Title
Analyzing heterogeneity and complexity of white matter using deep learning

Permalink
https://escholarship.org/uc/item/1q253116

Author
Sharan, Richika

Publication Date
2019

Peer reviewed|Thesis/dissertation
Analyzing heterogeneity and complexity of white matter using deep learning

A thesis submitted in partial satisfaction of the requirements for the degree
Master of Science in Computer Science

by

Richika Sharan

Committee in charge:
Professor Ambuj Singh, Chair
Professor Scott Grafton
Professor Xifeng Yan

September 2019
The Thesis of Richika Sharan is approved.

Professor Scott Grafton

Professor Xifeng Yan

Professor Ambuj Singh, Committee Chair

June 2019
Abstract

Analyzing heterogeneity and complexity of white matter using deep learning

by

Richika Sharan

In this work, we identify the heterogeneity of regions of white matter across individuals and the complexity of different regions of white matter using diffusion MRI data. We analyze the heterogeneity of a region across individuals by computing the pairwise difference between voxels. We review various complexity measures like entropy and Kolmogorov complexity. Using autoencoders, we analyze the inherent dimensionality of regions of white matter structure as a means for measuring complexity. The intrinsic complexity of a region can be determined by how effectively it can be predicted given its neighborhood. We pose this as the problem of inpainting a three dimensional region of the brain given its context with deep generative modeling using Generative Adversarial Networks (GANs) conditioned on the neighborhood. The discriminator is trained on differentiating between real and generated patches while the generator is simultaneously trained to adversarially fool the discriminator and minimize the voxel-wise reconstruction loss between the actual and generated patches. We study the comparative performance of a well-trained GAN across regions of the brain in multiple subjects and identify particularly challenging or complex regions of brain wiring. Finally, we analyze the correlation between these measures of heterogeneity and complexity.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>iii</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Human Connectome Project Dataset</td>
<td>3</td>
</tr>
<tr>
<td>2 Heterogeneity</td>
<td>7</td>
</tr>
<tr>
<td>2.1 Method</td>
<td>7</td>
</tr>
<tr>
<td>2.2 Results</td>
<td>8</td>
</tr>
<tr>
<td>3 Complexity</td>
<td>10</td>
</tr>
<tr>
<td>3.1 Related work</td>
<td>10</td>
</tr>
<tr>
<td>3.2 Complexity measures</td>
<td>11</td>
</tr>
<tr>
<td>4 Analyzing dimensionality of white matter using autoencoders</td>
<td>14</td>
</tr>
<tr>
<td>4.1 Autoencoders - Introduction</td>
<td>14</td>
</tr>
<tr>
<td>4.2 Related work</td>
<td>16</td>
</tr>
<tr>
<td>4.3 Using autoencoder to measure complexity</td>
<td>17</td>
</tr>
<tr>
<td>4.4 Results</td>
<td>19</td>
</tr>
<tr>
<td>4.5 Autoencoder drawbacks</td>
<td>20</td>
</tr>
<tr>
<td>5 Generative modeling of white matter regions</td>
<td>23</td>
</tr>
<tr>
<td>5.1 Generative Adversarial Network - Introduction</td>
<td>25</td>
</tr>
<tr>
<td>5.2 Related work</td>
<td>27</td>
</tr>
<tr>
<td>5.3 Using GANs to measure complexity</td>
<td>29</td>
</tr>
<tr>
<td>5.4 GAN architecture</td>
<td>30</td>
</tr>
<tr>
<td>5.5 Training details</td>
<td>35</td>
</tr>
<tr>
<td>5.6 Results</td>
<td>35</td>
</tr>
<tr>
<td>6 Correlation between measures</td>
<td>41</td>
</tr>
<tr>
<td>7 Conclusion</td>
<td>45</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

White matter is the tissue in the brain composed of nerve fibers that connect the nerve cells present in the grey matter and carries the nerve impulses between neurons. Analyzing white matter enables better understanding of morphological features of the brain across population of subjects which results in effective modeling of white matter. Modeling of white matter has various applications, some of which are described below.

Modeling the normal range of neuroanatomical variability in a healthy population can be useful for anomaly detection. Patients with white matter structures which lie outside the normal range can be identified as outliers. Anomaly detection is also useful for identifying patterns which are indicative of a particular disease.

Medical imaging is an important tool for clinical diagnosis and scientific research. However, the quality of medical scans can be affected by factors like artifacts or positioning outside the field of view resulting in noise which leads to lost information. The missing information can be reconstructed synthetically using post processing methods. Modeling of white matter structures can be useful for denoising of scans since we learn the inherent behavior of white matter structures and can thus reconstruct the missing regions. Medical image analysis using machine learning requires a large number of sub-
objects for sufficient generalization. While data augmentation (e.g., translation, rotation) is useful to alleviate this problem to some extent, it leads to the training data being correlated. Thus, generative modeling of white matter can lead to generating synthetic medical scans which is useful for dataset augmentation.

White matter lesions are tissues that don’t look like normal brain tissue. The impact of white matter lesions on normal brain functioning is not well understood and the resulting brain dysfunction due to lesions varies across the population. Lesion reconstruction is the method of reconstructing the white matter lesion region to map their connectivity to other regions of the brain. This is helpful for quantification of the neuroanatomical connectivity changes leading to better prediction of the resulting brain dysfunction. Generative modeling of white matter can be useful for lesion reconstruction.

In this work, we analyze the heterogeneity of regions of white matter across individuals and the complexity of different regions of white matter. Heterogeneity is the measure of variance of a region of white matter across the population. Complexity is the measure of neuronal connections in a region of white matter. We analyze a region of white matter in terms of heterogeneity and complexity to understand the neuronal connections crossing the region and how the connections vary across the population. Quantifying heterogeneity and complexity is useful for effective modeling of white matter and also helps in determining error bounds and uncertainty associated with the modeling.

The next section describes the Human Connectome Project Dataset that we use for this work. Chapter 2 mentions how we measure heterogeneity of a region of white matter across individuals and the corresponding results on different regions of the brain. Chapter 3 describes complexity of a region of white matter and the related work that we use as a baseline for complexity measure. It also mentions different complexity measures that have been studied in literature like entropy, spatial information and Kolmogorov complexity. Chapter 4 introduces autoencoders and delineates analyzing dimensionality
of white matter using autoencoders as a means to measure complexity. We compute this measure of complexity across different regions of the brain and identify regions with high complexity. Chapter 5 presents generative modeling of white matter regions using Generative Adversarial Networks which is also used as a complexity measure in terms of inpainting a region of white matter conditioned on its neighborhood. We study the comparative performance of a well-trained Generative Adversarial Network on different white matter regions to identify particularly challenging regions of brain wiring. Chapter 6 describes our study on the correlation between the measures of heterogeneity and complexity. We then conclude and mention future work in Chapter 7 and 8 respectively.

1.1 Human Connectome Project Dataset

1.1.1 Description

Diffusion-weighted magnetic resonance imaging (dMRI) measures the random motion of water molecules within a brain tissue. dMRI is useful for imaging structural connectivity in the brain. The data we use was collected as part of the Washington University-Minnesota Consortium Human Connectome Project (Essen et al. 2013). Participants were recruited from Washington University (St. Louis, MO) and the surrounding area and gave informed consent. The data is derived from the 630 healthy participants (358 female, 272 male). Diffusion volumes, known as voxels, were collected with a spatial resolution of $1.25 \times 1.25 \times 1.25 \text{ mm}^3$ resulting in the brain being bound into a $146 \times 174 \times 146$ grid.
1.1.2 Data Processing

The preprocessing pipeline used for in this work was identical to that used in Volz et al. [2018]. A 6D orientation distribution function (ODF) \( \Psi(\theta) \) was calculated on a set of 642 approximately-evenly spaced directions \( \theta \in \Theta \) on a tessellated icosahedron. The ODF magnitudes were rescaled such that the sum of each ODF is \( \sum_{\theta \in \Theta} \Psi(\theta) = 1 \).

Multi-dimensional anisotropy (MDA) value for each direction \( \theta \) was calculated using

\[
MDA(\theta) = \frac{1 - \mu}{\sqrt{1 + 2\mu^2}}
\]  

(1.1)

where

\[
\mu = \left( \frac{\Psi(\theta)}{\Psi(\theta_{\text{min}})} \right)^{2/3}
\]  

(1.2)

where \( \theta_{\text{min}} \) is the direction with the smallest ODF magnitude.

MDA values were calculated for the four largest local maxima in every ODF resulting in values denoted by MDA0, MDA1, MDA2, and MDA3 in decreasing size. The four
corresponding directions \( \theta_0, \theta_1, \theta_2, \theta_3 \) were also extracted and saved as 3D vector fields for each of \( \theta_0 \ldots \theta_3 \). Prior work on the same data found that ODF peaks become very noisy after the 4th direction (Volz et al. [2018]).

The MDA vectors represent the most probable underlying oriented white matter microstructure within a voxel. It is also worthwhile to note that MDA vectors are a directed representation of an ODF which are fundamentally undirected. We account for this in our work.

We only use MDA0 (which has the largest magnitude) as it has the highest signal-to-noise ratio. We experimented with using additional MDA peaks but that did not give better results when compared to only using the first peak. This could possibly be due to additional orientations being associated with noise since they are in white matter regions where it is unlikely there are multidirectional crossing fibers. This is also supported by a study on the same dataset by Volz et al. [2018] reporting that 32.9\% of white matter are unidirectional.

Thus, the dataset is pre-processed such that every scan is represented in a 3D space where every point corresponds to a voxel represented by a 3D vector. For training the machine learning models, we split the dataset into training (70\%, 442 subjects), validation (15\%, 94 subjects) and testing (15\%, 94 subjects).
Figure 1.2: The plot has heterogeneity on one axis and complexity on the other. Each subplot shows a two dimensional slice of a three dimensional white matter region from two different subjects. The subplots show the first MDA vector with the $x$ and $y$ vectors represented by the arrows and $z$ vector represented by the color. The shadowed reflection of the arrows represents the symmetric property of the MDA vectors. Going from left to right, (a) Bottom left plot shows a region with low heterogeneity and low complexity since the vectors are unidirectional and consistent across the two subjects. (b) Top left shows a region with high heterogeneity and low complexity where most of the vectors are aligned in the same direction but the vectors are different across the two subjects. (c) Bottom right shows a region with low heterogeneity and high complexity since the vectors are in different directions but are still consistent across the two subjects. (d) Top right shows a region with high heterogeneity and high complexity where the vectors are aligned in different directions and also varies across the two subjects.
Chapter 2

Heterogeneity

Heterogeneity of a region of white matter is the measure of variance in that region across the population. White matter regions can have the same or different type of wiring in subjects. While modeling white matter, it is important to also model the variability in regions across the population. Quantifying heterogeneity gives a measure of uncertainty associated in the modeling of a particular region.

2.1 Method

We measure heterogeneity of a region of white matter across individuals by computing the average pairwise difference between the voxels in that region. In particular, we take the L2 norm of the difference between the 3D vector representing a voxel in one subject with the 3D vector representing the same voxel in another subject.

Let $X_v$ be the 3D vector in voxel $v$ for subject $X$ and $Y_v$ be the corresponding 3D vector in voxel $v$ for subject $Y$. Dissimilarity in voxel $v$ between subject $X$ and subject
$Y$ is computed as,

$$D(X, Y, v) = \min(\|X_v - Y_v\|_2, \|X_v + Y_v\|_2)$$  \hspace{1cm} (2.1)$$

Since we extract the directed vector representation of an ODF which is fundamentally undirected, we account for this in the above formulation by computing the difference between a MDA vector or its reflection around the origin to another MDA vector or its reflection around the origin, where the origin is the center of a given ODF.

We average $D(X, Y, v)$ across all voxels in a region across multiple subjects to find the average voxel-wise difference in a region. To normalize across different regions, we divide the average voxel-wise difference in a region with the average vector magnitude in that region. The resultant is the measure for heterogeneity of a region.

### 2.2 Results

The heterogeneity measure described above is computed for different regions of the brain across multiple subjects and we observe that the values vary across the regions. Regions like the corpus callosum, which is mostly unidirectional, have low values. On the other hand, we observe higher values near the boundary of white matter and grey matter. This could be due to more leeway for differences in wiring because of differently directed crisscrossing fibers in this region or because the normalization process affects the voxels in this region the most due to small vector magnitudes. Figure 2.1 shows the heterogeneity measure across different regions of the brain.
Figure 2.1: The plot shows the axial, sagittal and coronal cross sections of the brain from a subject. The MDA0 magnitudes from the diffusion MRI are plotted in the background and the normalized voxel-wise difference of a region across the population is superimposed on top. The colormap is from black representing 0 value to red to yellow being the highest value. Thus, regions in yellow have the highest normalized voxel-wise difference. We observe lower values in the corpus callosum and higher values near the boundary of white matter and grey matter, which is apparent in the middle picture.
Chapter 3

Complexity

Complexity of a region of white matter is the measure of neuronal connections in that region. Regions can have a single bundle of unidirectional neurons, like in the corpus callosum, or multiple differently directed crisscrossing neurons. Analyzing the inherent complexity of white matter is useful for better understanding and subsequent modeling of white matter. However, measuring the complexity of a region is not as straightforward. Lloyd [2002] provides a list of complexity measures which can be grouped under three questions: how hard is it to describe?, how hard is it to create?, and what is its degree of organization? We describe some complexity measures in this chapter which answer these questions.

3.1 Related work

Volz et al. [2018] used 630 diffusion scans from the Human Connectome Project to infer the number of fiber orientations per voxel in a model-free fashion. They introduce a probabilistic white matter atlas delineating the number of trackable diffusion directions in each voxel. MDA values (MDA0-MDA3) for a voxel are extracted from the four largest
peaks in every ODF. They considered an ODF peak trackable if its corresponding MDA measurement was greater than a fixed threshold (0.01). If the MDA0 value of a voxel is above the 0.01 threshold in at least 50% of the subjects, it has fiber orientations in at least one direction. Similarly, if the MDA1 value of the same voxel is above the 0.01 threshold in at least 50% of the subjects, it has fiber orientations in at least two directions. This is repeated for all the MDA values (MDA0-MDA3) to obtain a deterministic trackable directions mask (dTDM) which labels each voxel with the number of distinguishable orientations ranging from 0 to 4.

We use this as a baseline for complexity measure with more number of distinguishable fiber orientations in a region implying that the region is more complex and vice versa.

3.2 Complexity measures

Different measures have been proposed in various fields for estimating the complexity of an image. We describe some of these measures below.

3.2.1 Entropy

Shannon entropy is a measure of information in data such as a random variable. It is defined as

$$S = - \sum_i P_i \log P_i$$  \hspace{1cm} (3.1)

As $- \log P_i$ represents the amount of information conveyed by an event, entropy gives the average information or uncertainty of a random variable. When a low probability event occurs, it carries more information than a high probability event. Entropy answers the how hard is to describe? question as a measure of complexity. However, it isn’t a good measure of image complexity as it does not take into consideration the spatial distribution
of pixels. For instance, Figure 3.1 shows two binary images with the same entropy of 1 but different image complexities ((b) has more randomness than (a)). Research in the field of image complexity alleviates this problem by image partitioning but it does not apply well to our problem since the values in our data are not as discrete, unlike grayscale images.

### 3.2.2 Spatial information

Spatial information is an indicator of edge energy and also answers the *how hard is to describe question?* as a measure of image complexity. Separate horizontal and vertical Sobel filters are applied to the image to find the horizontal and vertical edges respectively, then the total edge energy or spatial information at a pixel is computed as the Euclidean distance.

\[
SI = \sqrt{s_h^2 + s_v^2}
\]  

(3.2)
Spatial information works well as a measure of complexity for natural images with the added advantage of being robust to compression. However, it does not work well for our data as we do not have such a well defined concept of edges in the brain.

### 3.2.3 Kolmogorov complexity

Kolmogorov complexity defines the complexity of an object to be the length of the shortest computer program that can reconstruct the object. It answers the *how hard is it to create?* question as a measure of complexity. Data which is more random (e.g. 4rhqf7b4dsa) is harder to create and thus more complex than more ordered data (e.g. abababab).

However, Kolmogorov complexity is not computable. It is approximated using compression techniques. We discuss one such compression technique using autoencoders in the next chapter.
Chapter 4

Analyzing dimensionality of white matter using autoencoders

Based on Kolmogorov complexity, the inherent complexity of data can be estimated from how efficiently it can be compressed and then reconstructed. Thus, we can measure complexity by embedding data into a low dimensional space and then evaluating how efficiently it can be reconstructed. We achieve this objective with autoencoders.

4.1 Autoencoders - Introduction

Autoencoders are an unsupervised learning technique used for representation learning i.e. learning representations of data which extract meaningful features. Autoencoders learn a direct encoding from inputs to their representations. Let $f_\theta$ be a feature extracting parameterized function known as the encoder which transforms the input $x$ to a feature vector $h$. Another parameterized function $g_\theta$ known as the decoder maps the feature
vector $h$ back into the input space producing a reconstruction of the original input $\hat{x}$.

\begin{align}
h &= f_\theta(x) \tag{4.1} \\
\hat{x} &= g_\theta(h) \tag{4.2}
\end{align}

Equation 4.1 represents the encoder and Equation 4.2 represents the decoder. The parameters of the encoder and decoder are learned together while learning an accurate reconstruction of the input such that the reconstruction error $L(x, \hat{x})$ is minimized over training data. To capture the structure of the input distribution, some constraint has to be imposed such that the autoencoder does not learn the identity function which has zero reconstruction error for all inputs. This is achieved using different forms of regularization.

One form of regularization is attained by constraining the feature space to be lower dimensional than the input space. If $x \in \mathbb{R}^n$ and $h \in \mathbb{R}^m$, we impose $n > m$ leading to embedding in a lower dimensional space. Thus, a bottleneck is imposed in the network which forces a compressed knowledge representation of the original input. The parameters $\theta$ are learnt such that the reconstruction error is minimized.

\[ J(\theta) = \min \sum_t L(x^t, g_\theta(f_\theta(x^t))) \tag{4.3} \]

where $x^t$ is the training sample.

This minimization is done using optimizers such as stochastic gradient descent. The functions $f_\theta$ and $g_\theta$ are

\begin{align}
f_\theta(x) &= a_f(W_1 x + b_1) \tag{4.4} \\
g_\theta(h) &= a_g(W_2 h + b_2) \tag{4.5}
\end{align}
Figure 4.1: Autoencoder network has a bottleneck hidden layer which forces a compressed knowledge representation of the input. \cite{Jordan2018}

where $a_f$ and $a_g$ are the activation functions (identity function, sigmoid, hyperbolic tangent etc.), $W_1$ and $W_2$ are the weight matrices, and $b_1$ and $b_2$ are the bias vectors. The set of parameter $\theta$ consists of $\{W_1, W_2, b_1, b_2\}$.

If the activation functions are linear, the dimensionality reduction would be similar to Principal Component Analysis. Using non-linear activations helps the autoencoder in learning non-linear manifolds. An example of this is depicted in \ref{fig:4.2}.

\section{4.2 Related work}

\cite{Pinaya2018} use Human Connectome Project (\cite{Essen2013}) data from
healthy individuals to model neuroanatomical deviations between the input data and the reconstructed data using autoencoders. They use the trained autoencoder to measure the total and regional neuroanatomical alteration in patients with schizophrenia and autism spectrum disorder. They report that the model was able to find distinct values of total neuroanatomical deviation for each disease relative to their healthy control group. The model also identifies patterns of neuroanatomical deviations for the two diseases.

We use autoencoders to measure the neuronal connections in white matter regions and to identify complex regions of brain wiring.

### 4.3 Using autoencoder to measure complexity

In the autoencoder framework, we have an input $X$, is a region of white matter, that passes through the feature space to output $\hat{X}$, which is a reconstruction of the original input $X$. If the features of the input data were independent of each other, the compression
and subsequent reconstruction would not be efficient. If some sort of structure exists in
the input data in the form of correlations between the input features, this structure can
be learnt by the autoencoder.

We use **convolutional autoencoders** consisting of 3D convolutional filters that con-
serve the 3D spatial structure of the input. The convolutional filters are convoluted with
the input map to produce an output map followed by a non-linear activation function.
The encoder consists of layers composed of $n$ (hyperparameter) 3D convolutional filters
followed by a MaxPooling operation which downsizes the input to a layer. This is fol-
lowed by the decoder composed of the layers in the encoder in reverse and UpSampling
(instead of MaxPooling) which increases the size of the input to a layer. The autoencoder
is trained with a custom mean-squared reconstruction error accounting for symmetricity
of MDA vectors denoted by

$$L(X, \hat{X}) = \min(\| X - \hat{X} \|^2, \| X + \hat{X} \|^2)$$  \hspace{1cm} (4.6)

We measure the comparative performance of the trained autoencoder across different
white matter regions with higher reconstruction error signifying higher complexity of the
region. To normalize across different regions, we divide the reconstruction error in a
region with the average vector magnitude in that region.

### 4.3.1 Training details

We randomly sample $8 \times 8 \times 8$ regions of the brain containing mostly white matter
from the 442 training subjects. Autoencoder downsizes the input to half the original size
i.e. the bottleneck layer has half the size of the input layer. The full model architecture
is shown in Figure [4.3] The code is written in Keras with Tensorflow backend. We use
ReLU (Rectified Linear Unit) activation function in all the layers apart from the last
layer which uses tanh activation. The convolutional filters are (3,3,3) in size and the MaxPooling and UpSampling size is (2,2,2). We use ADAM optimizer (Kingma and Ba, 2014) with a learning rate of 0.0005. We use early stopping to prevent over-fitting. The training is done on 1 GeForce GTX TITAN GPU.

4.4 Results

Figure 4.4 shows examples of original and decoded images from the validation set. We observe that the decoded images are blurry (Figure 4.4 (a) and (b)) and not very accurate (Figure 4.4 (c) and (d)) at times.

The average reconstruction error is computed for a region of the brain across the subjects in the validation set and we compute this for multiple regions of the brain and observe that the values differ across the regions. Figure 4.5 shows the normalized reconstruction error across different regions of the brain. We observe that the error is higher in the cortex but also in deep white matter regions with high bending. The two symmetrical pockets in the axial view correspond to the *crossing pockets* area that Volz et al. (2018) mentions. The dTDM mask by Volz et al. (2018) labelled this region with a
value of three i.e. the number of distinguishable orientations is three. Thus, we observe
that regions with differently directed fiber orientations have higher reconstruction error
justifying our intuition that such areas have higher complexity.

4.5 Autoencoder drawbacks

We observe that the reconstructed regions are blurry. This is because mean squared
error prefers blurry output over highly accurate textures. This is possibly because with
mean squared loss, it is easier for the model to predict the mean of the distribution which
minimizes the mean voxel-wise error than learn the nuances of the input distribution.
One of the ways to alleviate this problem is to use adversarial loss.

We use vanilla autoencoders which analyze existing data to discover structure within
the data that can be leveraged to produce a compressed representation of the input.
However, this type of autoencoders cannot be used for generative modeling of data i.e.
they cannot be used to generate new data. Variational autoencoders (VAE) and Gener-
ative Adversarial Networks (GANs) are better suited for generative modeling of data. It
has been empirically shown that GANs synthesize more realistic images than VAE. We
discuss GANs in the next chapter.
Figure 4.4: The images are 2D slices of magnitude only MDA vectors from 3D white matter regions. The left image in every subplot is the original and the right image is the decoded image after passing through the autoencoder. We observe that the decoder images are blurry.
Figure 4.5: The plot shows the axial, sagittal and coronal cross sections of the brain from a subject. The MDA0 magnitudes from the diffusion MRI are plotted in the background and the normalized reconstruction error of a region across the population is superimposed on top. The colormap is from black representing 0 value to red to yellow being the highest value. Thus, regions in yellow have the highest normalized reconstruction error. We observe higher values in the cortex and deep white matter areas with high bending (ends of the corpus callosum in the middle picture). We also observe higher values in the symmetrical pockets marked 'X' in the axial view.
Chapter 5

Generative modeling of white matter regions

Lloyd [2002] provides a list of complexity measures which can be grouped under three questions: how hard is it to describe?, how hard is it to create?, and what is its degree of organization? The degree of organization measure of complexity quantifies the difficulty of describing the structure in a system and the amount of information shared between the parts of a system as a result of this structure. Inherent complexity of a region of white matter can be determined by how effectively it can be predicted given its neighborhood. This measure of complexity answers the what is its degree of organization? question. If a region of white matter can be easily deciphered from its neighborhood, the region is not very complex and vice versa.

While quantifying complexity using this measure, we also study generative modeling of white matter. We learn the inherent behavior of white matter structures in order to generate missing or new white matter regions. Generative modeling of white matter has various applications like denoising and generation of new scans and lesion reconstruction.

Inpainting is the task of filling missing regions based on the available information.
Figure 5.1: These figures show a 2D slice from a 3D white matter region. The subplots show the first MDA vector with the x and y vectors represented by the arrows and z vector represented by the color. The shadowed reflection of the arrows represents the symmetric property of the MDA vectors. The left figure (a) shows $X'$ with $X$, the region to be generated, in the red box. The right figure (b) shows $\text{context}(X)$ which is the input to the model.

We pose the problem of determining how effectively a white matter region can be predicted given its neighborhood as inpainting the white matter region given its neighborhood/context.

Let $X$ be the region of white matter whose complexity we want to measure.

\begin{align*}
X' &= X + \text{context}(X) \\
\hat{X} &= G_\theta(X' \odot M)
\end{align*}  
\(5.1\) \(5.2\)

where $M$ is a binary mask that masks $X$ in $X'$. $G_\theta$ is the parameterized model which generates $\hat{X}$, reconstruction of the original center $X$. This is represented in Figure 5.1 where $X'$ is shown in (a) and $X' \odot M$, which is the input to $G_\theta$, is shown in (b).

We solve the problem of inpainting and generative modeling with Generative Adversarial Network (GAN) conditioned on the context.
5.1 Generative Adversarial Network - Introduction

In a GAN framework, there are two models - generator and discriminator - which compete with each other in the training process. The generator synthesizes new data instances and the discriminator evaluates the authenticity of the generated data i.e. if it belongs to the training dataset or not. Thus, the generator captures the data distribution and the discriminator estimates the probability that a sample belongs to the input data distribution rather than being generated synthetically. Equilibrium is achieved when the generated data is indistinguishable from the actual data.

Let \( G_{\theta_g} \) and \( D_{\theta_d} \) be the parameterized functions representing the generator and discriminator respectively. We define a prior over the input noise variable \( z \), which is the input to the generator, as \( p_z(z) \). We learn the generator’s distribution \( p_g \) over the input \( x \) and the mapping to the data space is done by \( G_{\theta_g}(z) \). The discriminator \( D_{\theta_d} \) outputs the probability that \( x \) came from the data distribution rather than \( p_g \). The discriminator is trained to maximize the probability of assigning the correct label to both training samples and generated samples from the generator. The generator is trained to minimize the likelihood of the discriminator being correct on generated data. Thus, the training process is two-player minimax game represented by the following equation,

\[
\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{\text{data}}(x)} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right] \quad (5.3)
\]

[Goodfellow et al. 2014] shows that for a fixed \( G \), \( D \) converges to

\[
D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_g(x)} \quad (5.4)
\]

Also, the minimax game in Equation 5.3 has a global optimum for \( p_g = p_{\text{data}} \) if \( G \) and \( D \) have enough capacity. Thus, from Equation 5.4, \( D \) converges to \( \frac{1}{2} \) when \( p_g = p_{\text{data}} \).
Figure 5.2: This figure shows the training of GAN. Discriminator distribution \( D \) (blue, dashed line) discriminates between samples from the data distribution \( p_{\text{data}} \) (black, dotted line) and the generator distribution \( p_g \) (green, solid line). The lower horizontal line represents the domain from which the input to the generator, \( z \), is sampled and the line above that represents the domain of the input \( x \). The upward arrow between the lines represents the mapping \( G(z) \) to the data space with the distribution \( p_g \). In (a), \( p_g \) is becoming similar to \( p_{\text{data}} \) and \( D \) is a sub-optimal classifier. In (b), \( D \) is converging to \( D^*(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_g(x)} \) and gradients of \( D \) guides \( p_g \) to regions which are more probable to be labelled by \( D \) as true. In (d), the discriminator and generator have converged when \( p_g = p_{\text{data}} \). The discriminator cannot differentiate between the two distributions such that \( D(x) = \frac{1}{2} \). (Figure 1, Goodfellow et al. [2014])

Input to the discriminator cannot be differentiated as it can be from either the generated distribution \( p_g \) or the data distribution \( p_{\text{data}} \), hence it has 50% accuracy. Figure 5.2 depicts the training process of GANs.

When the discriminator classifies the generated sample as false with the high probability, \( \log(1 - D(G(z))) \) saturates. This hampers the flow of gradients from the discriminator to the generator thus stalling the training process. To avoid this, instead of training the generator to minimize \( \log(1 - D(G(z))) \), we can train it to maximize \( \log(D(G(z))) \) which gives higher gradients. Thus, instead of minimizing the likelihood of the discriminator being correct on the generated data, the generator maximizes the likelihood of the discriminator being wrong on the generated data.

Since the objective function consists of jointly minimizing and maximizing conflicting
objectives, the training process can be unstable.

5.1.1 Inpainting using GAN

Image inpainting is the process of recreating the missing regions of an image such that they are visually realistic and consistent with the surrounding regions of the image. Image inpainting can be done by GANs conditioned on the available regions of an image. The generator synthesizes the missing regions while the discriminator determines the consistency of the generated region with the global image. Thus, the objective function of a conditional GAN would be

\[
\min_{\theta_g} \max_{\theta_d} \left[ \mathbb{E}_{x \sim p_{\text{data}}(x)} \log D_{\theta_d}(x|y) + \mathbb{E}_{z \sim p(z)} \log (1 - D_{\theta_d}(G_{\theta_g}(z|y))) \right]
\]  

(5.5)

where \( y \) is the input variable on which the generator and discriminator are conditioned.

5.2 Related work

Natural images: Inpainting of natural images has been studied in the computer vision domain using traditional approaches (Barnes et al. [2009], Bertalmio et al. [2000], Efros and Leung [1999]) and recently using deep learning, which has emerged as a promising paradigm. Yang et al. [2016] propose a CNN based patch synthesis method based on joint optimization of image contents and texture constraints which works very well for high resolution images. GANs was introduced by Goodfellow et al. [2014] and have been applied extensively in the field of computer vision. Pathak et al. [2016] use context encoder which is an encoder-decoder pipeline to generate missing regions of an image. They train this network with L2 reconstruction loss and an adversarial loss. Iizuka et al. [2017] improve upon this by introducing two discriminators, global and local. Global
discriminator takes the entire image as input and tests if it is coherent as a whole while
the local discriminator looks at only the generated region to ensure local consistency. Yu
et al. [2018] propose a coarse-to-fine architecture with contextual attention for the gener-
ator such that the coarse network produces a rough outline of the missing region and the
fine network builds upon the coarse result along with contextual attention, which uses
the features of the relevant known patches, to generate a final image. They use local and
global Wasserstein GAN for adversarial losses and L1 reconstruction loss. There have
been studies focusing on generative face inpainting as well. Li et al. [2017] use a seman-
tic parsing network after the GAN network with a pixel wise softmax loss for consistent
inpainting of face images. Yeh et al. [2016] use a trained GAN and later search for the
closest encoding of the corrupted image to the input face image in the latent space. The
image is then reconstructed using the encoding.

**Medical imaging:** Low dose CT scans are beneficial as they reduce radiation expo-
sure to the patient but leads to increased noise and artifacts in the scans. GANs have
been applied to *denoising of low dose CT scans*. Wolterink et al. [2017b] first applied
GAN to cardiac CT image denoising. Yang et al. [2018b] use Wasserstein GAN along
with perceptual loss which compares the perceptual features of the denoised output with
that of the ground truth image in the feature space. Yi and Babyn [2018] use GANs
along with a sharpness detection network for denoising of CT scans of a series of dose
levels and anatomical regions. GANs have been used for *translation between brain scans*
i.e. MR scans to CT scans (Nie et al. [2016], Wolterink et al. [2017a]), T1 weighted scans
to diffusion MRI (Gu et al. [2019]), MRI cross-modality (Yang et al. [2018a]). Armanious
et al. [2018b] use GAN on three different tasks: PET-CT translation, correction of MR
artifacts and PET denoising. GANs have also been applied to improving resolution of
MRI images (Sánchez and Vilaplana [2018], Chen et al. [2018]), generation of 2D MRI
scans (Han et al. [2018]) and generation of 3D MRI scans with tumors (Shin et al. [2018])
Chapter 5

5.3 Using GANs to measure complexity

We use conditional GANs to measure the complexity of a white matter region. We sample $16 \times 16 \times 16$ regions from the brain containing mostly white matter from the 442 training subjects. Given this $16 \times 16 \times 16$ region, we mask the center $8 \times 8 \times 8$ region, $X$, whose complexity we want to measure. The remaining forms the context, $\text{context}(X)$, which is the conditional input to the generator. The generator reconstructs the masked $8 \times 8 \times 8$ region, $\hat{X}$, which is superimposed back on the context to form the generated region. This generated region along with the actual unmasked $16 \times 16 \times 16$ ground truth region becomes the input to the discriminator during training. This is depicted in Figure 5.3.
During inference, the $16 \times 16 \times 16$ region with the masked center is fed to the generator which reconstructs the missing region and the reconstruction error between the actual and generated region is computed. The challenge is to create visually realistic and plausible regions which are coherent with the neighborhood regions. We measure the comparative performance of a well trained generator across different white matter regions with higher reconstruction error signifying higher complexity of the region. To normalize across different regions, we divide the reconstruction error in a region with the average vector magnitude in that region.

### 5.4 GAN architecture

The architecture is shown in Figure 5.4.

#### 5.4.1 Generator

We use a **coarse-to-fine architecture** similar to Yu et al. [2018]. The coarse network produces an initial rough prediction for the missing region which is then superimposed
on the context. This is the input to the refinement network which generates the final refined output. Since the refinement network sees a more complete scene than the coarse network which only sees the context, it has a larger receptive field and generates a more accurate reconstruction of the missing region.

Instead of standard convolutional layers, we use **dilated convolutions** ([Yu and Koltun 2016](#)). Dilated convolutions increase the receptive area each layer uses as input without increasing the number of learnable weights in the kernel. This is done by spreading the kernel across the input with defined gaps (dilation rate). Thus, dilated convolutions with increasing dilation rates in consecutive layers exponentially increase the receptive fields without losing resolution or coverage. This is depicted in Figure 5.6. Dilated convolutions play an important part in the generator since they increase the spatial support of the voxels in the missing regions such that the support contains known voxels in the context. An example showing the benefit of using dilated convolutions is depicted in Figure 5.7.

We use a custom mean-squared error as the reconstruction loss to account for the symmetricity of MDA vectors. We refer to it as **symmetric MSE** and is defined as

\[
    L(X, \hat{X}) = \min(\| X - \hat{X} \|_2^2, \| X + \hat{X} \|_2^2)
\]  

(5.6)

While inpainting a region, there can be many plausible reconstructions for any given
Figure 5.6: The figures show consecutive dilated convolution layers applied to a 2D input. The red dots represent inputs to successive $3 \times 3$ kernels and the green area is the receptive field captured by the kernel on the initial input layer. (a) Layer 1: 1-dilated convolution applied on the initial input leading to a receptive field of $3 \times 3$. (b) Layer 2: 2-dilated convolution applied to layer 1 leading to a receptive field of $7 \times 7$. (c) Layer 3: 4-dilated convolution applied to layer 2 leading to a receptive field of $15 \times 15$. (Figure 1, Yu and Koltun [2016])

context. In challenging cases, there can be a plausible reconstruction of the missing region with voxels that are very different from the ground truth. Thus, strict enforcement of the reconstruction loss may hamper the training process. To avoid this, we incorporate spatial discount into the reconstruction loss. Intuitively, voxels near the boundary of the missing region have less ambiguity than ones closer to the center of the region. Thus, we have a penalty associated with the prediction based on the spatial position of the predicted voxel with respect to the context. We implement this using a weight mask $M'$ such that the weight of each voxel in $M'$ is set as $\gamma^l$, where $l$ is the distance of predicted voxel to nearest context voxel. Here, $\gamma$ is a hyperparameter which we set to 0.99.

The coarse network is trained with spatially discounted symmetric MSE while the refinement network is trained with symmetric MSE as well as adversarial loss. The
Figure 5.7: (a) shows the original region in the red box and the context outside the box. (b) shows the reconstructed region (in the red box) from a generator without dilated convolutions. (c) shows the reconstructed region from a generator with dilated convolutions. We can see the benefit of dilation by comparing the bottom right region inside the red box in (b) and (c), the former fails to capture the entire contextual information while the latter captures it due to larger receptive field and thus is more accurate.

The adversarial loss for the generator is

$$\max_{\theta_g} \mathbb{E} \left[ \log(D_{\theta_d}(G_{\theta_g}(z)) \right]$$

(5.7)

which maximizes the likelihood of the discriminator being wrong on the generated data.

The reconstruction loss is responsible for ensuring that the infilled region is plausible and consistent with the neighborhood but it tends to average the possible modes in the prediction, similar to what we saw with autoencoders. Adversarial loss ensures that the infilled region looks visually realistic and has the added benefit of picking out a particular mode from the distribution.

### 5.4.2 Discriminator

The discriminator network consists of 3D convolutional layers that function as feature extractors and is followed by a fully connected layer with sigmoid activation as the
The output of the fully connected layer is a continuous value corresponding to the probability of the input being real. We use convolutions with a stride of 2 to downsize the input to a layer while increasing the number of convolutional filters. We use \textbf{batch normalization} layer (Ioffe and Szegedy [2015]) after all the convolutional layers to stabilize training and facilitate the propagation of gradients. Batch normalization limits \textit{covariate shift} which refers to the change in the distribution of the input to a learning algorithm. In a neural network, the input distribution changes with each layer as the activations of the previous layer are the inputs to the next layer. Batch normalization is implemented by normalizing the activations of each layer which reduces the interactions between the layers.

The discriminator is trained with cross-entropy loss based on the actual and generated patches.

\[
\max_{\theta_d} \mathbb{E} \left[ \log D_{\theta_d}(x) + \log(1 - D_{\theta_d}(G_{\theta_g}(x))) \right] \tag{5.8}
\]

Figure 5.8: Discriminator network
5.5 Training details

In this section, we mention training details which help to stabilize and accelerate training.

We train the discriminator and the generator iteratively on each batch of the input. We use \textit{Leaky ReLU} as activation functions in all the layers of the generator and discriminator apart from the last layer of both. The last layer of the generator has the output values clipped whereas the last layer of the discriminator has sigmoid activation. We use \textit{L2 weight regularization} in all the layers of the generator and discriminator(with a hyperparameter of \(1e^{-5}\)) to prevent overfitting. We also employ some tricks for GAN training. We use \textit{one-sided label smoothing} while training the discriminator by replacing the labels for the real samples with a value slightly less than one, such as 0.9 (Salimans et al. [2016]). We also train the discriminator on \textit{noisy labels} by flipping the labels for 5\% of the samples.

The code is written in Keras with Tensorflow backend. We use ADAM optimizer (Kingma and Ba [2014]) with a learning rate of 0.00065 for the discriminator and 0.0005 for the generator. The training is done on 2 GeForce RTX TITAN GPU and takes an average of 10 hours.

5.6 Results

Figure 5.10 and Figure 5.11 show examples of original and decoded images from the validation set in which the reconstructions are accurate and not as accurate respectively. The average symmetric MSE is computed for a region of the brain across the subjects in the validation set and we compute this for multiple regions of the brain and observe that the values differ across the regions. Figure 5.9 shows the normalized symmetric
Figure 5.9: The plot shows the axial, sagittal and coronal cross sections of the brain from a subject. The MDA0 magnitudes from the diffusion MRI are plotted in the background and the normalized reconstruction error of a region across the population is superimposed on top. The colormap is from black representing 0 value to red to yellow being the highest value. Thus, regions in yellow have the highest normalized reconstruction error. We observe higher values in the cortex and lower values in the corpus callosum and front-back corridors (seen in the axial view). We also observe higher values in the symmetrical pockets marked 'X' in the axial view.

MSE error across different regions of the brain. We observe that the error is low in white matter regions with unidirectional fiber orientations such as the corpus callosum and the front-back corridor in the hemispheres. The two symmetrical pockets in the axial view correspond to the *crossing pockets* area that Volz et al. [2018] mentions. The dTDM mask by Volz et al. [2018] labelled this region with a value of three i.e. the number of distinguishable orientations is three. Thus, we observe that regions with differently directed fiber orientations have higher reconstruction error and thus higher complexity. This could possibly be due to multiple plausible reconstructions for any given context for such regions. We also notice that the reconstruction errors are slightly higher for GANs than autoencoders. This might be due to GANs sacrificing some error for reconstruction of realistic regions due to the adversarial component of the generator total loss.
Figure 5.10: The images are 2D slices of MDA vectors from 3D white matter regions. The subplots show the first MDA vector with the $x$ and $y$ vectors represented by the arrows and $z$ vector represented by the color. The left image in every subplot shows the region whose complexity we want to measure in the red box along with its context which is outside the box. The right image in every subplot shows the reconstructed region in the red box superimposed on the context. We can observe that the reconstructions are accurate even in cases like (d) when the context has sparse non-zero vectors.
Generative modeling of white matter regions

Chapter 5

(a)

(b)
Figure 5.11: These figures show examples in which the reconstructed regions are not very accurate. We can observe that some reconstructed vectors have incorrect direction and/or magnitude.
Chapter 6

Correlation between measures

In the above chapters, we discuss pairwise difference as a measure of heterogeneity and the performance of autoencoders and GANs as a measure of complexity. We use the deterministic trackable directions mask (dTDM) by Volz et al. [2018] mentioned earlier as our baseline measure for complexity. We measure the normalized pairwise difference, normalized reconstruction error for autoencoders and GANs across multiple regions (166400 regions) with a stride of 2 over all the axes on all the validation subjects and extrapolate to all the voxels. We then measure the correlation between these measures i.e. normalized pairwise difference, normalized reconstruction error for autoencoders and GANs and dTDM. We use Spearman coefficient for measuring correlation since one of our variables, dTDM, is ranked. Spearman coefficient measures the strength and direction of the monotonic relationship between ranked variables with higher values representing higher correlation and vice versa. The results are shown in Table 6.1. We observe that the measures highly correlate with each other.

Figure 6.1 shows the heterogeneity values from all the regions binned into 20 ranges and the corresponding average dTDM value, normalized average autoencoder reconstruction error and normalized average GAN reconstruction error for the bins in the subplots.
Table 6.1: Spearman coefficient between the different measures ($p<0.001$ for all)

<table>
<thead>
<tr>
<th></th>
<th>Pairwise Diff (Heterogeneity)</th>
<th>Autoencoders (Complexity)</th>
<th>GANs (Complexity)</th>
<th>dTDM (Complexity)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pairwise Diff</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Autoencoders</td>
<td>0.90</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>GANs</td>
<td>0.87</td>
<td>0.91</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>dTDM</td>
<td>0.85</td>
<td>0.86</td>
<td>0.81</td>
<td>–</td>
</tr>
</tbody>
</table>

(a), (b) and (c) respectively. We can observe that the curve for GAN reconstruction error and dTDM increases smoothly whereas the curve for autoencoder reconstruction error increases steeply and then plateaus. One possible explanation for this is the averaging effect of mean squared error used in the autoencoder model which tries to predict the mean of the distribution such that the mean voxel-wise error is minimized. Thus, the reconstruction error across various regions does not vary much.

It is intuitive that the complexity measures correlate with each other. As for the complexity measures correlating with the heterogeneity measure, it is difficult to decouple the two when we model complexity in a region across multiple subjects. Thus, heterogeneity inadvertently gets coupled with the complexity measure. Also, the normalized pairwise difference might not be an absolute measure for heterogeneity. This is likely because of the normalization process in which we divide the pairwise difference in a region with the average vector magnitude in that region. Lower magnitude vectors like in the cortex have more noise and thus more leeway for difference as compared to the higher magnitude vectors with more signal like in the corpus callosum. The normalization process affects vectors with small magnitude the most and thus the corresponding voxels have higher pairwise differences. Therefore, the normalization process does not preserve the pairwise difference perfectly and has an easier/harder time to do so where there are less/more
fiber crossings. This leads to the correlation with complexity as less/more fiber crossings signify less/more complexity.

Given two coupled variables, we need an absolute measure for either of the variables to effectively quantify the individual variables. This is similar to two unknowns in a set of linear equations of the form \( aH + bC = x \) where \( H \) and \( C \) are the unknowns. One possible way to solve this set of equations is to have an absolute measure for one of the unknowns. The methods we use for measuring heterogeneity and complexity do not provide an absolute measure and thus, it is difficult to quantify the two concepts individually.
Figure 6.1: Heterogeneity values from all regions are binned into 20 ranges and the corresponding average dTDM value, normalized average autoencoder reconstruction error and normalized average GAN reconstruction error are shown in (a), (b) and (c) respectively.
Chapter 7

Conclusion

In this work, we quantify the concepts of heterogeneity and complexity as a means of better understanding the morphology of the brain. We measure the voxel-wise difference between a region as a measure of heterogeneity and evaluate this measure on different white matter regions. We quantify complexity in terms of inherent dimensionality of a region using autoencoders and observe that the reconstructed regions are blurry due to the averaging effect of the mean squared loss function. We propose an improved complexity measure in terms of conditional dependency between a region and its neighborhood using GANs. GANs are also a means of generative modeling of white matter regions which has further applications like denoising and generation of new scans and lesion reconstruction. We study the comparative performance of a trained GAN on different white matter regions to identify particularly challenging regions of brain wiring. Also, inpainting of 3D diffusion MRI scans has not been done before and we obtain realistic reconstructions which cannot be distinguished from actual regions many of the times. We compute the correlation between these measures of heterogeneity and complexity to find that they highly correlate with each other since these measures do not provide an absolute quantification and thus, it is difficult to decouple the two concepts.
Chapter 8

Future work

A straightforward extension to this work is to study other complexity and heterogeneity measures which can be effectively decoupled. We can also compute the uncertainty estimates ([Kendall and Gal 2017]) for the generative model. Aleatoric uncertainty measures the noise inherent in the data and epistemic uncertainty captures the uncertainty about which model generated the data. We can capture the uncertainty associated with the reconstruction of a region of white matter which can be helpful for studying the neuroanatomy of the brain.

We can experiment with generative modeling for reconstruction of regions bigger than $8 \times 8 \times 8$ and also reconstructing multiple regions at once. The challenge with bigger regions is the computation cost since this would need more GPUs for training. As for multiple regions, one would need to reconstruct the regions together such that the regions are consistent with each other and their contexts.

We can also study the interpretability of the GAN model to find voxels in the context which are responsible for the predicted voxels in the missing regions. This can be achieved by masking voxels in the context during prediction and measuring the effects or by attention modeling.
Appendix A

Network Architectures

We describe the GAN generator and discriminator architecture in detail in this section. For simplicity, we use the following abbreviations, K (kernel size), D (dilation rate), S (stride size), C (channel number) and BN (Batch normalization). We use Leaky ReLU with $\alpha = 0.3$ for all activations apart from the last layer of generator and discriminator.

**Discriminator:** K3C32S1BN - K3C64S2BN - K3C128S2BN - K3C128S1BN - K3C128S1BN - dense layer to 1.

**Generator - Coarse network:** K3C128S1 - K3C128S1 - K3C128S1 - add - K1C64S2 - K1C3S1 - clip.

**Generator - Fine network:** K3C128S1 - K3C128S1 - K3C128S1 - add - K3C128S1D2 - K3C128S1D4 - K1C64S2 - K1C3S1 - clip.
Bibliography


H. Yu and S. Winkler. Figure 1, image complexity and spatial information. In *2013 Fifth International Workshop on Quality of Multimedia Experience (QoMEX)*, pages 12–17, July 2013. doi: 10.1109/QoMEX.2013.6603194.