by several magnitudes.

The original NIfTI format also does a very good job at storing data efficiently, but this solution provides a more lower level control over the way of storing the data. This is beneficial as the data are stored in numpy format, making it easier to ignore certain data type safety checks, data type conversions, and leaving behind the NIfTI format's additional complexity of the orientation, transformation, and many more nuances that are part of the NIfTI header.

This has the undoubted drawback of not being able to use FMRIB software library (FSL) tools natively on our datatypes, such as fsleyes for simply viewing a record. But thanks to the

opensource nature of the FSL suite, with a few additional lines of code, support can be added for our datatypes (included in Appendix **??**).

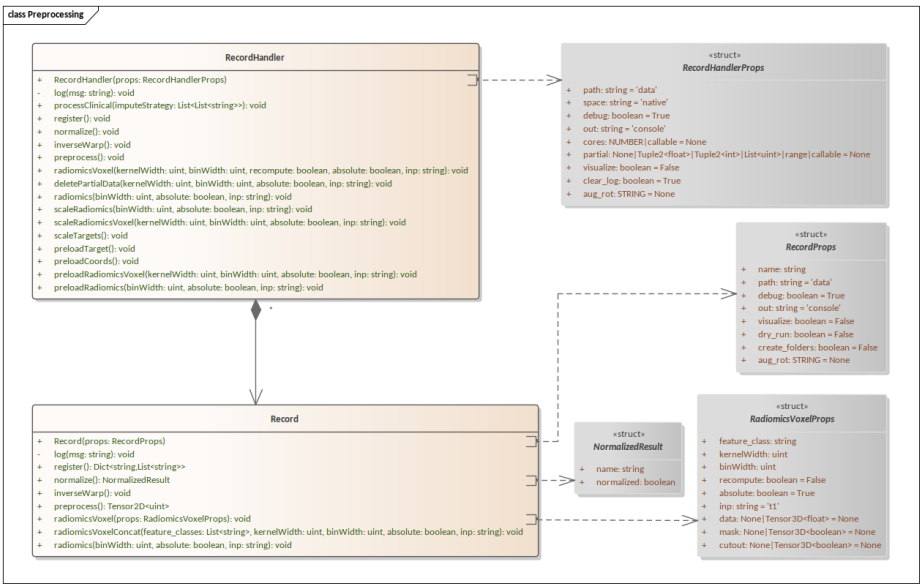

Figure A.3: Class Diagram: Preprocessing

The class diagram above contains the two classes responsible for the preprocessing of the data. The RecordHandler being the controller itself that handles the high level operations on a collection of records. And the records handling the low level computations on the data itself.

A.2 Common Functions

There are set of static common functions, which give the low level backbone of the entire project. They are grouped into two categories, util and visual. Where the prior one contains everything from simple data type castings, external FSL library system calls, to computing radiomics, and more. And the latter one is a collection of functions for visualizing data.

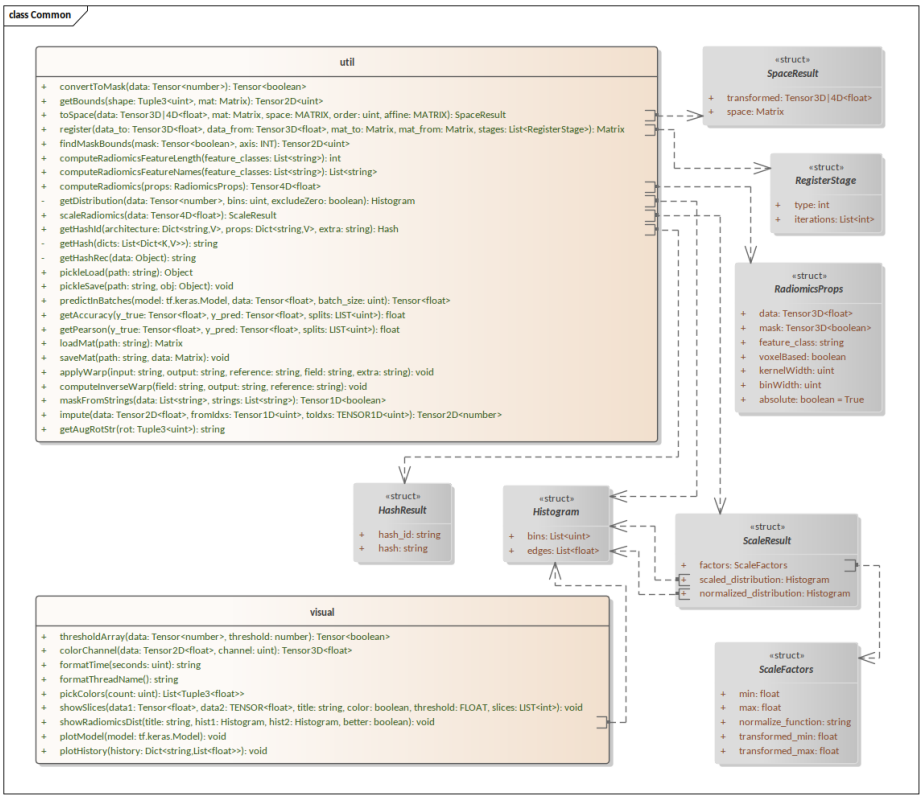

Figure A.4: Class Diagram: Common

A.3 Preprocessed Data

Preprocessing the data are composed from the next few high level operations done by the RecordHandler:

1. Register some records (not needed for most of them)
2. Compute all normalized records
3. Compute the inverse FNIRT warp field (they weren't provided)
4. Convert native records into our numpy format (compute the affine transformations, merge different sources, etc.)
5. Compute scaling factors across all native records
6. Preload native records
7. Convert normalized records into our numpy format
8. Compute scaling factors across all normalized records
9. Preload normalized records
10. Construct normalized coordinate maps
11. Warp normalized coordinate maps into native space

12. Scale and preload coordinate maps
13. Impute clinical data

After preprocessing, the next set of logical grouping and files are left:

Figure A.5: Files: Preprocessed

However, this project focuses on only the voxels inside the Basal Ganglia, meaning the model on a datapoint level is never going to operate outside of that region. Thus, the voxels of the ROI can be 'preloaded' so the DataGenerator class, used for feeding the model data, can very efficiently only load the datapoints that it needs. Furthermore this can also be applied when computing the radiomic features, as it only needs to be computed for the ROI.