UCLouvain

École polytechnique de Louvain

Fast matching in fingerprints dictionary using deep learning - application to estimation of brain microstructure

Author: **Louise Adam**
Supervisor: **Benoit Macq**
Readers: **Gaëtan Rensonet, Christophe De Vleeschouwer**
Academic year 2020–2021
Master [120] in Mathematical Engineering

Abstract

The impact and interest of deep learning has been growing recently and has empirically shown great precision and efficiency performance. Especially feed-forward networks are very fast to evaluate once trained and can theoretically learn any input-output mapping. This work investigates different methodologies to accelerate very significantly a slow but easily interpretable dictionary search, at the cost of a loss of interpretability. We concentrate particularly on the use of neural networks, which in this work perform the best in both precision and time efficiency, but also depict a more "black-box" character. This master thesis focuses on the problem of brain microstructure estimation using MRI signals and a dictionary of Dw-MRI fingerprints, with the goal to advance our knowledge of psychiatric and neurological disorders.

Acknowledgements

I would like to thank Professor Benoit Macq, supervisor of this master thesis, for giving me the opportunity to work on this fascinating subject, and also for his guidance. Furthermore, I would like to show my gratitude to Gaëtan Rensonnet who took the time to coach me and to help me carry out this work, by being a very valuable thought partner. I learned a lot through their insightful input and constructive feedback. Finally, I would like to thank Professor Christophe De Vleeschouwer, for accepting to be a jury member.

Contents

List of Figures

Acronyms

CSD	Constrained Spherical Deconvolution
CSF	Cerebrospinal Fluid
DL	Deep learning
DTI	Diffusion Tensor Imaging
Dw-MRI	Diffusion weighted Magnetic resonance imaging
ES	Exhaustive Search
FFNN	Feed Forward Neural Network
GBoost	Gradient Boosting
MRI	Magnetic Resonance Imaging
MSE	Mean Squared Error
MAE	Mean Absolute Error
NMR	Nuclear Magnetic Resonance
NN	Neural Network
NNLS	Non Negative (linear) Least Square
PFG	Pulse Field Gradient
PGSE	Pulse Gradient Spin Echo
RF	Radio Frequency (theoretical part), Random Forest (methodology)
SE	Spin Echo
SNR	Signal to Noise Ratio

Symbols

y	Dw-MRI signal
M	Number of sequences (measurements) in an acquisition protocol, size of a Dw-MRI signal
$\mathcal{P}$	Protocol
$\mathcal{D}$	Dictionary of fingerprints for a voxel
F_i	Single-fascicle sub-dictionary
N	Number of fingerprints in a single-fascicle dictionary
K	Number of fascicles in a voxel
Ω	Parameters, i.e. microstructural properties of diffusion environment
ν	Volume fraction of fascicles in voxel
r	Radius index of fascicles in voxel
f_{in}	Density index of fascicles in voxel
B	Magnetic field
ω	Frequence of precession
γ	Gyromagnetic ratio
δ	Duration of gradient
G	Intensity of gradient
Δ	Time between two gradient pulses
S	Signal
D	Diffusion coefficient

Introduction

Neurological disorders are the third cause of disability and premature death in the EU. Moreover, their prevalence is expected to increase with the progressive ageing of the population [2]. As a response to this, Dw-MRI provides a particularly promising non-invasive imaging tool for the study of the brain's microstructure and for the diagnosis of brain disorders.

Existing complex models already relate measured data to biological features of the brain white matter. Examples are axonal diameter estimations [3, 4], orientation and volume fraction of the axonal bundles [5, 6] and neurite dispersion [7]. Of all these models, the most physically realistic ones are numerical models known as Monte Carlo simulations [8], but the parameters used for these models are hard to estimate. One approach to estimate them is building a dictionary and using fingerprinting [9]. The problem is that for the problem of this work, this dictionary becomes multi-compartmental. As a consequence the dictionary search is slow and scales poorly with an increasing problem size, which makes it unusable for a too complex model. Hence the need for more efficient alternatives.

In parallel, the impact and interest of deep learning and machine learning models has been growing recently as deep neural networks have empirically shown great performance on a diverse range of problems [10]. Especially feed-forward networks are very fast to evaluate once trained, and can theoretically learn any input-output mapping. Their major disadvantage is their "black-box" behaviour that makes it hard to understand what occurs "under the hood". Having an interpretable model can be a major advantage to detect problems, to improve it or to adapt it for generalization.

In this Master's thesis we investigate models to efficiently solve the fingerprinting problem while preserving accuracy. We compare methods ranging from the exact (but inefficient) solution of the original problem, that is completely based on the biology of the problem, to an entirely "black-box" solution using a feed-forward deep neural network.

The machine learning models are trained on synthetically generated data. Having infinitely available data is theoretically very interesting for the training, but the synthetic aspect can also lead to overfitting. Moreover, the importance of the quality of this data comes to light when generalizing the models to experimental data.

Further, the approach will be generalized to other fingerprinting problems. Any dictionary search could be replaced by a neural network provided that enough data is available or can be generated.

Chapter 1

Background

In this chapter, we describe the physical and biological characteristics of the brain, that will be relevant in the problem description. We also cover the principles of MRI, the theories of Monte Carlo simulations and of deep learning algorithms.

1.1 Biology

1.1.1 The brain

The brain is the central organ that controls our thoughts, memory, movements and the function of many organs within our body. It receives information and processes it, thanks to 86 billions interconnected **neurons**. A neuron is a specialized cell that can transmit information through an electrical or chemical signal. Neurons have a cell body, an axon and dendrites (Figure 1.1). They connect through synapses, the contact points where one neuron communicates with another.

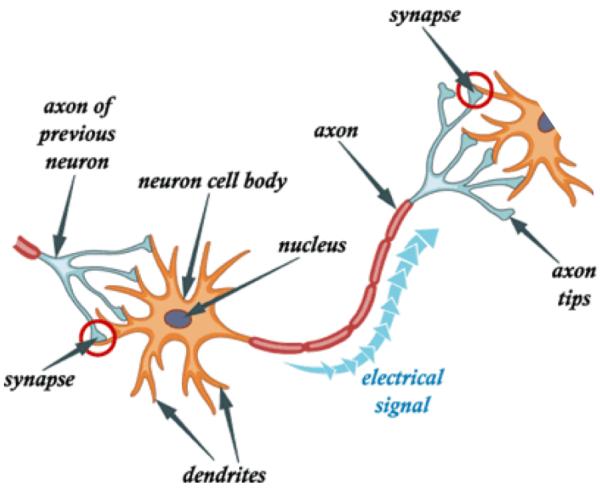

Figure 1.1: Schematic representation of biological neural cell [11].

The cell bodies of the neurons lie on the outer parts of the brain and form the grey matter. The inner part is filled with the axons that are responsible for transmitting the signals. The axons are surrounded by myelin, a very important insulating layer. The myelin is white, hence the lighter color and name of the **white matter** (Figure 1.2).

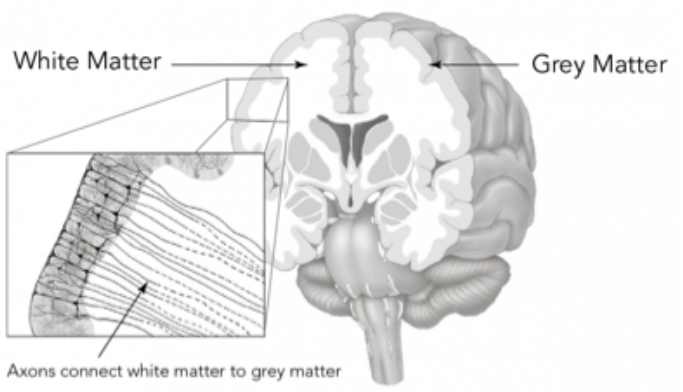

Figure 1.2: Representation of the brain with lighter inner part (white matter) and darker outer part (grey matter). Image from [12].

While grey matter is increasingly well understood [13, 14], knowledge about white matter remains very limited. One way to gain more information about white matter is using microstructure imaging [15]. This means that images of the brain are used to obtain information about properties of the tissues. More precisely for this work, some key properties of the microscopic structure of white matter will be estimated.

Images of the brain can be obtained with Dw-MRI (Diffusion-weighted Magnetic Resonance Imaging) which is a non-invasive and very precise brain imaging technique. Gaining information about the white matter microstructure should help detect and understand neurological disorders like Alzheimer, alcoholism and schizophrenia [16].

1.1.2 Axons in the white matter

As described in previous section, the white matter is filled with axons that are surrounded by a layer of insulating myelin. Parallel groups of axons are packed together to form fascicles (Figure 1.3). A fascicle can be described using many properties: the myelin around the axons has a certain thickness, axons can be crammed together or not (which gives an axonal density of the fascicle) and the axons all have a certain radius (which gives a radius index for the fascicle). In the white matter, several fascicles of axons with different properties intersect.

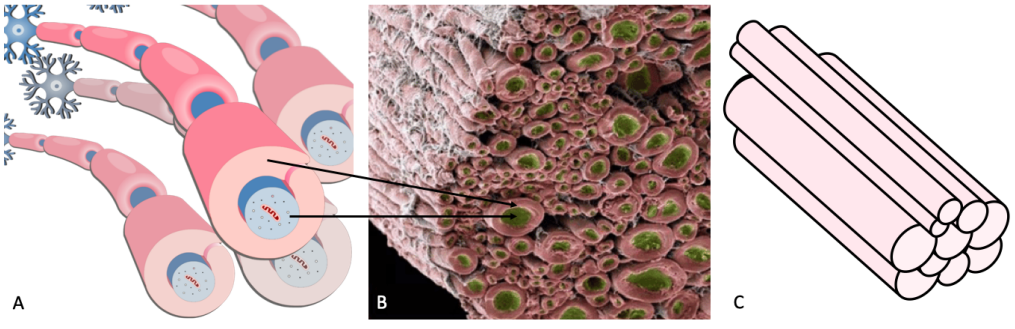

Figure 1.3: Microanatomy of white matter. A) Axons (blue) are surrounded by myelin (pink), a membrane tightly wrapped around the axons [17]. B) Parallel axons group together to form fascicles [18]. C) schematic representation of axons in a fascicle, further described in section 2.1.

In practice the axons are irregular and not perfectly parallel, which leads to structural disorder. Model simplifications made in this work are described in section 2.1.

1.2 Magnetic Resonance Imaging (MRI)

Magnetic resonance imaging (MRI) is a medical imaging technique to form pictures of the anatomy of the body. MRI scanners use strong magnetic fields, magnetic field gradients and radio waves to generate images. This technique is popular because it provides very precise images of body anatomy in a non-invasive way and without exposing the body to radiation [20].

This technique is an application of Nuclear Magnetic Resonance (NMR). Indeed, it is based on the knowledge that some atomic nuclei can absorb and re-emit Radio Frequency (RF) energy when placed in an external magnetic field. Radio frequencies are electromagnetic waves with a frequency ranging from $10kHz$ to $300GHz$ [21]. In medical MRI, it is the hydrogen atom that is used to absorb the RF energy because the human body is mostly made out of water (and so is the brain) [19, 20].

1.2.1 Spins of atoms

Every atom nuclei has a spin, which means they are rotating. A rotating charged particle induces the creation of a magnetic field, aligned with its axis of rotation. Without any external force, the magnetic moments of the atoms are randomly oriented and the total net magnetization is null.

But when an external magnetic field B is applied, the orientation of the spins will align (parallel or anti-parallel) with the magnetic field, resulting in a non-zero net magnetic field or **macroscopic magnetization** (represented on Figure 1.4).

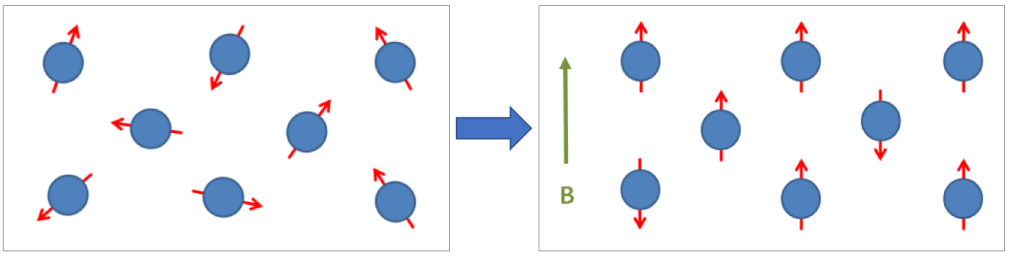

Figure 1.4: Spins of atoms align when a magnetic field is applied.

1.2.2 Steps of MRI

MRI makes use of this magnetic property of the hydrogen atom and introduces pulses of radio waves to get a signal (Figure 1.5 and 1.6) [22, 19].

1. First the brain is exposed to a strong magnetic field, which makes the spin of the atoms line up.

2. Then a RF pulse is applied. This disrupts the orientation of the atoms.

3. The atoms precess (see Figure 1.5) and by doing this, emit a RF signal that can be detected. The frequence of precession of the spins ω is linked to the force of the magnetic field B through the Larmor equation (γ is the gyromagnetic ratio, a constant specific to the atom):

$$\text{Larmor Equation: } \omega = \gamma B$$

4. The RF signal dies away and the spins realign with the magnetic field, this is called the T_2 relaxation.

The detected signal depends on the amount of spins that participate in the process, hence in our case the amount of hydrogen atoms in the tissues.

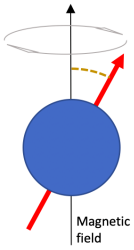

Figure 1.5: Spin precession of the atoms when an RF pulse is applied in a magnetic field.

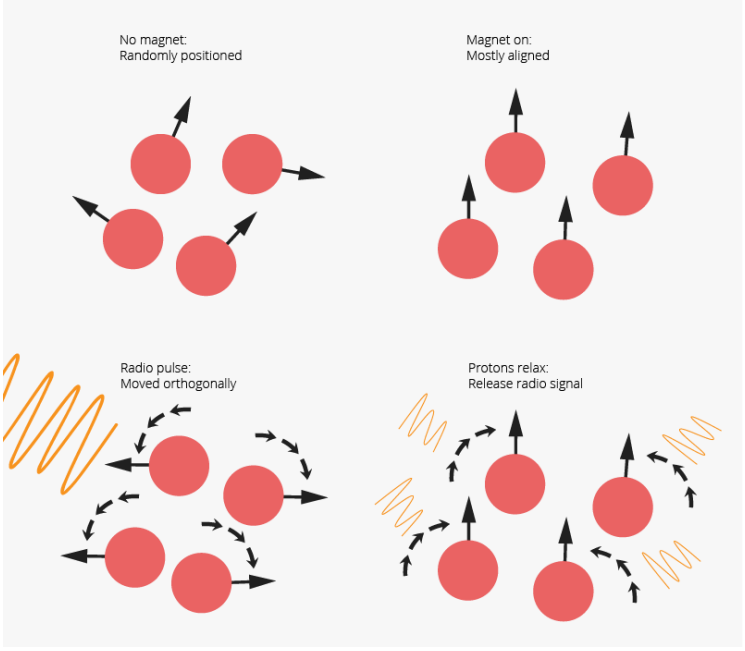

Figure 1.6: Steps of MRI [22].

1.3 Diffusion weighted MRI (Dw-MRI)

MRI can also be used to evaluate diffusion and is then called Diffusion weighted MRI (Dw-MRI). It extends the utility of MRI and enables to capture neuronal tracts in the nervous system. This section covers only a part of Dw-MRI technique and applications. For further insights, refer to [19, 23, 24].

1.3.1 Diffusion gives information about microstructure

The Dw-MRI technique is based on the diffusion of water in the tissues. Diffusion of water gives information about the 3D microstructure of the tissue. Indeed, if there are a lot of obstacles in a direction, the diffusion will be smaller in that direction, as shown in Figure 1.7. On this figure, water molecules will diffuse easily horizontally but will not be able to move vertically.

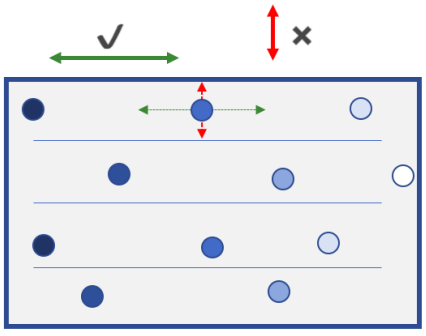

Figure 1.7: Diffusion gives information about the structure. The color gradient of the molecules (from dark to light blue) indicates the diffusion direction.

1.3.2 Measuring diffusion

To measure the diffusion through MRI, a magnetic **gradient** is applied, which will label the position of the atoms. After a certain time, a gradient of the same intensity and opposite direction is applied to refocus the spins. If the atoms did not move, the spins are completely refocused (left part of figure 1.8). In the other case, the phase shift of the spins is related to the displacement (right part of figure 1.8) [19].

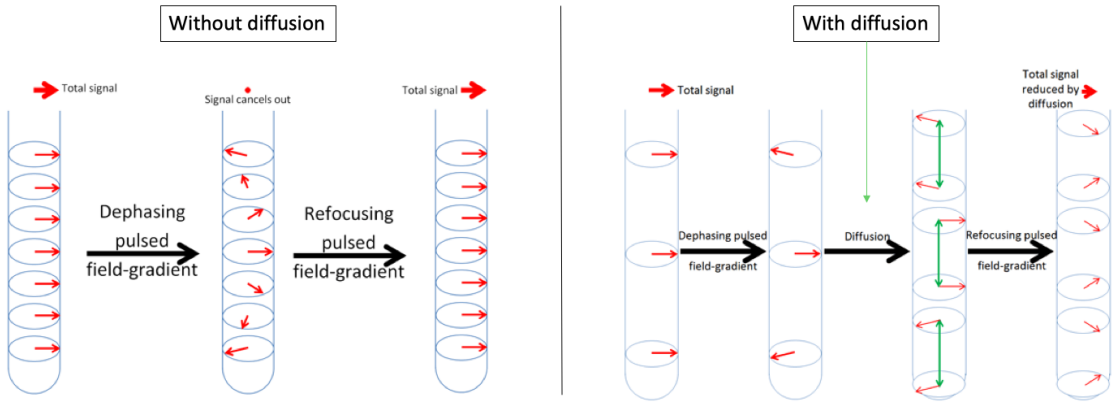

Figure 1.8: Gradient Pulse with and without diffusion. The phase shift of the spin atoms gives information about the displacement and this way also about diffusion [25].

1.3.3 Dw-MRI Sequences

The sequence of pulses used for Dw-MRI is called the Pulse Field Gradient (PFG) Spin Echo (SE) Sequence, or PGSE Sequence. The steps are depicted in Figure 1.9 and are the following [24]:

1. A $\frac{\pi}{2}$ RF pulse disrupts the orientation of the spins
2. A gradient pulse of duration δ and intensity G is applied. It dephases the spin according to their position
3. A π RF pulse reverses the sign of the precession
4. A second gradient is applied. This gradient should refocus the spins if they did not move. Otherwise the phase shift is related to their diffusion

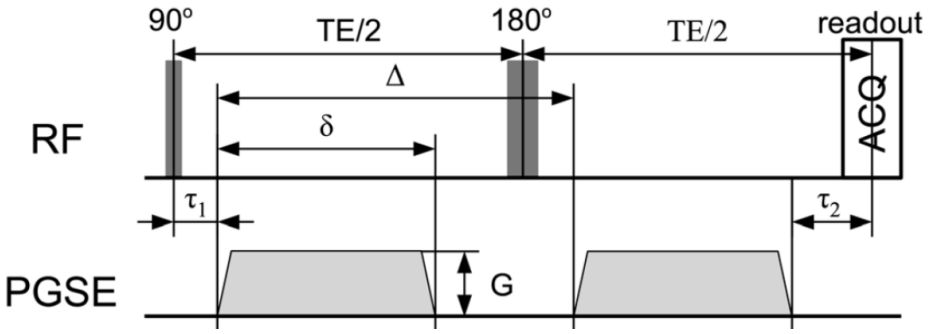

Figure 1.9: PGSE sequence (bottom row) and traditional SE sequence (top row). Figure taken from [24].

The PGSE sequence is characterized by a b_{value}, related to the duration δ and intensity G of the gradient, the length between two gradient pulses Δ and properties of the targeted element, i.e. hydrogen (through γ, the gyromagnetic constant) [19]:

$$b_{value} = \gamma^2 G^2 \delta^2 (\Delta - \frac{\delta}{3})$$

This sequence can also be adapted by changing the spin echo or by changing the gradient pulse. An example of the changing of the spin echo is STE (Stimulated Echo). In this case 3 impulses are given, which enables to make much longer sequences with more widely spaced gradients. Regarding the second case, when changing the gradient pulse, the gradient can take a sinus or cosinus shape (OGSE) or any other shape. A rotating field gradient could also be applied, where the gradient changes direction during the pulse [24].

1.3.4 From signal to diffusion coefficient

As expected (and explained briefly in previous sections), it is possible to measure the diffusion coefficient via the attenuation of the Dw-MRI signal.

By approximating the diffusion as free diffusion and by solving the physical differential equation of diffusion, the following relationship can be obtained [19]:

$$S = S_0 e^{-b_{value} \cdot D} \tag{1.1}$$

$$\ln\left(\frac{S}{S_0}\right) = -b_{value} \cdot D \tag{1.2}$$

In this equation, b_{value} is related to the chosen scanner parameters and is considered as known, the ratio $\frac{S}{S_0}$ is what is measured and D is the desired diffusion coefficient.

Using this equation we can see that a high b_{value} will give a lower signal, for a same diffusion. We can also note that diffusion (through the diffusion coefficient D) is inversely proportional to the signal ratio. This means a higher diffusion is characterized by a higher signal attenuation (i.e. less signal). On the DW-MRI-scan of Figure 1.10, lighter regions have a higher diffusivity. The signal is maximal when there is no diffusion or when the gradient is null, for example when the b_{value} is taken very small.

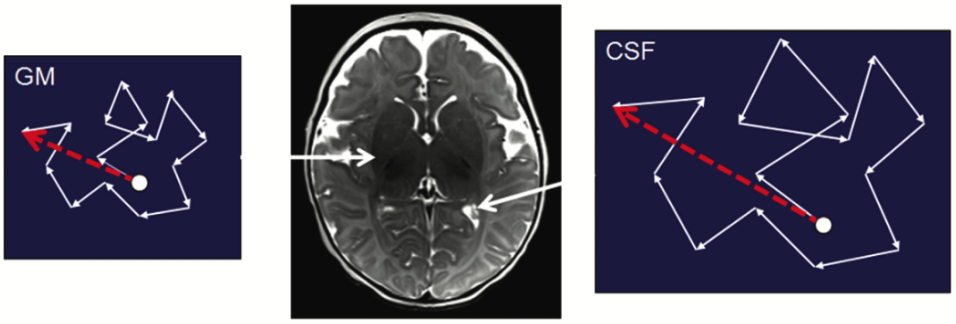

Figure 1.10: Graphical representation of the movement of water molecules in the Cerebrospinal fluid (CSF) on the right and Grey Matter (GM) on the left. CSF has a higher diffusivity than GM and is therefore colored lighter on the Dw-MRI image [26].

1.3.5 Types of diffusion

As shown in Figure 1.10 and Figure 1.11, different types of diffusion exist and can be associated to different types of tissue [19, 23].

- **Unrestricted diffusion** corresponds to diffusion without obstacles. It is the case for example in free water or more precisely for the brain, CSF. This type of diffusion is isotropic.
- Diffusion can also be restricted, but with random barriers and obstacles, which means there will be no prefered direction for the diffusion. This type is called **restricted isotropic diffusion**.
- Finally, diffusion can be **restricted** by barriers only in certain directions, which makes the diffusion **anisotropic**. This is the case for example in axons.

1.3.6 Diffusion Tensor Imaging (DTI)

To get information about the diffusion in 3 dimensions, gradients are applied with the directions taken on the sphere. The diffusion coefficient then becomes a symmetric diffusion tensor. Estimation of the 3D microstructure based on the diffusion tensors is called Diffusion Tensor Imaging (DTI), the idea is represented on Figure 1.11. It combines diffusion MRI with a diffusion tensor model [23].

Diffusion tensors are symmetric, which means they have 6 degrees of freedom. We thus need 6 PGSE sequences with different orientations to find them (and also a reference sequence with a low b_{value}). The diagonal components represent the diffusion coefficients along 3 principal directions and the eigenvalues of the tensors relate to interesting properties of the tissues. The trace also represents the diffusion in the voxel.

To link this technique to the brain, we can note that cerebrospinal fluid (CSF) can be seen as isotropic unrestricted diffusion, which is modelled with a sphere as diffusion profile (left column of Figure 1.11) and that axons are a typical example of anisotropic restricted diffusion, modelled as a long ellipsoid profile (right column of 1.11).

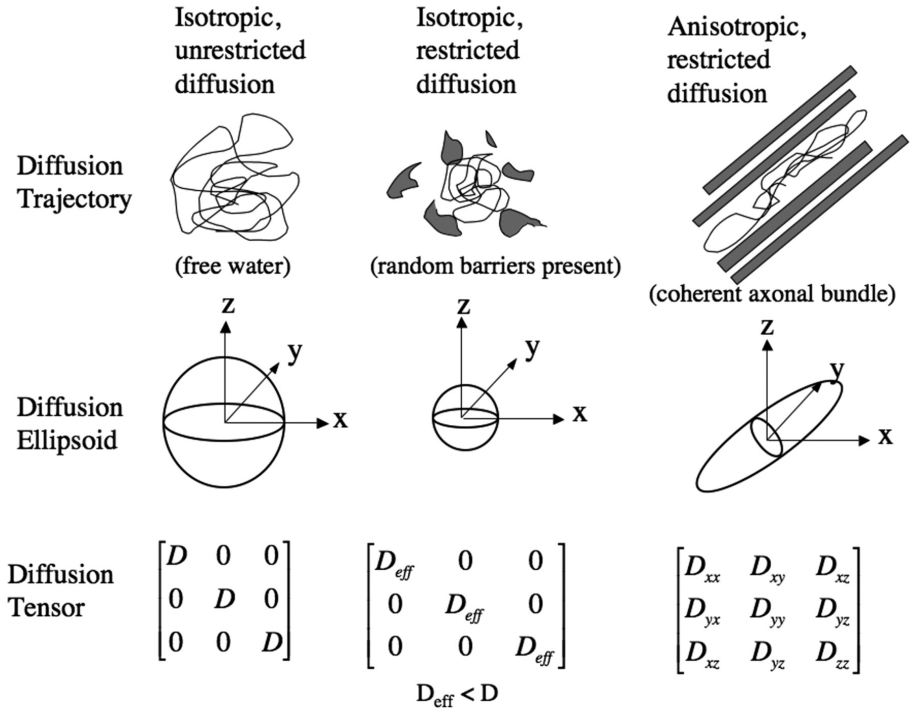

Figure 1.11: Diffusion Tensor Imaging (DTI) [23].

A limitation of this technique to estimate the microstructure of the tissues is that water molecules have to follow a Gaussian distribution, which means unrestrained diffusion.

1.3.7 Superposition principle

For the estimation of the microstructure, the technique of DTI will be combined with the superposition principle. This principle states that the signal obtained for the tissue can be seen as the sum of the signals obtained in the different compartments of the tissue [27]. It can be expressed by equation 1.3, where r_1, r_2 and ν_1, ν_2 represent respectively the radius index and the volume fraction of each fascicle. The principle is also represented on Figure 1.12.

$$signal(b, g) = \nu_1 \cdot signal(b, g, r_1) + \nu_2 \cdot signal(b, g, r_2) + ... \tag{1.3}$$

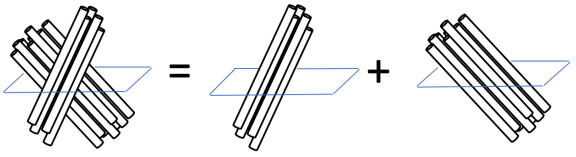

Figure 1.12: Superposition principle [27].

This works if there is no physical exchange between the compartments (or when these exchanges can be neglected), but it is not always the case. When the myelin disappears for example (Multiple sclerosis), there could be more exchange between the compartments. In this work, the different fascicles of axons are considered as non exchanging compartments, which has been shown to hold reasonably well in practice [27].

1.4 Monte Carlo Simulations

In recent years, a number of methods have been proposed to numerically construct white matter configurations. Having an analytical model for the reconstruction of white matter, Monte Carlo simulations can be used to validate and generate geometric configurations of white matter. Constructing realistic numerical representation of tissues is always linked with a choice of balance between realism and complexity. Figure 1.13 is an example of a realistic white matter voxel. The influence of various geometrical parameters present in white matter are shown, such as global angular dispersion, tortuosity, presence of Ranvier nodes and beading [1, 28].

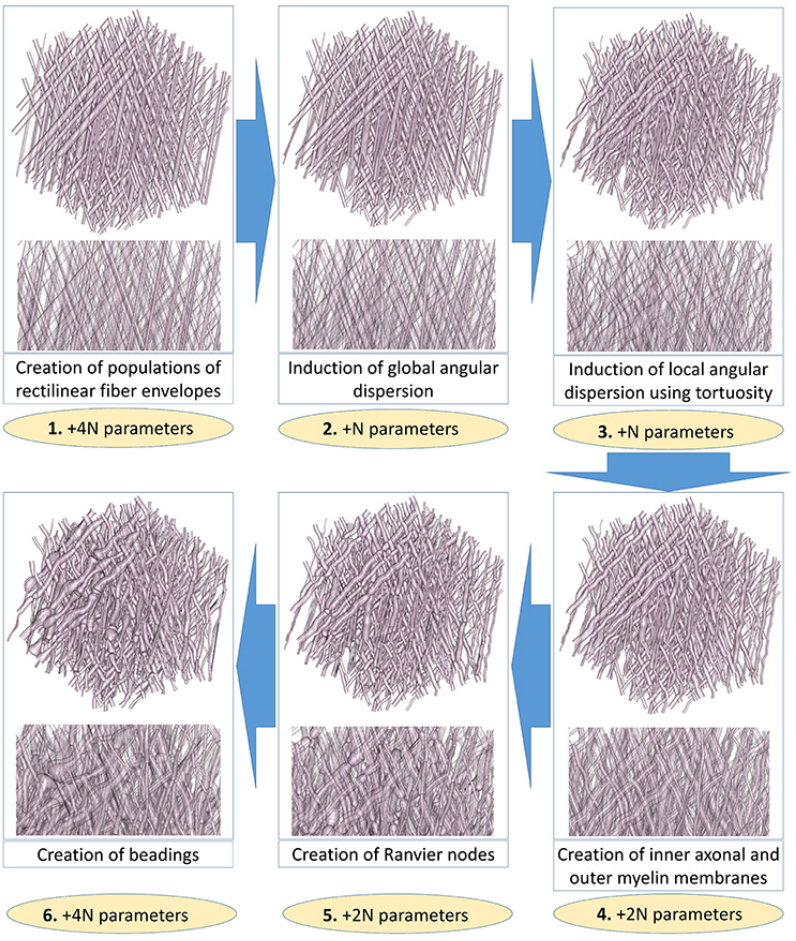

Figure 1.13: Realistic white matter substrates for Monte Carlo simulations. Building from (1) simple straight cylinders representing intersecting populations of axons, and adding gradually features such as (2) dispersion, (3) tortuosity, (4) myelin sheaths, (5) Ranvier nodes and (6) beadings [28].

More technically, following the idea of Dw-MRI, the spins are modeled as random walkers in a 3-D environment which is the brain and its microstructure. For each of the spins, that are initially uniformly distributed across the environment, a trajectory is computed and together with the PGSE sequence, is used to derive the corresponding phase and therefore the corresponding noise-free attenuation signal [29].

Monte Carlo simulators are based on the biology of the brain and can be viewed as a digital twin of the brain white matter. In this work, Monte Carlo simulations are used to create the canonical dictionaries of fingerprints that are described in section 2.5 and to generate synthetic data (described in section 3.1).

1.5 Deep learning of neuronal networks

Deep learning (DL) and Neural Networks (NN) are a subset of machine learning that uses networks capable of learning from data that is unstructured. Most part of the Neural network theory was developed before the 1990's, but like any other theory, to be actually useful they depended on several other factors. One of them is the training data that must be available in large numbers [31]. For this thesis the data is generated synthetically, which means an infinite amount could be available (without considering the necessary computer infrastructure to store the data and to train the network).

This section describes briefly how neural networks work and the hyperparameters that influence them, for further information refer to the book on which it is based [30].

1.5.1 Description

A neural network contains layers of interconnected nodes. Each node is a perceptron and is made of a linear combination of the nodes of the previous layer, weighted by some weights w. The perceptron feeds the signal produced by a multiple linear regression into an activation function that may be nonlinear. This result is then passed to the nodes of the next layer and the process can continue.

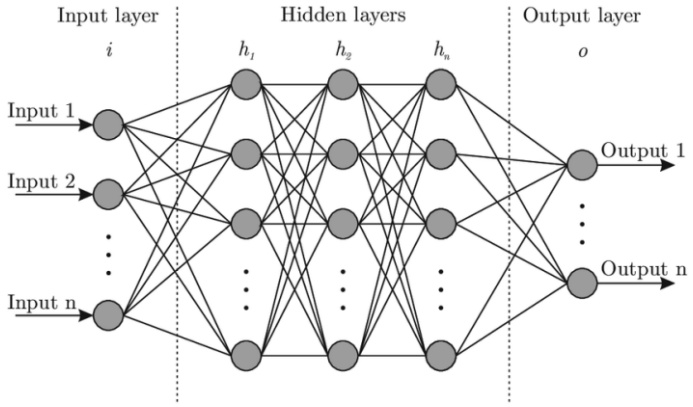

Figure 1.14: Schematic representation of a neural network with the input layer, the hidden layers containing hidden units and the output layer.

The activation function chosen for this work is the ReLU function, or Rectified Linear Unit function, as it is the most common function used for hidden layers [32]. It is common because it is both simple to implement and effective at overcoming some limitations of other activation functions, such as the vanishing gradients that prevent deep models from being trained. The ReLU function is calculated as follows: $f(x) = \max(0, x)$.

1.5.2 Training procedure

The input is fed to the network and after passing through the forward pass, a prediction is given as output. The objective of the training is to find the best weights to reduce the error between the predicted output and the desired output. To be able to do this, the data needs to be labeled, which means for every signal, the output is already known.

A neural network learns by using back propagation (as indicated on figure 1.15). It uses an optimization algorithm called Stochastic gradient descent that changes the weights and biases to minimize the error through a loss function. Different functions can be chosen to compute the loss,

a very common example is the Mean Square Error (MSE). The learning procedure is presented on Figure 1.15.

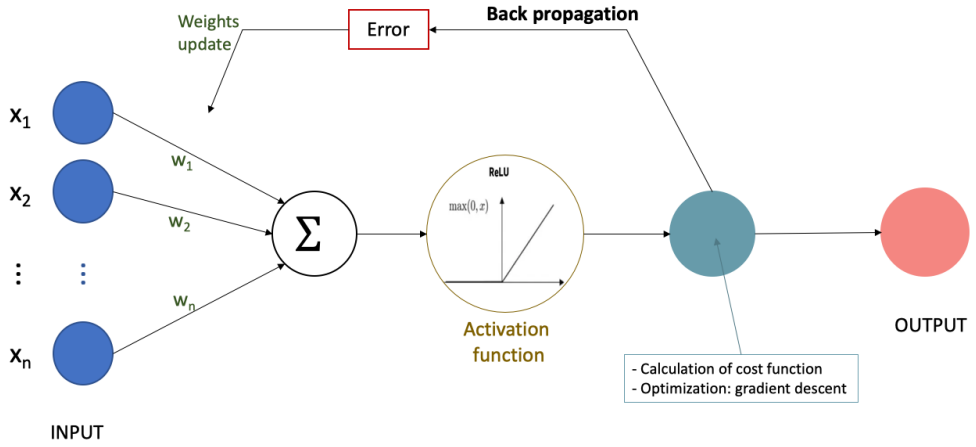

Figure 1.15: Learning process of a Neural Network with backpropagation on the weights. Inspired from [33].

1.5.3 Hyperparameters

Hyperparameters are the variables that determine the network structure and the variables that determine how the network is trained. They are set before training and can have an important influence on the final performance of the network.

Hyperparameters related to the network structure are the number of hidden layers and the number of hidden units. These numbers will depend on the complexity of the problem and on the amount of data that is available. Each hidden unit adds some degrees of freedom to the model, which allows the network to learn better, but also increases the complexity of the model.

Hyperparameters related to the training are:

- Learning rate: It is linked to the stepsize of the gradient descent and so it defines how quickly a network updates its parameters. A low learning rate slows down the learning process but converges smoothly, a larger learning rate speeds up the learning but may not converge.
- Dropout: It is the principle of cancelling random nodes during the training. This is a regularization technique to avoid overfitting.
- Activation function: This function brings non linearity to the network. Some common choices are the ReLU function, the sigmoid function or the hyperbolic tangent function.
- Number of epochs: Gives the number of times the whole training data is shown to the network while training
- Batchsize: It is the number of sub samples given to the network after which parameter update happens.

1.5.4 Architecture

The architecture of the network defines how the nodes are interconnected with one another and how many nodes and hidden layers are used. The most basic architecture is a fully connected multi layer perceptron (or MLP). As the name implies, in this type of network, all the nodes of one

layer are connected with all the nodes of the next layer. The neural networks build in this work will be based on this simple type of architecture.

Other types of well known architectures are [30]:

- Convolutional Neural Networks (CNN): very useful for the detection of patterns on images.
- Recurrent Neural Networks (RNN): they link nodes through multiple layers.
- Locally connected Neural Networks: unlike fully connected networks, the nodes of one layer are only connected to some nodes of the next layer.

These are only examples, as it is possible to build more or less any kind of architecture. This flexibility of deep learning models will be used in the perspectives of this work because it represents an opportunity for improvement.

Chapter 2

Problem description

This chapter presents the problem to be solved. It first makes the link with the underlying biology by describing the simplifications that are made, and it defines the objective, which is the estimation of some properties of the brain microstructure. It then develops the forward problem used to generate synthetic data and the inverse problem of estimation. Next, a section describes more precisely the input and output of the models. Finally the existing solution of fingerprinting is presented. This method is an exhaustive search through a dictionnary of fingerprints which gives a precise solution, only at the cost of an undesirable time complexity. As a response to this, the next chapter will then cover other possible ways to estimate the microstructure more efficiently.

2.1 Model simplifications

Constructing realistic numerical representations of tissues must strike the right balance between realism and computational complexity. The models presented here are a simplification of what we saw in the chapter on biology.

First, the simplest approach is to consider idealized tissue geometries such as modeling axons as cylinders. We also consider that these axons are infinitely long and thin, like straws, and that they are bundled together in fascicles. Crossing fascicles are an approximation for the complex patterns of interwoven axons. Moreover, using the superposition principle (section 1.3.7) [27], we assume the contribution of the each fascicle to the signal is independent.

Second, in a fascicle all axons are considered to be perfectly parallel (no undulation or dispersion), with a unique radius (interpreted as a radius index) and a global axon density. This means that we consider all axons of a fascicle to have the same properties. In this work we will consider that one voxel contains only two fascicles. All these simplifications are represented on Figure 2.1.

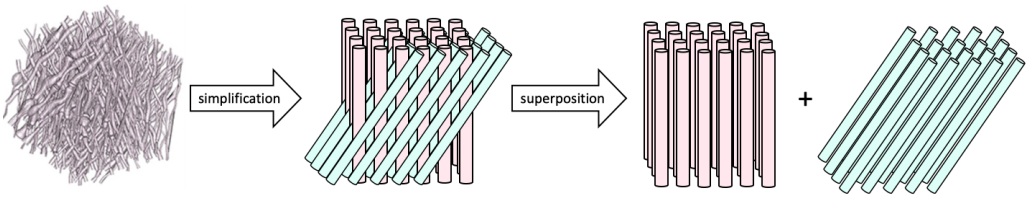

Figure 2.1: Simplification of the biology through following assumptions: the axons are parallel cylinders bundled in two fascicles, in a same fascicle the axons have uniform properties. Left part of image from [28].

2.2 Objective: estimation of microstructure

The objective of this thesis is to estimate key-properties of the microstructure of the brain white matter (see Figure 1.2 for the representation of the brain and brain white matter).

On a Dw-MRI image, the brain is divided in voxels, which are pixels in 3 dimensions (as can be seen on the left part of Figure 2.2). A Dw-MRI signal is represented by one value for each voxel. Further, each voxel is approximated as containing 2 populations of axons, called axon fascicles (pink and blue on the right part of Figure 2.2). This choice is one of the simplifications described in section 2.1.

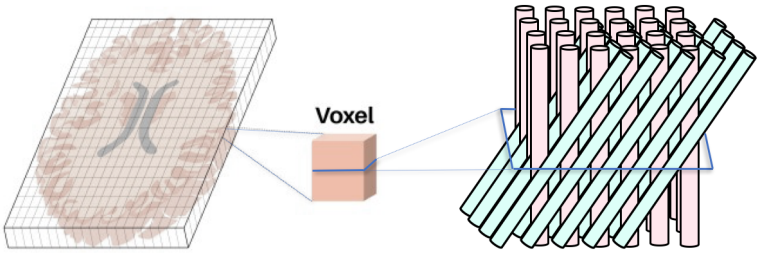

Figure 2.2: Composition of a brain voxel with the two axon fascicles (pink and blue). Left part of image from [34].

The 2 axon fascicles have different properties, which are the properties that will be estimated:

- A *volume fraction*: this value gives for each fascicle the fraction of the volume it takes in the voxel. Since the axons almost fill the voxel, the sum of the two volume fractions must be equal to 1.
- An *axon radius index*: this index is related to the mean radius of the axons of the fascicle
- An *axon density index*: This value gives an idea of the density of tissue in a fascicle.

We estimate three properties for each one of the two fascicles in the voxel, which makes 6 properties to estimate. These properties will be used frequently in the rest of this project, so it is useful to know their symbol and their magnitude

property	abbreviation	symbol	magnitude	units
volume fraction	nu	ν	~ 0.6	-
radius index	rad	r	$\sim 1 \cdot 10^{-6}$	m
density index	fin	f	~ 0.6	-

Table 2.1: Overview of desired tissue properties with their abbreviation (used for graph axes), symbol, magnitude and units.

The *orientation of each fascicle* is given by an angle relative to the z-axis. The orientations will also be needed and will be estimated beforehand. This is described in section 4.3.1.

2.3 Forward and inverse problems

The relationship between the Dw-MRI signal for one voxel and the properties of the microstructure in a voxel can be modeled in two directions, as represented on Figures 2.3 and 2.4. The forward problem consists in generating the right Dw-MRI signal based on the physics of Dw-MRI that are modeled as a mathematical function f. This function depends on an acquisition protocol ($\mathcal{P}$) and

on the properties of the microstructure (Ω). The other direction of the relation, i.e. the inverse problem, tries to estimate the microstructure of the voxel ($\hat{\Omega}$) based on a Dw-MRI signal (y).

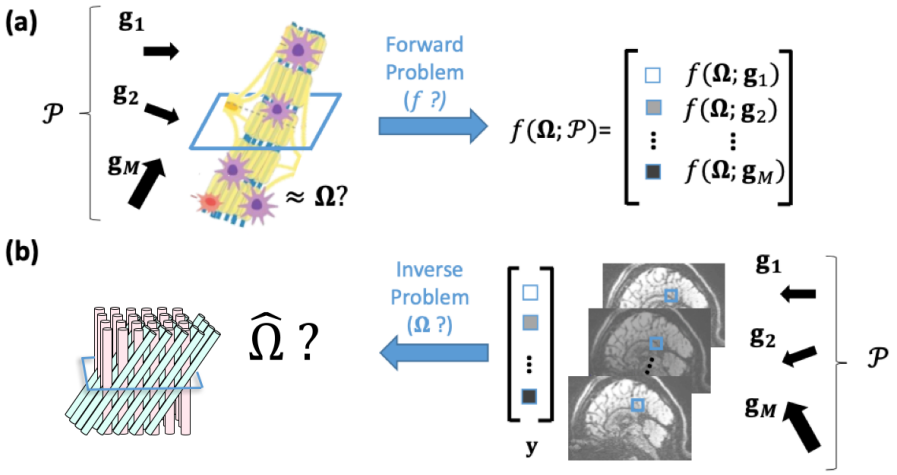

*Figure 2.3: **(A)** Voxel-level Forward problem to obtain the Dw-MRI signal based on a mathematical model (f), an acquisition protocol ($\mathcal{P}$) and the properties of the microstructure (Ω) and **(B)** the associated inverse problem consisting in finding an estimation for the microstructure of the voxel ($\hat{\Omega}$) based on a Dw-MRI signal (y). Parts of image taken from [1].*

To solve the forward problem, Monte Carlo simulations are used. This forward pass makes it possible to generate synthetic labeled data, i.e. Dw-MRI signals for which the underlying microstructural properties are known. The inverse problem is solved first through fingerprinting (section 2.5) and next through alternative methods described in section 3.

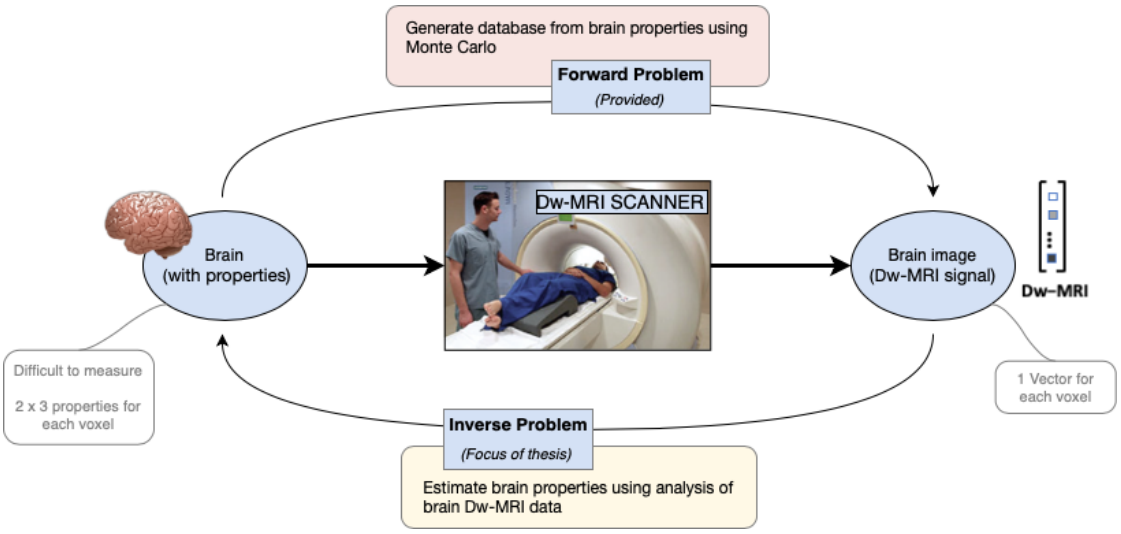

Figure 2.4: Overview of the forward and inverse problems: the forward problem uses Monte Carlo to generate synthetic Dw-MRI signals corresponding to tissue properties of the brain, the inverse problem estimates the tissue properties of the brain based on Dw-MRI signals.

2.4 Input and Output

Since the goal is to estimate the properties for one voxel, the input of the models will be a DW-MRI signal related to that voxel for M measurements with different acquisition parameters that depend on the acquisition protocol (different magnetic gradient intensities and orientations). This input vector is represented by the y vector on Figure 2.5. The size of this vector (M) depends on the acquisition protocol which is described in the section 3.1 on data generation.

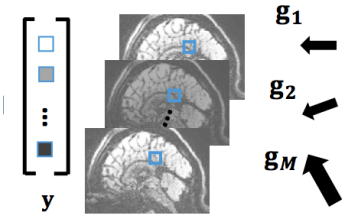

Figure 2.5: Dw-MRI-signal for 1 voxel [1].

One vector contains a lot of information about a single place (or voxel) of the brain. The desired outputs are the six properties of the fascicles in that same voxel. The goal is to get estimates as close as possible to the real values, therefore the problem is a regression problem.

To summarize, the methods that solve this problem take as input a DW-MRI vector corresponding to one voxel and give as output an estimation of the 6 desired properties for this voxel.

$$\underbrace{\text{DW-MRI signal}}_{[1\times M]} \longrightarrow \text{Model} \longrightarrow \underbrace{\text{axons fascicle properties}}_{[1\times 6]}$$

2.5 Fingerprinting

The inverse problem of estimating the microstructure can be solved using an Exhaustive Search (ES) through a generated dictionary of fingerprints. This section is based on the work done in [1].

2.5.1 Monte Carlo Dictionary

For this method, a dictionary $\mathcal{D}$ of Monte Carlo Dw-MRI fingerprints is computed with each fingerprint corresponding to a unique microstructural configuration. At runtime, for every voxel, the method then aims at finding the optimal combination of single-fascicle configurations.

The dictionary $\mathcal{D}$ is made up of K matrices F_k that are dictionaries relative to one fascicle in one direction (shown on Figure 2.6). To obtain a matrix F_k, a reference matrix F_0 is computed and then rotated in a direction u_k, which is the orientation of the fascicle k for which the properties will be estimated. To rotate the fascicles, their respective orientation has to be known beforehand. This is done in a precomputational phase described in 2.5.2.

$$\mathcal{D} = [F_1|F_2|...|F_K]$$

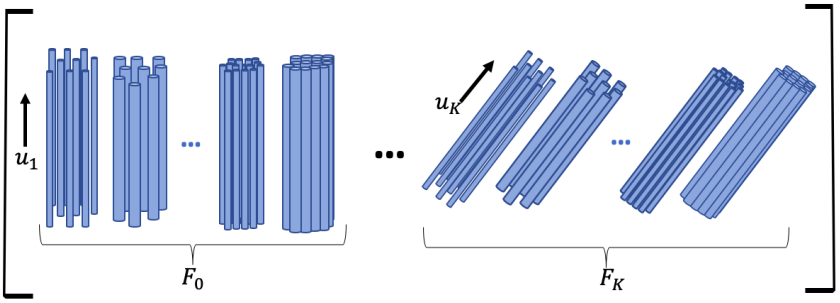

Figure 2.6: Visualisation of the Dictionary $\mathcal{D}$ as a combination of F_k's sub-dictionaries. The F_k's are rotated versions of a canonical fingerprints dictionary obtained with Monte Carlo simulations.

The number of rows of every F_k is defined by the number of acquisition parameters M, linked to the magnetic gradients and dependent on the chosen protocol. This number is the same as the size of a Dw-MRI signal. The number of columns of each F_k represents the number of possible fascicle configurations that are acquired and for which a matching will be tried. This value is denoted as N and is chosen to be equal to 782 [1]. A higher value of N would give a higher precision. Moreover, when increasing the number of parameters to simulate the model, N also has to be increased. K represents the number of different fascicle orientations that one voxel contains, which was chosen to be 2 for this thesis. The next equations show the dimensions of F_k and $\mathcal{D}$:

$$F_k : (M \times N)$$
$$\mathcal{D} : (M \times KN)$$

This dictionary is a powerful tool, inspired from the biophysics of the brain, that will be used to solve the problem of finding the right fascicle properties through dictionary matching.

2.5.2 Orientation estimation

The dictionary associated to one voxel is built by rotating and concatenating the canonical dictionary in the directions of the fascicles. To do this, the orientations are estimated based on the Dw-MRI signal. This step already brings errors, as will be detailed in section 4.3.1.

In this work, the Constrained Spherical Deconvolution (CSD) method is used because it is open source and easy to use [35] [36]. But as presented in [37], this is not the only possibility.

2.5.3 Exhaustive dictionary search

Let's recall that the objective of the model is to associate each one of the 2 fascicles of the voxel to a fingerprint (which represents a single-fascicle configuration) of the dictionary. For this we use the assumption that the 2 fascicles have an independent influence on the Dw-MRI signal of the voxel, which is called the superposition principle (see Figure 1.12).

To solve this problem, the first method that will be used is an exhaustive search. We try to find the combination of K fascicles that is the closest to the Dw-MRI signal and so associated to the smallest error. As mentioned earlier in chapter 2, here we consider the voxel as containing two fascicles, which means $K = 2$.

Figure 2.7 and 2.8 schematize the method. The first fascicle of orientation u_1 is associated to the subdictionary F_1 in $\mathcal{D}$. All the N possible configurations in F_1 give a signal s_1. The same holds for the second fascicle of orientation u_2 that also has N possible configurations with N possible signals s_2 in F_2. Using the superposition principle (section 1.3.7), the total signal relative to the voxel with two fascicles is the sum of these two signals ($s_{tot} = s_1 + s_2$). The objective is then to find the best combination of these two fascicle configurations, which means finding the best combination in $N \cdot N = N^2$ possibilities. The best possibility is the one with the estimated signal s_{tot} the closest (with respect to the 2-norm) to the Dw-MRI signal. The result then gives the identities of the combination and is of size 2.

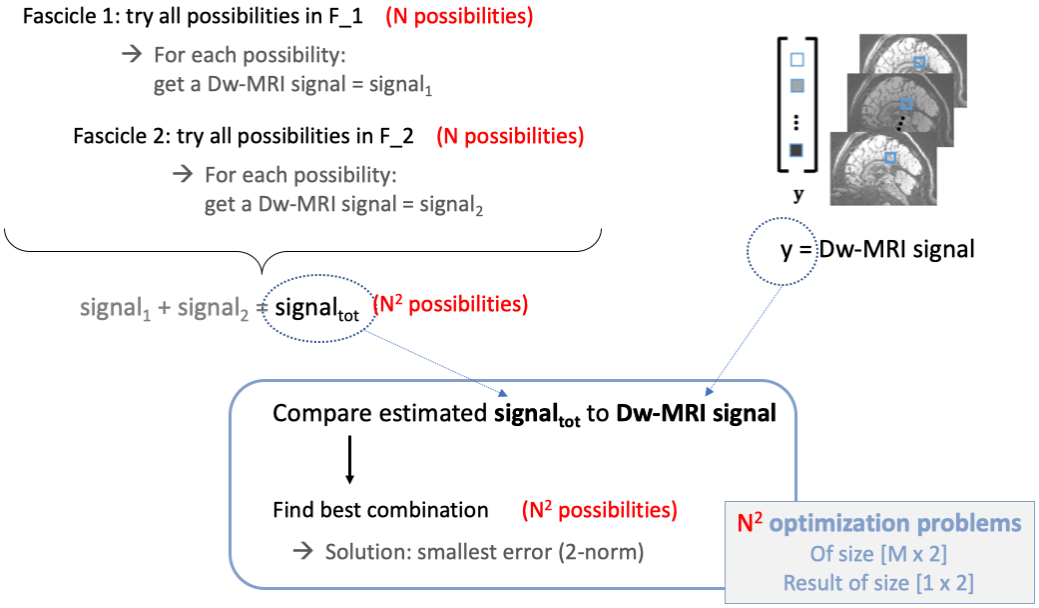

Figure 2.7: Dictionary matching: finding the best combination of fingerprints out of N^2 possibilities. The best combination is the one that has a minimal MSE between the reconstructed signal and the original one.

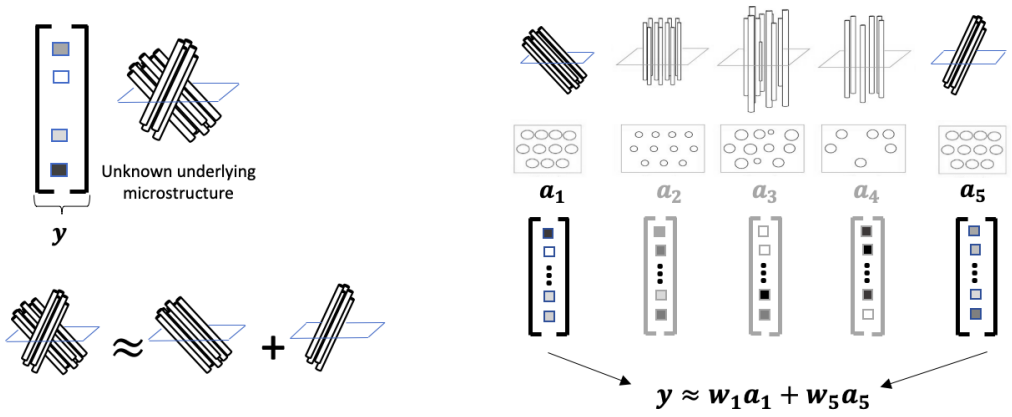

Figure 2.8: Dictionary matching: visual representation of selecting two fingerprints for which the sum of signals matches best the original Dw-MRI signal

To summarize and generalize to K fascicles, we try to find the best solution searching through the N^K possibilities. This means solving N^K optimization problems, each one of size $[M \times K]$. The result is then of size $[1 \times K]$ and gives the combination of K fingerprint that best corresponds to the microstructure of the voxel. The next equations express this problem.

$$\hat{\mathbf{w}} = \underset{\mathbf{w}\geq 0}{\operatorname{argmin}} \left\| \mathbf{y} - \left[\mathbf{F}^1 | \dots | \mathbf{F}^K \mid \mathbf{A_{csf}}\right] \cdot \begin{bmatrix} \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_K \\ w_{csf} \end{bmatrix} \right\|_2^2 \tag{2.1}$$
$$\text{subject to } |\mathbf{w}_k|_0 = 1, \quad k = 1, \dots, K$$

The sparsity constraints on the sub-vectors w_k guarantee that only one fascicle configuration per single-fascicle dictionary F_k contributes to the measured signal y.

2.5.4 Limitations

The dictionary search is a brute force method that searches through all the possibilities. The advantage is the high precision and certainty of the result, but this comes at the expense of a very high time complexity.

Indeed, the problem has a complexity depending on $\mathcal{O}(N^K)$. Since N has to be high to contain enough possibilities and so, allow enough precision, the total number of possibilities (equal to the size of the problem) increases really quickly when considering a number of fascicles $K > 1$. This method is therefore not adapted to solve problems when considering more than 1 fascicle in the voxel. Moreover, the problem size also scales poorly with the number of parameters that are taken into account because a larger number forces N to increase.

Another drawback is that for this method, the orientations of the fascicles in the voxel need to be known or estimated beforehand. There is thus need of a pre-computational phase to estimate these orientations, which leads to some errors and is also time consuming.

Chapter 3

Methods

The objective of the thesis is to compare different methods to solve the problem depicted in chapter 2 with an improved approach vs the exhaustive dictionary search. As described in chapter 2, the models will estimate some key properties of the tissue microstructure based on a Dw-MRI signal. These methods are alternatives to the precise but inefficient exhaustive dictionary search.

The first part describes the data used for the machine learning algorithms, since it is a very important component of this analysis. Next the three alternative methods are described, starting from one that still relies a lot on the understanding of the physics and biology of the problem and going to one that is completely black box. To push it a bit further, this last method could even take advantage of the structure of the data. The hope with this shift is to gain time and memory efficiency while keeping a good precision (Figure 3.1).

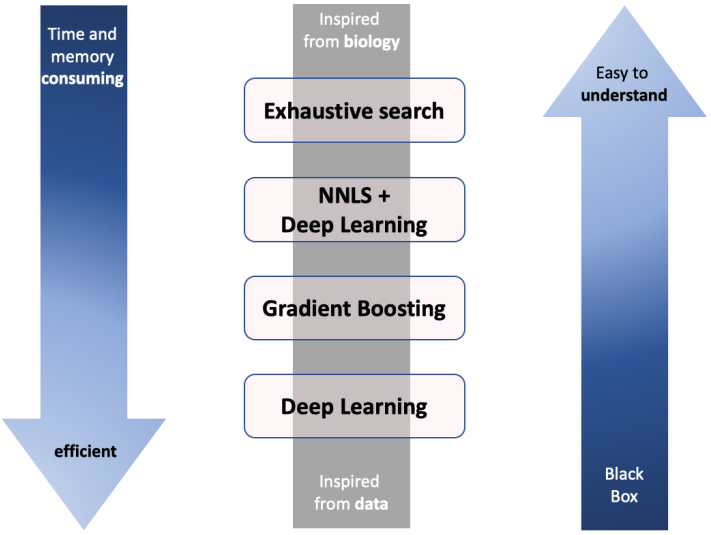

Figure 3.1: The different methods described in this master thesis: the exhaustive dictionary search and three alternative methods using machine learning and deep learning.

Choice of methods

The comparison of the methods will highlight advantages and drawbacks of deep learning techniques. In the last years, neural networks have been shown to outperform a number of machine learning algorithms in a very efficient way [38]. However, once the learning is done, it becomes nearly impossible to understand how the network makes the predictions.

Understanding how the model learns and what it learns could be very useful. This is why decision-trees have also been considered. Decision-tree based models are very powerful machine learning tools that also allow interpretation. Indeed they are made out of a series of trees in which every node uses a feature to divide into two branches.

Another way to better use the physics behind the model is to first extract some features out of the data by solving an optimization problem with dictionary matching and only after this, give the "filtered" version of the data to a deep learning network.

Methodology

The machine learning models are trained by minimizing the mean squared error (MSE) between the predictions and the targets (i.e. the 3 properties for the 2 fascicles). Using the MSE as loss function is interesting because it penalizes more the outliers (compared to the MAE for example).

The problem of estimating the properties is a regression problem. To train the models, labeled data is used. This data is synthetically generated, as described in section 3.1.

The evaluation of the models is done using the mean absolute error (MAE) because it shows directly the difference between predictions and targets.

All the codes implemented for this work are available on Github[1].

[1] `https://github.com/adamlouise/MasterThesis`

3.1 Data Generation

The data used for this thesis (i.e. the Dw-MRI signals presented above) is generated synthetically. As described in section 2.3, the problem of finding the microstructure in each voxel based on Dw-MRI can be divided in a forward problem and an inverse problem. The forward problem computes a Dw-MRI signal for the voxel based on a relevant set of tissue parameters Ω, a protocol $\mathcal{P}$ and a function f that relates the Dw-MRI signal in a single voxel to a gradient g and microstructural parameters. This forward model is the one used to generate synthetical Dw-MRI signals through Monte Carlo simulations.

Having data that is generated synthetically means an infinite amount is available. This is very interesting for the analysis of the different models and the training of deep learning and machine learning models.

3.1.1 Acquisition Protocol

The set of acquisition parameters is called the acquisition protocol $\mathcal{P}$, this can be seen as the MRI machine settings. The protocol used in this thesis is the the *MGH-USC Adult Diffusion protocol* of the *Human Connectome Project (HCP)* described in [39]. Varying the gradient intensity g and the time between two consecutive pulses Δ generates shells of different values of b_{value}. The protocol comprises 4 PGSE HARDI-shells containing:

- 64 gradient directions at $b_{value} = 1000 smm^{-2}$
- 64 gradient directions at $b_{value} = 3000 smm^{-2}$
- 128 gradient directions at $b_{value} = 5000 smm^{-2}$
- 256 gradient directions at $b_{value} = 10000 smm^{-2}$
- 40 b_0 images interleaved throughout the protocol

This makes a total of $M = 552$ values, associated to 1 voxel.

3.1.2 Forward problem used for data generation

The DW-MRI signal S at echo time under the application of a magnetic field gradient profile $g(t)$ in a voxel of white matter is assumed to arise from the independent contributions of K fascicles of axons with principal unit orientation $u_1, ..., u_K$ occupying fractions $\nu_1, ..., \nu_K$ of the physical volume of the voxel (section 1.3.7). Considering this, the signal S can be written using the equation 3.1. Here A_{fasc} is the signal corresponding to a fascicle with a certain microstructure Ω_k and orientation u_k.

$$\begin{aligned} S &= M_0 \cdot \left[\sum_{k=1}^{K} v_k A_{\text{fasc}} \left(\mathbf{\Omega}_k, \mathbf{u}_k; \mathbf{g} \right) + v_{\text{csf}} A_{\text{csf}} \left(D_{\text{csf}}; \mathbf{g} \right) \right] \\ &= \sum_{k=1}^{K} w_k A_k \end{aligned} \tag{3.1}$$

In this work the influence of Cerebrospinal fluid (CSF) will be neglected and the first orientation is always chosen along the z-axis.

3.1.3 Addition of noise

Signals from Magnetic Resonance Imaging (MRI) are corrupted by Rician noise, which makes image-based quantitative measurement difficult and negatively affects processing and analysis works. The Rician distribution is the probability distribution of the magnitude of a circularly-symmetric bivariate normal random variable [41].

To quantify the noise, the signal to noise ration (SNR) is used. This ratio is defined by equation 3.2 [1]. Here S_0 is the T_2-weighted signal when no diffusion gradient is applied (i.e. $g(t) = 0$ for all t), with echo time TE and T_{2k} the T_2 value of compartment k. σ is the standard deviation of the Gaussian noise process in an MRI receiver coil.

$$SNR = \frac{S_0}{\sigma} \tag{3.2}$$

$$S_0 = M_0 \cdot \sum_{k=1}^{K} \nu_k \exp\left(-\frac{TE}{T_{2k}}\right) \tag{3.3}$$

For SNR (Signal to Noise Ratio) large enough (larger than 3) [41], the Rician noise distribution starts to approximate the Gaussian disctribution. See [40, 41, 42] for more on this topic.

To make the synthetic data as close as possible to the real data, **Rician noise** is applied with a uniform distribution and different intensities. The parameter that gives the noise intensity is the SNR. A high SNR gives a signal of good quality while a low SNR gives a signal highly corrupted by noise. The Rician noise model is given by equation 3.4 where e_1 and e_2 are Gaussian random variables with zero mean and standard deviation σ, and where S is the true underlying signal.

$$S_{noisy} = \sqrt{(S + e_1)^2 + e_2^2} \tag{3.4}$$

An important requirement for the methods is that they should also be able to perform well on noisy data, in order to generalize to real data. In this work the models were trained on signals generated with an SNR uniformly distributed between 50 and 100 and were then tested on signals with SNR values ranging from 10 to 100.

3.1.4 Scaling

The labels (i.e. the properties of the tissues) are each independently rescaled with the `StandardScaler()` (from the python library `sklearn` [43]) such that they would have a mean of zero and a standard deviation of 1. This is important since the properties have very different ranges of values (see Table 2.1); for example the radius has a very small magnitude compared to the other properties and would therefore be neglected in the cost function and so in the training of neuronal networks, which is based on a gradient descent.

The importance of scaling for gradient descent based algorithms can be seen in the formula 3.5; the feature value x will affect the step size of the gradient descent and the difference in ranges of features would cause different step sizes for each feature [44].

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)} \tag{3.5}$$

To analyse the performance of the network, the real errors can be obtained by descaling the outputs with the function `inverse_transform()` of the same scaler.

3.1.5 Training and validation data set

The data set is divided in a training and a validation set. These two parts will never be mixed as it is important to keep a validation set that has never been seen during training. The number of samples chosen to train the networks in the methods of sections 3.2, 3.3 and 3.4 is $400,000$. The validation set contains $100,000$ samples.

To get better performances, the number of training samples can be increased, but this is of course associated to a higher computational time and memory cost. Moreover the results of sections 3.2.2.4, 3.3.2.1 and 3.4.2.4 show that this amount is large enough for all models trained in this chapter.

3.1.6 Baseline

In order to assess the performance of the methods, the Mean Abslute Error (MAE) will be used to compare the predictions with the ground-truth. To get an idea of the range of MAE we would like to achieve here, we can compare it to a baseline, which is a score we can achieve with no efforts. To keep things simple the predictions of the baseline model will be the mean. Since the data is standardized, the $mean = 0$ for each property.

When estimating our scaled test set by 0, we get the following mean error:

$$\text{MAE} = 0.833$$

The Mean Absolute Error developed for each property gives:

volume fraction	radius index	density index
0.709	0.866	0.865

Table 3.1: Detail of baseline error on the properties

This baseline can be obtained very easily and gives an upper bound on the accepted error for the methods described in this chapter.

3.2 NNLS followed by Deep Learning

NNLS (Non Negative (Linear) Least Squares) followed by Deep Learning is a two-stage resolution which first includes an optimization problem. The sparse solution of this optimization problem is then fed to a deep learning network. This method is based on the work in [1].

3.2.1 Description

The next approach builds on the first one (reference method, i.e. exhaustive dictionary search) to improve the time complexity. It is a 2-stages method that first solves an optimization problem, making a rough dictionary matching with the dictionary described before, and then applies a deep learning network to the result.

The 2 stages are:

1. Solving a NNLS optimization problem
2. Applying a Deep Learning algorithm, more precisely a Feed Forward Neural Network (FFNN)

3.2.1.1 NNLS

The first step is the optimization problem. In this step, the input (Dw-MRI vectors) are projected into a latent space. The aim by doing this is that the important information will be filtered and it will afterwards be easier for the deep learning network to learn from the data.

This step consists in solving a Non Negative (Linear) Least Square problem (NNLS). The NNLS finds the optimal weights $\boldsymbol{w}$ that would, when multiplied by a precomputed single fascicle dictionary, give the best description of the original signal y (equation 3.6).

$$\hat{\boldsymbol{w}} = \text{argmin}_{w \geq 0} ||y - \mathcal{D} \cdot \boldsymbol{w}||^2 \tag{3.6}$$

In this equation, $\mathcal{D}$ is the dictionary of precomputed signals F_k of single fascicles rotated along orientations u_k:

$$\mathcal{D} = [F_1|F_2|...|F_K]$$

The optimal weights $\hat{\boldsymbol{w}}$ are a mapping of the vector y (Dw-MRI signal) into a latent space of single fascicles. Therefore they are made out of 2 parts which each correspond to one of the fascicles of the voxel.

The difference with the first approach is that the exhaustive search is replaced by a single (large) optimization problem. This means the N^K problems of size $[M \times K]$ become 1 large optimization problem of size $[M \times (KN)]$. The result of this problem is a combination of one or more fingerprints for each fascicle of the voxel.

Because of the characteristics of the problem, we have the guarantee that the optimal solution will be found. First the number of variables is finite and the objective decreases at each iteration, which means the algorithm always converges. Second, the problem is convex, which means the optimum is always global. By knowing that the solution w is sparse, selecting $w = 0$ as initial candidate makes the algorithm particularly efficient.

Orientation estimation Again for constructing the dictionary, the orientations of the fascicles have to be known and thus estimated. The same approach is used as the one for the exhaustive search (section 2.5.2) and will be analysed in section 4.3.1.

Weights The vector $\hat{w}$ can be divided in two parts, each linked to a rotated fingerprint dictionary F_k (a version of F_0 rotated in direction u_k). It contains a weight associated to every fingerprint and represents the "best" combination of fingerprints to approximate the fascicles k of the voxel. These vectors of weights are naturally sparse (property of the NNLS optimization problem). In this master thesis, they contain around 8 non zero elements (out of a vector of size 1564), those numbers are very specific to one type of dictionary, i.e. one model of fascicle (namely, hexagonal packing) [1]. Figure 3.2 represents an example of such a vector, on the left part, the values of $\hat{w}$ are all plotted to highlight the sparsity of this vector. Unlike the original exhaustive dictionary search (section 2.5.3), there is here no guarantee that one of the weights corresponds effectively to the right fingerprint for the approximated fascicle, in other words there is no obvious link between the non zero values of this vector and the fascicle configuration.

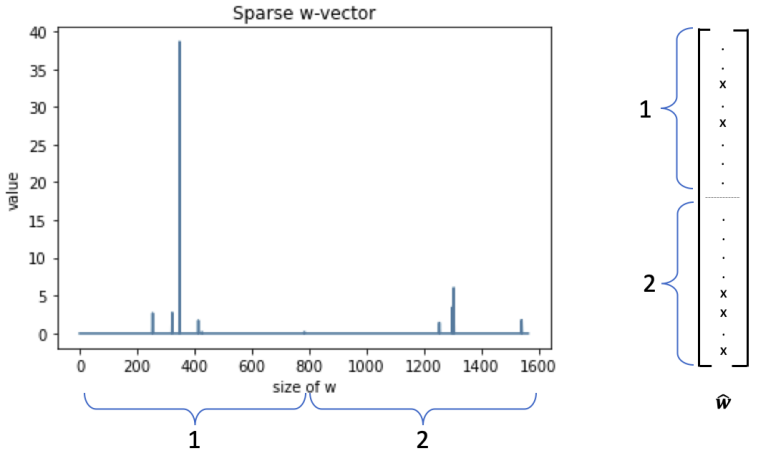

Figure 3.2: Visualisation of a sparse vector w: the vector can be divided in 2 parts that each correspond to one of the two sub-dictionaries and thus also to one of the two fascicles. Each part contains only a small number of non-zero values.

3.2.1.2 Deep Learning Network

Once the vector of weights is obtained in the first step, it is fed to a deep learning network in the second step. This deep learning network is more precisely a feed forward, fully-connected Multi Layer Perceptron or MLP. It is a machine learning technique that uses a network of weights to learn from the data. It thus needs to be trained. More explanation about this kind of networks is provided at section 1.5.

Architecture The built network splits the vector $\hat{w}$ in the middle and applies the same FFNN independently to both parts of the vector (as depicted in the diagram of Figure 3.3). The results are then combined and fed to a common FFNN. The advantage of applying the same network to both parts is that this split network is trained twice as much. Also the two parts of $\hat{w}$ are independent and represent the same since they are each the projection of 1 axon on a single axon dictionary. Considering this, it seems logical to apply the same network to both parts. The output of the network is a vector of size $3K$ that gives the estimation of the properties of the K fascicles in the voxel.

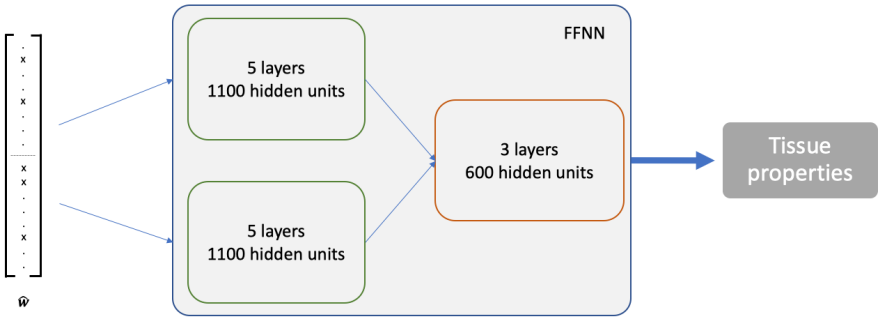

Figure 3.3: Architecture of the split neural network (the numbers of hidden units corresponding to the network build for this work).

3.2.1.3 Overview of the method

Figure 3.4 gives an overview of the method and of the two stages. To recap, the Dw-MRI signal of the voxel is given as vector to an optimization problem called NNLS. This optimization problem maps the signals into a latent space corresponding to the weights associated to the fingerprints for each fascicle. These weight vectors are then split and given as input to a deep learning neural network. The network finally outputs estimations of the properties of the two fascicles in the voxel.

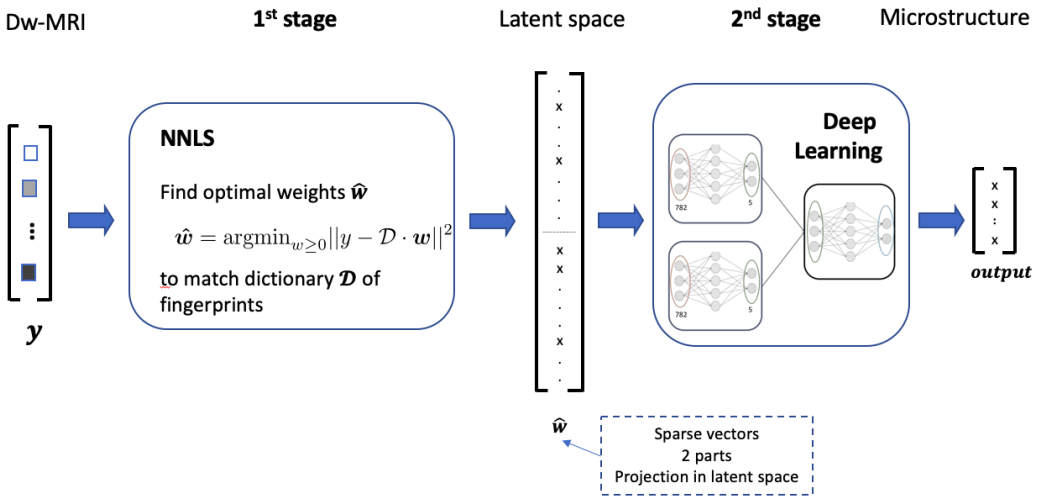

Figure 3.4: Description of the 2 stages of the method. First an optimization problem is solved, the input is mapped into the latent space of fingerprints through a sparse vector. Next this vector is given as input of a neural network to output the desired properties.

3.2.2 Training

As explained in section 1.5, the performance of the network depends on its training which in turn depends on many aspects. For this thesis, the training was performed on a set of $400,000$ labeled samples that were obtained by solving the NNLS problem on Dw-MRI data with SNR between 50 and 100 (section 3.1.3).

The network is trained with the `Adam` optimizer, an algorithm for first-order gradient-based optimization of stochastic objective functions, introduced in [45]. It was chosen because it is computationally efficient, has little memory requirements and is well suited for problems that are large

in terms of data and parameters. Moreover the hyper-parameters have intuitive interpretations and typically require little tuning [45], as will be confirmed in section 3.2.2.3.

3.2.2.1 Learning

To analyse the learning process of a network, it can be very interesting to look at the learning curve. This learning curve depicts the error (or accuracy) obtained at each epoch (i.e. each update of the weights).

For this network, the learning curve is shown on Figure 3.5. The error is the mean of the absolute error on the 6 properties. This graph confirms that the network is able to learn because the error decreases, it also shows that the learning is quite smooth and that the final precision is around 0.31, which is a lot better than the baseline error of 0.833. Further it gives information about the difference between the training error and the validation error. When there is a large gap between the training and validation curves, it means that the network is overfitting, it learns properties from the training data that are not in the validation data set. But when these curves completely stick together, it often means that there is room for improvement by enlarging the network. The size of the network as well as regularization (to avoid overfitting) can be controlled through the hyperparameters.

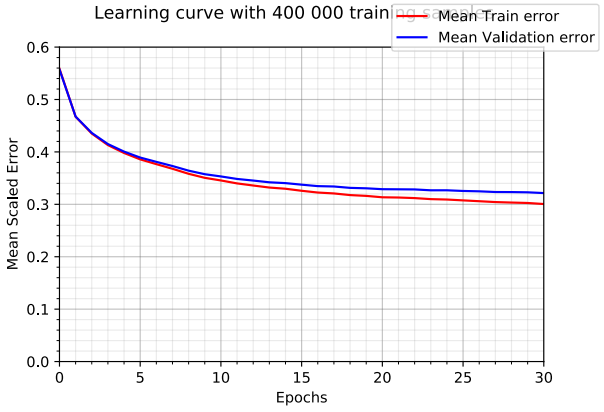

Figure 3.5: Learning Curve showing the decrease of the error over the updates and comparing training (red) and validation (blue). The "mean scaled error" is here the MAE over the scaled properties.

Next it is also interesting to look separately at the learning curves for each of the estimated properties. This is shown on the graphs of Figure 3.6. From these graphs we can derive that the learning for the same properties on the 2 fascicles is linked (for example ν_1 for fascicle 1 and ν_2 for fascicle 2).

Further we also see that the error for the estimation of the radius is proportionally smaller than the ones for ν and f. This can be explained by the fact that this error is standardized (since it was computed by comparing true and estimated standardized properties). Moreover the values of the radius indexes in the dictionary range over a large interval compared to real biological values. This might explain the impression of smaller radius error compared to the other properties.

To get a better idea on how this scaled error translates to the real units, we can look at the one-sided confidence intervals on the absolute error of each property and compare this with their typical magnitude (see table 3.2). This table confirms that the estimation of the radius index is not so good compared to the typical magnitude: 95% confidence that the error is smaller than $1.33\mu m$, compared to the typical magnitude of $0.5\mu m$.

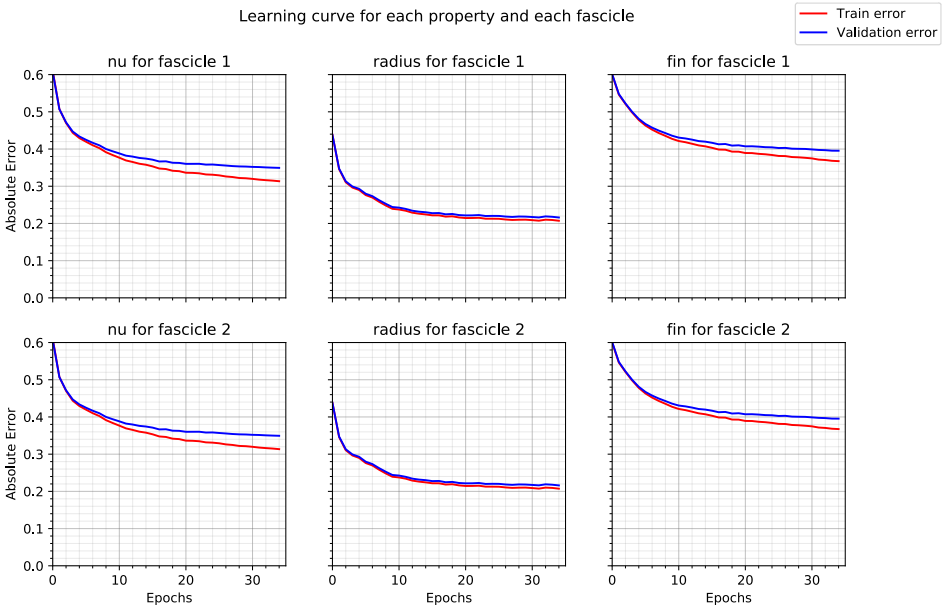

Figure 3.6: Learning Curves for six properties.

property	**nu**	**radius index**	**density index**
fascicle 1	0.189	$1.33 \cdot 10^{-6}$	0.207
fascicle 2	0.189	$1.23 \cdot 10^{-6}$	0.208

Table 3.2: One-sided 95% Confidence intervals. The intervals range from 0 to the values of the table since they are taken on the absolute error.

3.2.2.2 Architecture choice

As described in section 3.2.1.2, the chosen architecture for this network is based on the structure of the data by splitting it. For this network, the number of layers and hidden units chosen is detailed in table 3.3. The degrees of freedom in the network are the weights. Their number is computed by multiplying two by two the each layer with the next one (starting with the input size multiplied by the size of the first layer) and add everything together.

	layer	number of hidden units
Split NN	num_w_l1	200
	num_w_l2	600
	num_w_l3	200
	num_w_l4	50
	num_w_out	50
Degrees of freedom of split NN		$35 \cdot 10^4$
Final NN	num_f_l1	300
	num_f_l2	200
	num_f_l3	100
Degrees of freedom of final NN		$11 \cdot 10^4$
Total number of degrees of freedom		$46 \cdot 10^4$

Table 3.3: Number of hidden layers, hidden units and degrees of freedom of the different parts of the neural network

3.2.2.3 Parameter choice

Besides the architecture, some parameters also influence the training. To choose the best parameters possible, the network is trained with different values. Ideally a grid search on many values and parameters should be used, but as it takes a lot of time we decided to restrict it to the analysis of the three parameters that are thought to have the highest influence: the dropout, the learning rate and the batchsize [38]. Every training was performed using the complete training data set ($400,000$ samples) and during 35 epochs (i.e. updates). While one parameter was trained, the others were kept constant.

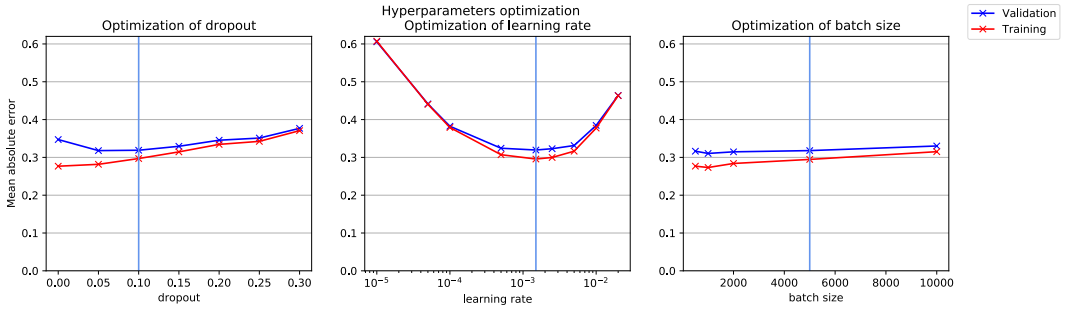

Figure 3.7: Optimization of three hyperparameters, the chosen values are indicated by the blue vertical lines.

Finally, looking at the validation curves of figure 3.7, the chosen values for these parameters are:

- `Dropout`: 0.1
- `Learning rate`: 0.0015
- `batchsize`: 5000

3.2.2.4 Influence of the size of the training set

Next the influence of the amount of data used was analysed. Indeed, neural networks are very sensitive to the amount of data as can be seen on figure 3.8. Here we see that for a small amount (less than $50,000$), the network is overfitting. We also notice that the improvement when going from $300,000$ to $400,000$ samples is marginal. It indicates that $400,000$ samples are enough for this analysis.

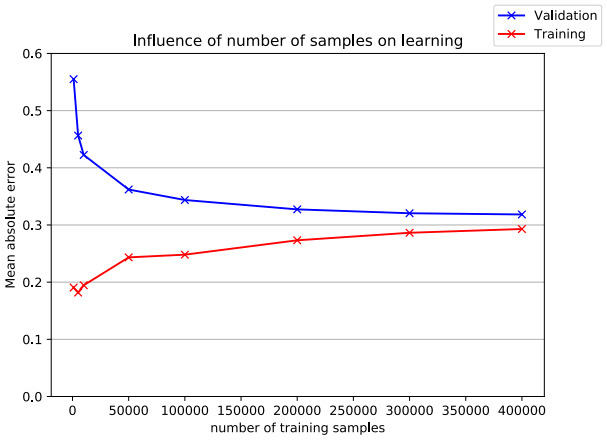

Figure 3.8: Influence of the number of samples on the performance of the network, showing that the chosen value of $400,000$ samples for the training is sufficient.

3.2.3 Advantages and Drawbacks

This method is inspired from the biophysics of the problem. First because it uses a pre-computed dictionary, which can be seen as a numerical twin of the brain. Secondly because the architecture of the deep learning network, that splits the weight vector depending on the number of fascicles in the voxel, maps the biological situation.

Since this method also uses a pre-computed dictionary of fingerprints, the orientations of the K fascicles in the voxel need once again to be known beforehand and thus to be estimated in a precomputational step. This step already induces some errors, as will be detailed in section 4.3.1.

Another drawback of this method is the difficulty to tune a deep learning network. Indeed the learning of the network will depend on many parameters, as described previously. These parameters highly influence the results and are not easy to tune.

The pre-computational time for this method is higher than for the reference method (Exhaustive dictionary search) because the deep learning network needs to be trained. But the training of the network needs to be done only once, then during the actual computation, only the forward pass is used, which is really fast. So the total prediction time for this method is the time to estimate the orientations, added to the time to solve the NNLS optimization problem and the time needed in the forward pass of the FFNN. This is a lot faster than the exhaustive search, as will be seen in chapter 4.

A remaining question and improvement clue for this method is how to take maximal advantage of the sparsity of the weights:

> *Is a deep learning network the most adapted tool for analysing sparse data and are there some networks that perform better than others on sparse data (like Recurrent NN)?*

3.3 Tree-based model

The idea of this next alternative method is to use a machine learning model that would learn from the data but that would still be interpretable. The solution to these requirements is a tree-based model. A decision tree is similar to the human decision-making process and is therefore easy to understand. Moreover, this type of model is very popular in machine learning and in biomedical engineering.

This section describes briefly the concepts of decision tree and of ensemble methods that combine them. For further information, please refer to [46] for decision trees, [47] for random forests and [48] for Gradient Boosting.

3.3.1 Description

Decision trees are sequential models that use a sequence of simple tests, each test compares a numeric attribute against a threshold value. They are build with nodes, branches and leaves. Each branch shows a decision and each leaf shows an output. They can be used for regression and can handle non-linear relationships quite well.

Decision trees mimic the human decision making process which gives them an advantage in terms of comprehensibility over "black-box" models, such as neural nets. The logical rules followed by a decision tree are much easier to interpret than the numeric weights of the connections between the nodes in a neural network. A single tree can be interpreted as a set of if-then rules and can be easily visualized in a tree plot where it is possible to interpret the splits [46].

But single decision trees are generally weak predictors and are sensitive to for example changes in the training data [49]. Successful approaches to generate stronger predictions combine multiple trees. These models are called ensemble models. The two main ensemble models are [50]:

1. Random Forest (RF) models, that build many decision trees in parallel.
2. Gradient boosting (GBoost) models, that build many decision trees sequentially.

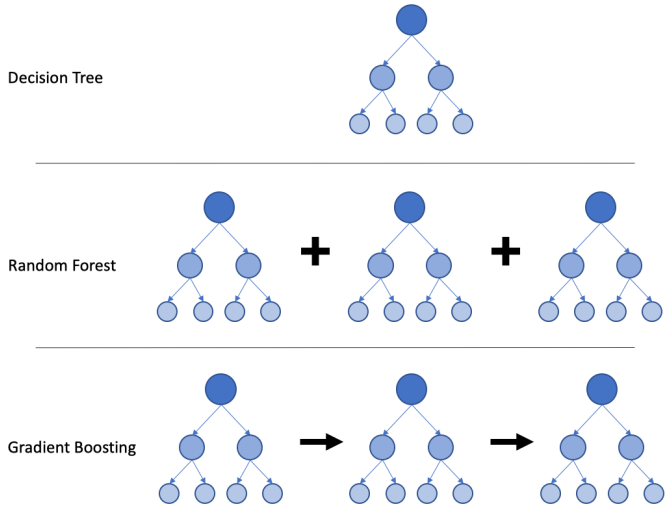

Figure 3.9: Schematic overview of the models using decision trees

3.3.1.1 Random Forest

Random forests (RF) are one of the best performing methods for constructing ensembles. They are an ensemble of many trees where each tree is trained separately on a bag of the data that is a random subset. This means that the trees are trained in parallel and that each tree does not depend on the others. The final prediction is an average of the predictions of all the trees [47].

To train a random forest model on our data, `RandomForestRegressor()` from `sklearn` is used [43]. The performance of this model is influenced by the *number of trees* that are built and by the size of each tree, which depends on the *maximum depth* allowed to construct them.

3.3.1.2 Gradient boosting

Like random forests, gradient boosting works with a set of trees. The difference is that the trees are not created in parallel but sequentially and so each tree depends on the previous one. The subsequent trees seek to predict how far off the original predictions were from the truth, by using residuals of the previous tree. In other words, the second tree no longer predicts the same target as the first one, but each subsequent tree slowly reduces the overall error [48, 50].

Gradient boosting has a very high predictive power but a lower interpretability. Compared to RF, it has more hyperparameters, so it is harder to tune. Gradient Boosting is also prone to overfitting the training data and is therefore not the best choice for data with a lot of noise, which can be the case in biomedical imaging [50].

To implement this model `XGBRegressor` from `XGBoost` was used [51]. As for random forests, Gradient Boosting depends on the *number of trees* that are built as well as their size (defined by *max depth*). Additionally, there is a *learning rate* which controls how subsequent trees are added together.

3.3.2 Fitting and Validation

As explained, the tree based models need to be trained on labeled data in order to make predictions. Throughout this section, Random Forest regressors and Gradient boosting regressors will be compared and analysed. The one that fits best to the problem of this work is then chosen to be used in chapter 4. First the chosen training data is justified and its influence is analysed. Next some key parameters are optimised.

3.3.2.1 Data used for the training

As in all machine learning techniques, the data on which the regressor is fitted plays an important role. For the following tests on random forests, we fix the values: `maxDepth`$= 12$ and `number_trees`$= 40$. Without fixing these parameters the models train without stopping (or after a really long time). For the gradient boosting method, no parameters where fixed so the default parameters are chosen by `sklearn`.

First, the number of samples on which the models are trained plays an important role. Theoretically, the more data is used, the best, but in practice a higher amount of data can increase the training time a lot. Figure 3.10 shows that the gradient boosting (GBoost) method is able to use more data at a reasonable time cost, whereas the training time for the random forest (RF) is very high already for $50,000$ samples.

Further, the machine learning models can be fitted on data with noise or on data without noise (i.e. the original pure signals before noise addition). The first one could help the model to extract only

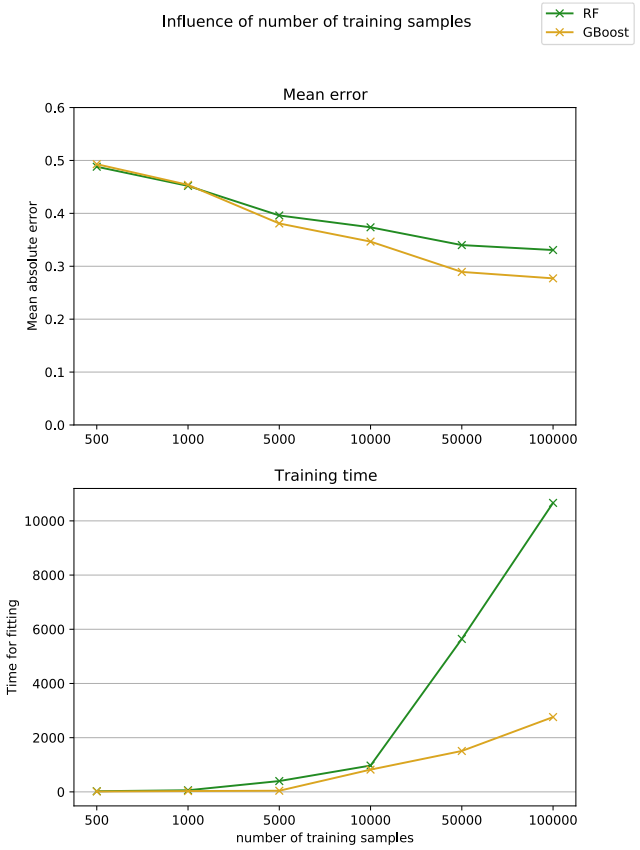

Figure 3.10: Random forest (RF) and Gradient Boosting (GBoost) models comparend based on the number of training samples. When this number increases, the error decreases but the training time increases rapidly, especially for RF.

the relation sought but is prone to overfitting. The second one could make the model more robust to noise, but information could be harder to extract.

Figure 3.11 shows the performances of two models for each technique, one (pink) trained on the noisy data set and the other one (green) trained on the data without noise. All the information about the data is described in section 3.1. Here we can see that for both RF and GBoost, the models trained on non noisy data perform well only on test data that is also non noisy. The gradient boosting model trained on non noisy data generalizes particularly poorly to noisy data, which confirms the theory that the gradient boosting technique is more prone to overfitting. Overall we can conclude from this graph that, in order to be robust to noise, the models should be trained on noisy data.

The choice was made to use the gradient boosting model for the rest of this work because it can use more data (and so extract more information) while keeping a reasonable training time. Indeed, for the sake of equality, the comparisons on Figure 3.11 is made on the same amount of data for RF and GBoost ($60,000$ samples), but when using $100,000$ samples or more, which is not possible for the RF (regarding the training time), GBoost improves its precision and performs better than the RF.

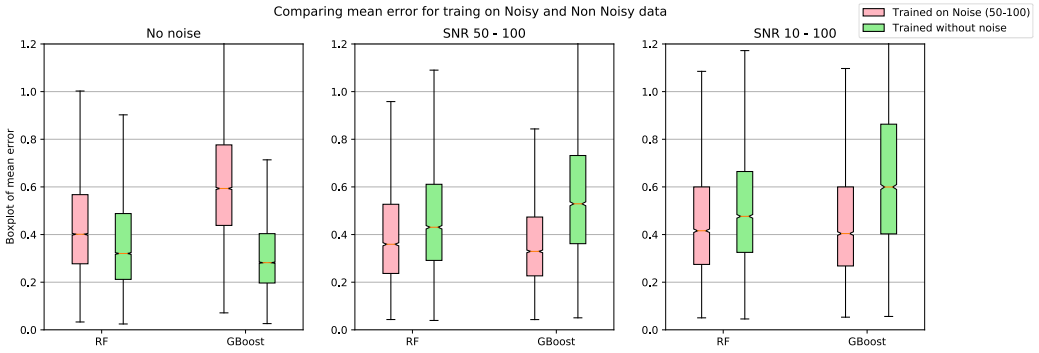

Figure 3.11: Performance of two regressors, one trained on noisy data and the other trained on non noisy data for the random forest model and the Gradient boosting model. The regressors are tested on data with different noise levels (training on only 60 000 training samples and testing on 30 000 samples)

3.3.2.2 Tuning of parameters

The next section focuses on the tuning of the main parameters of a XGBoost model, since this model was chosen rather than the RF model. In an ideal world, with infinite resources and where time is not an issue, a giant grid search with all the parameters together could be ran to find the optimal solution. In fact, it would be possible to do that with a really small dataset, but as the data grows bigger, training time grows too, and each step in the tuning process becomes more expensive.

Based on the graph of previous section (Figure 3.10) the number of training samples was chosen to be of size $400,000$ to match the training data set used for the other methods.

Since only a limited number of parameter choice possibilities can be tested, it is important to understand the role of the parameters, to chose the changing parameters well and to focus on the steps that are expected to impact the results the most.

Most data scientists see *number of trees*, *tree depth* and *the learning rate* as the most crucial parameters [52]. Here is a short description of these parameters as well as their default value in the python API reference [51].

- *Maximum depth* - `[default=6]`: Maximum depth of a tree. Increasing this value will make the model more complex, this can lead to better estimation, but also to overfitting
- *Learning rate* - `[default=0.3]`: Step size shrinkage used in update to prevent overfitting. After each boosting step, we can directly get the weights of new features, and the learning rate shrinks the feature weights to make the boosting process more conservative.
- *Number of estimators* - `[default=100]`: Number of gradient boosted trees. Equivalent to number of boosting rounds.

As for the other trained models of this work, the objective function is the MSE which is the default loss function for regression in `XGBoost`. The performance of the model is then assessed with the MAE, because this will show directly the difference between groundthruth and prediction.

The optimization of the parameters is shown on Figure 3.12. The default `XGBoost` parameters, indicated by the blue vertical line perform well and are therefore kept for the training of the final model (used in chapter 4).

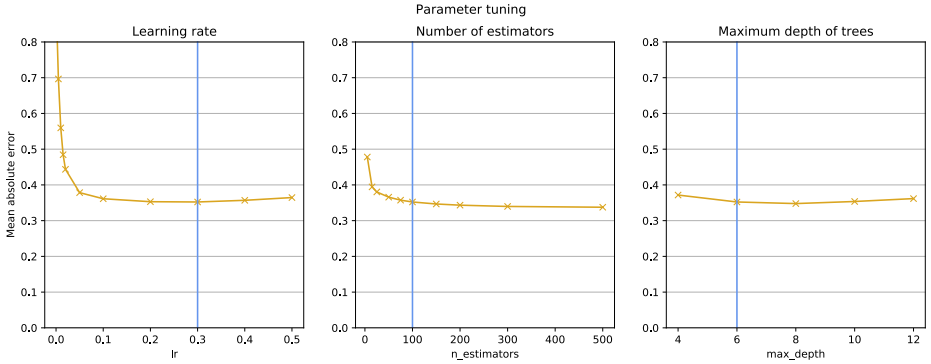

Figure 3.12: Tuning of parameters for the gradient boosting model. The default parameters (that are also the chosen ones) are indicated by the blue vertical line.

3.3.3 Advantages and drawbacks

The biggest advantage of this method is its efficiency to make the predictions while keeping a good precision; indeed it takes only around 1 second to estimate the properties for 10.000 samples.

`XGBoost` combined with the `MultipleOutputRegressor` [43] is easy to use and to tune as the default parameters are already very good for this problem.

Finally, compared to a neural network there is also less flexibility. For example there is no such choice as the architecture, a flexibility of neural networks that we used in section 3.2.1.2 to create a split NN based on the underlying biology. This means it is not possible to take advantage of the structure of the data and thus there is less room for improvement.

3.4 Deep Learning

The idea for the last alternative method is to apply a pure deep learning network. This model learns by itself with the feedback on a large amount of data, without care for the variables at play.

3.4.1 Description

The training was performed with the open-source `PyTorch` library in Python 3 [53]. The used optimizer is AdaGrad optimizer, the activation function is the $ReLU$ function and the loss function is the mean squared error (MSE) on the 6 output targets.

The chosen architecture of FFNN (Feed forward Neural Network) is a fully connected MLP (Multi layer perceptron). In this structure, all the units of one layer are linked to all the units of next layer. For more information about how the model works, see section 1.5.

3.4.2 Training

As described in section 1.5 and 3.2, the performance of the network is dependent on the training and on the structure of the network. Here again the NN was trained with the Adam optimizer (described in section 3.2.1.2).

3.4.2.1 Learning curves

Like in section 3.2, we can start to analyze the training by looking at the learning curves. From the graph on Figure 3.13, we can first deduct that the network learns quite well: it reaches a validation error of 0.261 (compared to the 0.833 baseline error). Further, Figure 3.14 shows that the same properties are again learned in the same way for both fascicles because the curves of the first and second row follow the same trend.

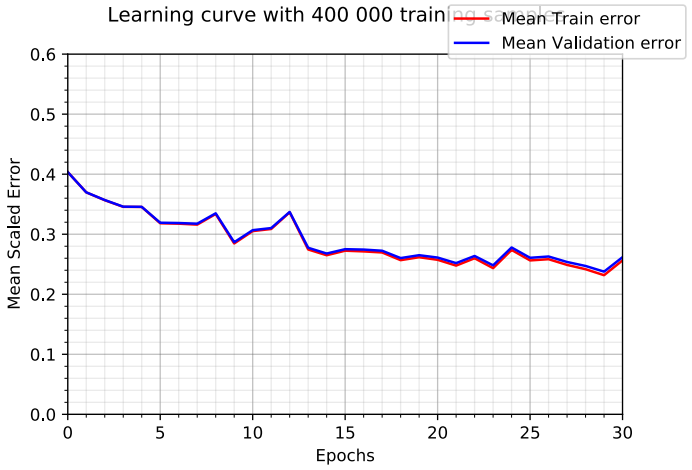

Figure 3.13: Learning Curve of the NN.

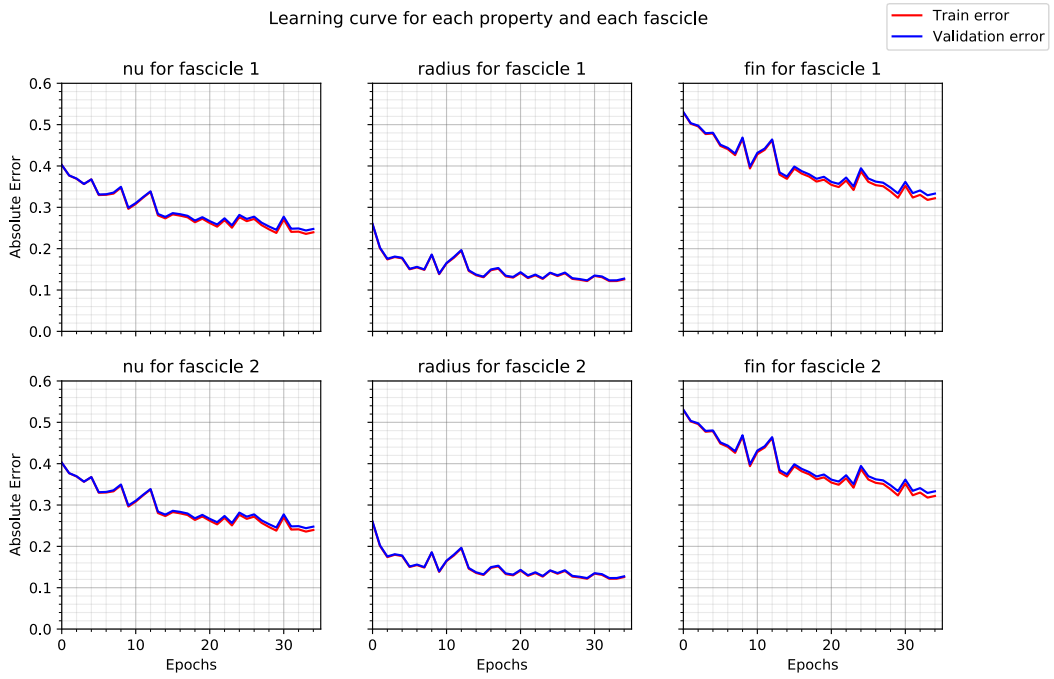

Figure 3.14: Detail of learning curves for the different properties.

3.4.2.2 Architecture

The built network contains 5 hidden layers for a total of around $3 \cdot 10^6$ degrees of freedom (computed by multiplying two by two the number of hidden units of each layer and adding it together). The details are shown in table 3.4.

layer	number of hidden units
hidden layer 1	300
hidden layer 2	800
hidden layer 3	1600
hidden layer 4	800
hidden layer 5	100
Degrees of freedom	$3 \cdot 10^6$

Table 3.4: Number of hidden layers, hidden units and degrees of freedom of the neural network

Enlarging the network can improve its performance, but only if enough data is available. If this is not the case, increasing the size of the network will have no impact on the error. After tests were performed on different numbers of layers and of hidden units in every layer, it has been noticed that these numbers do not have a lot of influence on the performance of the network, as long as the total number of degrees of freedom is large enough. In other words, the network just needs enough degrees of freedom to be able to learn from the provided data.

3.4.2.3 Parameters

Like the network used after the NNLS (section 3.2.1.2), this neural network is influenced by the same hyperparameters, from which the most important ones are the *dropout*, the *learning rate* and the *batchsize*. The optimization of these three key hyperparameters is shown on Figure 3.15.

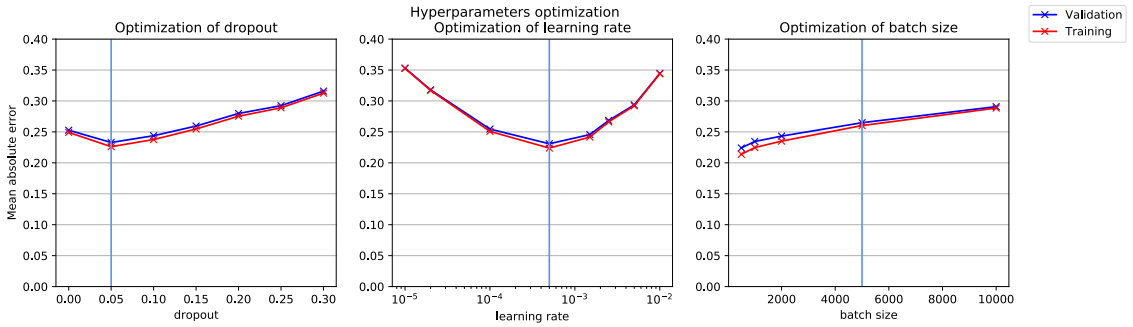

Figure 3.15: Optimization of three hyperparameters. The blue vertical line indicates the chosen value.

Based on the graph of Figure 3.15, the following values have been chosen:

- `Dropout`: 0.05
- `Learning rate`: 0.0005
- `batchsize`: 5000

3.4.2.4 Influence of data

To train the network, the data is probably the most important component. First the amount of data available has a high influence on the precision, but after a certain point, increasing the number of samples does not help to improve the precision anymore. At that point it is possible that the network needs to be made larger or that it cannot extract anymore features from the data. From the graph on Figure 3.16, we can conclude that the number of $400,000$ samples used to train the network is sufficient.

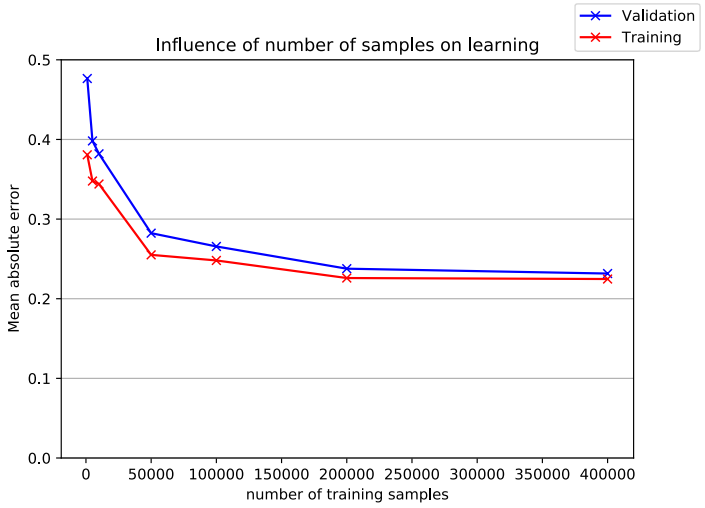

Figure 3.16: Influence of number of samples on the performance of the trained network. The chosen number of $400,000$ training samples seems to be sufficient.

A second important point is the noise on the data that will be used for the training. Two opposite hypotheses prior to the analysis are that either the model will perform better on noisy data when trained on noisy data, or it will be able to extract the features best when trained on pure data (no noise). The graph of Figure 3.17 shows this analysis. The network trained without noise has a

very high precision on data that has also no noise, but this precision decreases drastically (i.e. the error increases) when the network is used on noisy data (even on data with a very low noise level, SNR 80-100). More generally this is an instance of a network failing when the data distribution changes (the noiseless distribution is a delta distribution, while the noisy distribution is the Rician distribution). The other network was trained on noisy data with a low noise level (again SNR 80-100) and generalizes better to data with more noise.

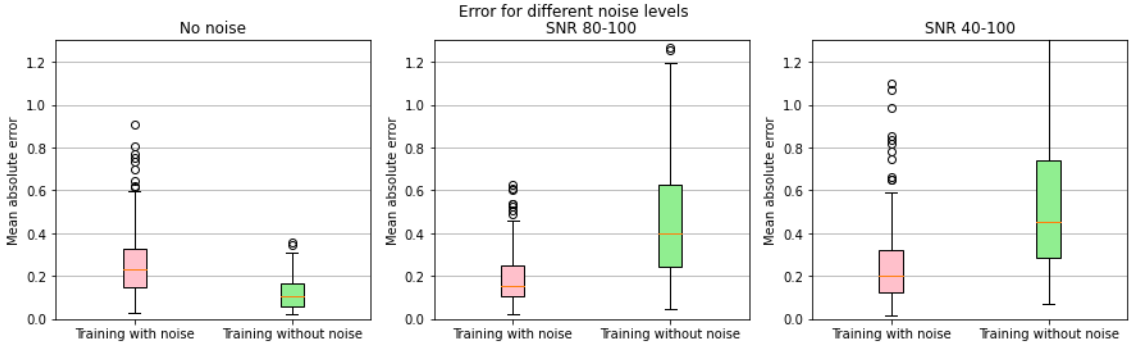

Figure 3.17: Comparing the performances of a network trained on data with a low noise level (SNR 80-100) and a network trained on pure (non noisy) data. The first network generalizes more to higher-noise data.

3.4.3 Advantages and drawbacks

Like for the gradient boosting model, the main advantage of the neural network is its efficiency: once it is trained, it can estimate the properties of thousands of samples in less than a second. Moreover, this efficient model also preserves the desired precision.

The most important part of a deep learning network is the data it receives for the training. Hence the need to have enough samples for the learning phase. Second, the quality of the data needs to be good enough so the network can detect patterns or extract features from it.

With this pure deep learning technique less assumptions are needed compared to the exhaustive dictionary search or the NNLS followed by deep learning:

- the orientations do not have to be estimated beforehand
- the superposition principle (section 1.3.7) is not assumed

On the other hand, the model is more sensitive to the structure of the Dw-MRI signals and does not generalize to changes in the data structure. It is also more sensitive to noise as it does not benefit from the NNLS optimization step working as a buffer.

3.4.4 Perspective: Taking advantage of the structure of the data

To go a step further, the black box can be improved by taking advantage of the structure of the data and changing the network structure accordingly. This injection of prior knowledge could make the network learn faster and with less data [54, 55].

For this improvement, we need to better understand the Dw-MRI signal. First, the structure of this signal depends on the used protocol. With the HCP (Human Connectome Project), the vector can be divided in 4 shells each corresponding to 1 b_{value} (see section 3.1). To take advantage of this, the Dw-MRI input vector could be divided into parts, corresponding to the shells. These parts would then be independently given as input to the same or to different networks. The last step

is combining the results in a final network. This technique is close to the one used in the split network of the second method (3.2.1.2), but the difference is that it would be built based on the structure of the data (and the acquisition protocol), which is not inspired by the biology of the problem like it is in method 2.

Another approach would be to use the fact that the signals are obtained by taking gradients on the sphere (see section 1.3.6). A **Spherical Neural Network** could take advantage of this property of the data [56, 57]. For this method the Dw-MRI signals are interpolated by spherical harmonics. It is then the coefficients of these interpollations that are given to a neural network. This change also enables the model to generalize easier to changes in the Dw-MRI signals (like signals including missing elements).

Chapter 4

Comparison of results and discussion

This chapter compares the exhaustive dictionary search to the three alternative solutions proposed and analyzed in chapter 3. These three methods use machine learning techniques to train and extract information from synthetically generated data. The goal is to gain efficiency while keeping a good prediction. Further it is also important to look at how the models generalize to changes in the problem statement and in the type of data structure or data acquisition.

4.1 Test data

The data used for the comparison tests of this section has been generated independently from the data used in chapter 3 for training and validation. It was generated by choosing a specific configuration (unlike the training and validation data that are generated based on random configurations).

This data set includes $15,000$ samples with varying Noise levels (i.e. SNR-values) and varying volume fractions for the first fascicle (i.e. ν_1-values). The chosen values are:

- $\nu_1 = [0.1, 0.2, 0.3, 0.4, 0.5]$: fascicle 1 will always be the smallest fascicle, a small value for ν_1 means there is a big imbalance between the two fascicles.
- SNR between $[(10-100), (30-100), (50-100)]$: the higher the SNR, the better the quality of the signal.

For each combination of ν_1 and SNR-value, 1000 samples were generated, which makes a total of 15000 samples. The other data generation parameters were kept as described in section 3.1. This specific configuration allows us to analyse the influence of the data properties on the performance of the trained models.

4.2 Efficiency

The primary objective of this work is to find alternative methods that would be more efficient than the exhaustive dictionary search. For this reason the first element of the comparison of the methods regards the prediction time. As explained in the description of the methods, the use of the dictionaries in the exhaustive dictionary search (ES) and in the NNLS solving requires a precomputational phase to estimate the axons orientations. This time comparison is showed in Table 4.1. All the tests were performed on a *1.6 GHz Dual-Core Intel Core i5* processor.

	Orientation estimation	Dictionary rotation	solving NNLS	Final prediction	Total prediction	Acceleration factor
Exhaustive search	1020.34	880.20	-	17865.11 (5h)	18745.65 (5h 15min)	1
NNLS + DL	1020.34	880.20	276.75	1.23	2178.52 (36 min)	8.6
Gradient Boosting	-	-	-	1.70	1.70	$1.1 \cdot 10^3$
Pure DL	-	-	-	1.53	1.53	$1.2 \cdot 10^3$

Table 4.1: Time $[s]$ needed for the four methods to make a prediction of the $15,000$ test samples and acceleration factor of each method compared to the exhaustive search (ES). The difference between ES and the alternative methods is striking.

From this table it is clear that the method with NNLS followed by Deep Learning already represents a very high improvement compared to the ES. Gradient boosting and pure deep learning, which only use machine learning algorithms, improve the efficiency even more.

Table 4.1 also highlights the importance of alternative efficient methods to the dictionary search. Indeed, the number of white matter voxels of a brain on a present-day high-resolution MRI scan is around $230,000$. To treat all these voxels with a dictionary search would take around 84 hours or 3 days and a half. With the NNLS method this takes 9 hours and less then a minute with the gradient boosting and pure deep learning models.

4.3 Precision

The second element of the comparison is of course the evaluation of the precision of the methods. Different steps in the prediction can lead to an error. The final prediction step (i.e. the dictionary search for the exhaustive method and the forward pass of a trained machine learning model for the alternative methods) brings some error, but for the exhaustive search and the NNLS, the orientation estimation also adds a non negligible error.

4.3.1 Orientation estimation

To create the dictionary (used for the ES and for the NNLS optimization), a canonical dictionary of fingerprints has to be rotated using the orientation angles of the fascicles. To do this, the angles have to be estimated based on the Dw-MRI signal. This step is already prone to some errors as shown on Figure 4.1. The histogram of the mean error for each sample shows that although half of the samples have a low angular error, the other half is prone to significant errors.

From the second part of Figure 4.1 we can infer that the angular error is nearly independent of the noise level, but is strongly linked to the volume fraction of the fascicles. It is clear that the orientations are harder to estimate when the fascicle has a lower volume fraction (the error for fascicle 1 decreases when ν_1 increases and the error for fascicle 2 increases when ν_1 increases). So the orientation estimation for the bigger fascicle (which is always the second one in our case) is easier then for the smaller fascicle. On the other hand, the influence of the noise is hardly visible.

Finally, to clearly see the influence of the orientation error on the final error, the performances of the exhaustive search using the estimated orientations and the true orientations are compared (respectively black and grey curves). The true orientations are the ones used for the data generation, they are not available for real data.

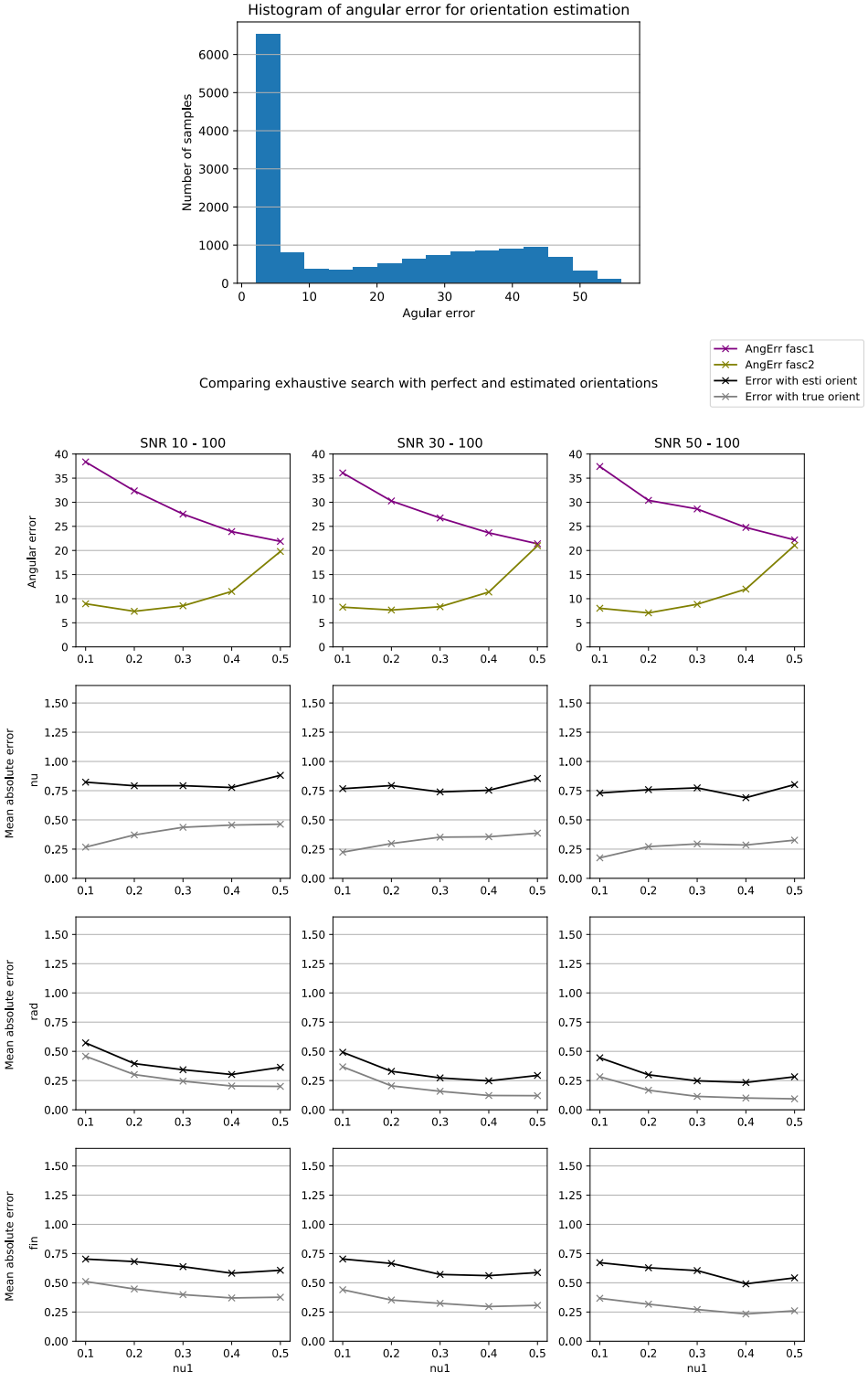

Figure 4.1: Angular Error for the orientation estimation and the effect it has on the predictions with the exhaustive search. (A) Histogram of the mean angular error; half of the samples have a relatively small angular error while the other half is prone to a significant error. (B) Influence of angular error on precision with exhaustive search through the comparison of this method used with estimated orientations and with the true orientations.

4.3.2 Final prediction step

Figure 4.2 shows the boxplots of the absolute mean error for the four methods. Two boxplots are for the exhaustive search: one with the orientation estimation (ES for Exhaustive Search) and the other one without orientation estimation (ES*). The faster models were trained on a noise level with SNR 50-100, but as we can see they all generalize well to higher noise levels, the differences with SNR 30-100 are particularly very small. A part of the error of the method using NNLS (blue) and of the exhaustive search (black) is due to the estimation of the orientations. This figure shows that the pure Deep Learning (DL) method is the alternative method with the smallest error in all cases. Moreover its error is quite close to the ideal error obtained with the exhaustive search without orientation estimation (ES* in grey).

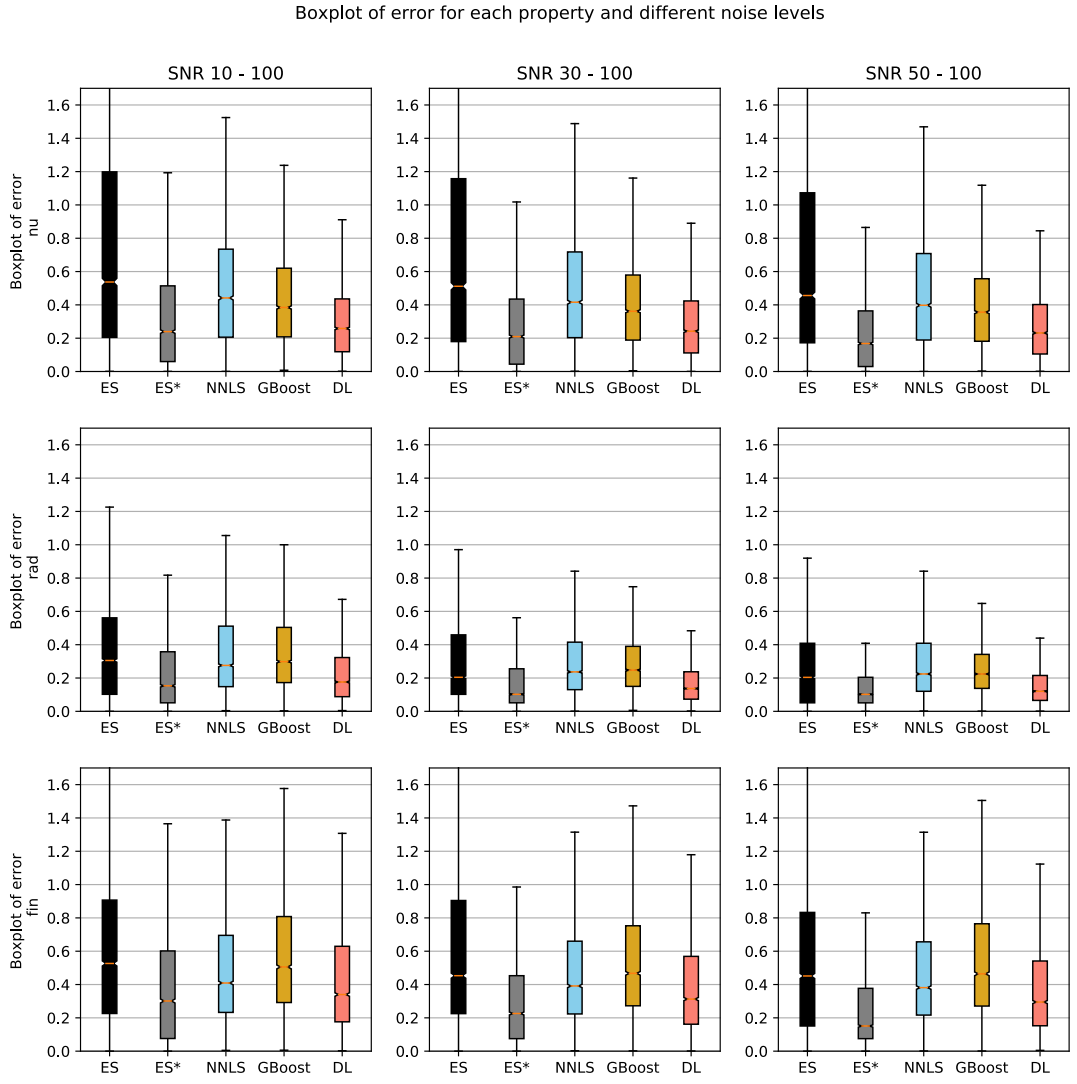

Figure 4.2: Comparing the precision of the Exhaustive search - with and without orientation estimation (respectively ES and ES) - and of the three alternative methods through the boxplots of the absolute errors (averaged over the two fascicles for each property). The pure deep learning method (DL) is the alternative method with the best performance.*

Next the influence of the volume fractions (ν or nu) on the performance of the networks is assessed. As a reminder, the volume fraction of the two fascicles add up to 1 ($\nu_1 + \nu_2 = 1$). For this

reason we only have ν_1 that varies from 0.1 to 0.5.

Figure 4.3 shows the absolute error for each property and noise level. On these graphs the errors are descaled and expressed in terms of the "real" units and values of the properties. The volume fractions clearly have an influence on the performance of the methods.

We notice that all models have some trouble when one fascicle is very small ($\nu_1 = 0.1$) and perform better when the fascicles are balanced. This is coherent with the physics of the problem, indeed it is more difficult to notice the influence of a small fascicle on the Dw-MRI data in the same way that it is more difficult to detect a small fascicle on an image. Further, the influence of the noise is small but noticeable.

Again the pure deep learning method (DL) performs better than the other alternative methods and is very close to the performance of the ideal exhaustive search (ES*), sometimes even surpassing it.

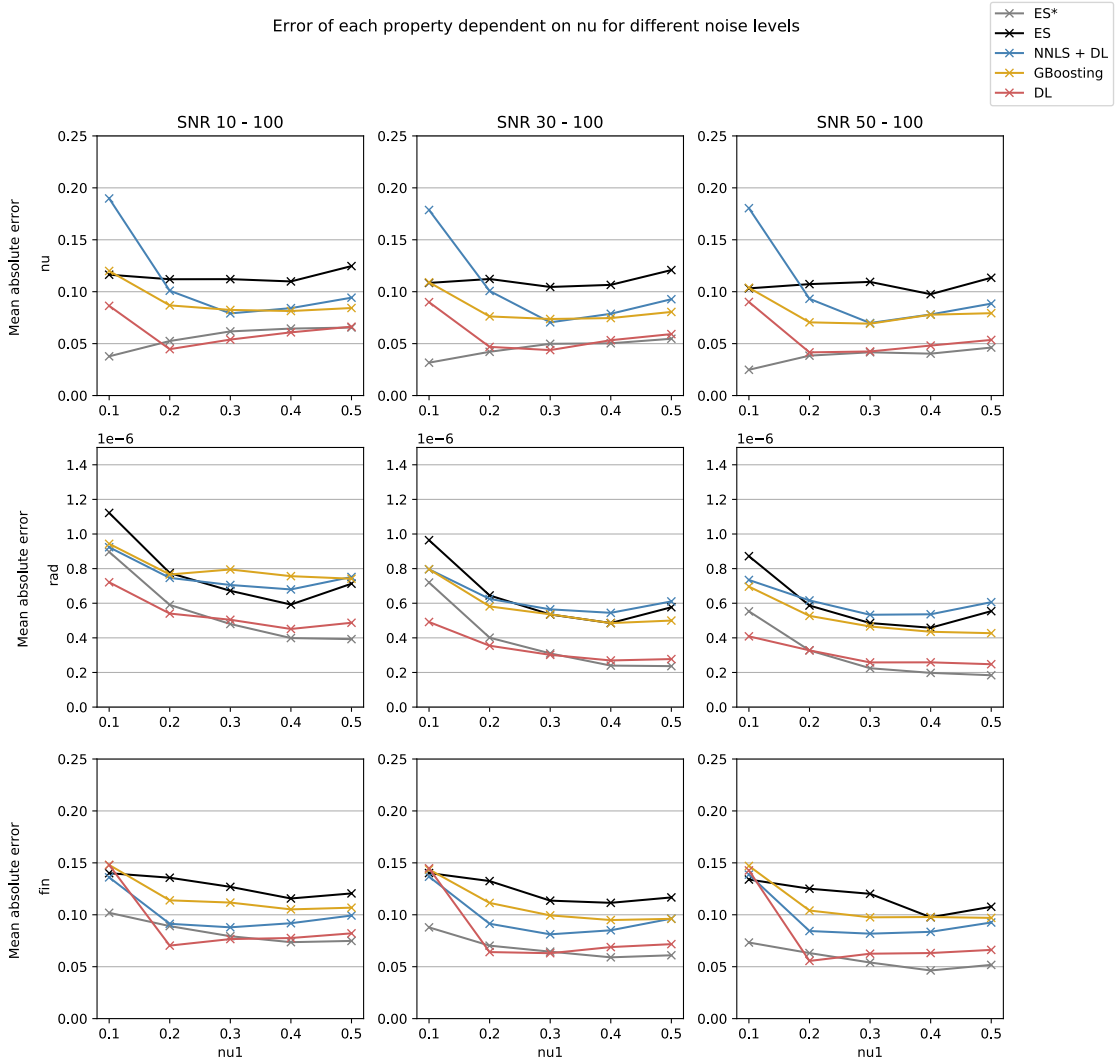

Figure 4.3: Absolute error (average over the two fascicles) for the different models over a range of values for the volume fractions of the first fascicle (nu_1). The volume fractions of the fascicles clearly have an influence on the performances of the different methods.

Finally, to further understand these results, we can analyze the error for both fascicles independently. This is done in Figure 4.4. The fact that the graphs of the errors for the estimation of ν are mirrored, shows that all the methods use or learn the relation $\nu_1 + \nu_2 = 1$. For the other two properties, we see that the error decreases when the volume fraction of a fascicle increases which confirms the trend shown in previous figure: the estimation is easier when the fascicle is larger.

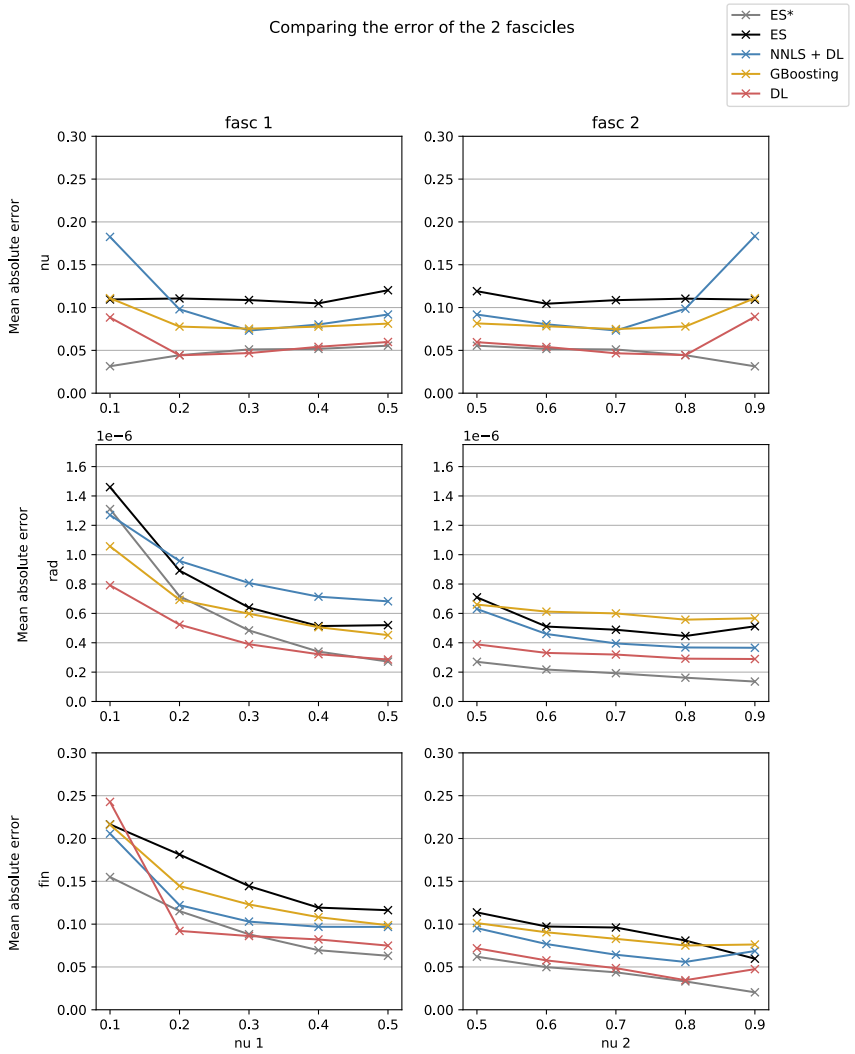

Figure 4.4: Comparing the estimation for the two fascicles using the absolute error on the true values. The volume fractions (first row) are better estimated when the two fascicles have the same size. The error for the radius (second row) and the density index (third row) of a fascicle decreases when its volume fraction increases.

.

4.4 Generalization and discussion

From the two previous sections, the pure deep learning model seems the best alternative to the exhaustive dictionary search. Indeed it has a good precision, can be easily trained and above all it is very efficient. Moreover as said in section 3.4.4, this method has room for improvement by taking advantage of the structure of the data and changing the architecture of the network accordingly. But a last important point to look at, is how these methods generalize to a change in problem or in data structure. The possibilities to generalize or not for each method are summarized in table 4.5 together with the precision, prediction time and advantages and drawbacks of the methods.

First, the **hypotheses and objectives** regarding the problem described in chapter 2 could change. We could for example want to consider three or more different fascicles in the voxel ($K > 2$). To do this, a new data set has to be generated accordingly and the alternative methods will need to be retrained on this new data set. The training takes only some hours, but generating a new synthetic data-set can take several days and a lot of memory. The efficiency of the Gradient Boosting model and of the pure deep learning model should not change significantly. The prediction time of the method using NNLS will increase but in a reasonable way (linearly with the size of the total dictionary) [1]. On the other hand, the prediction time with the exhaustive search will increase significantly.

We could also want to estimate other properties. For this change, the models using machine learning and deep learning have once again to be retrained on the new selection of properties. For the exhaustive search on the other hand, once the best combination is found, we have access to all the properties of the fascicles.

A second attention point is the influence of a **change in the data**. Indeed the protocol used to generate the Dw-MRI image is not unique and could change from one MRI machine to another. In this case, the models using pure deep learning and gradient boosting will have to be completely retrained on new data. As opposed to this, the method that first solves the NNLS problem has showed to be robust to a change in protocol [1] and so should not especially be retrained. The exhaustive search is quite independent of the chosen protocol because the dictionaries have to be all regenerated for each voxel anyway.

The Dw-MRI dataset could also lack some measurements. If the missing elements are known, we can just replace them by a chosen value and verify that the influence on the results is negligible. If we don't know which values are missing, the input vector will have a different size and the two last methods will be completely unusable. A solution to this would be to interpolate the Dw-MRI signals and train the network on the coefficients of the interpolation (mentionned in section 3.4.4). For the NNLS on the other hand, the adaptation is easy since deleting the corresponding rows in the dictionary is sufficient to find the weights and then use the neural network as usual.

Further, the objective is that these alternative models would generalize to **experimental data**. As seen in [1], models trained on synthetically generated data do not always perform well on real data. The gap between the performance on real data and on synthetic test data means that the real data is too noisy or that this synthetic data is not yet close enough to the reality. This gap also emphasizes the dependence of the methods on the synthetic data used for training.

Finally the approach of this master thesis could be generalized to **other problems** in any domain of application using a dictionary search. The exhaustive dictionary search could thus be replaced by a neural network provided that a forward function is available to generate enough data to train this network.

		Exhaustive search	NNLS + Deep Learning	Gradient Boosting	Deep Learning
Method & Performance	Mean precision (test data set)	0.580 (0.297 with true orientations)	0.443	0.407	0.321
	Total prediction time (test data set)	5h 15min	36 min	1.7 sec	1.53 sec
	Steps of computation	- Orientation estimation - Dictionary creation - Dictionary search	- Orientation estimation - Dictionary creation	- Forward pass	- Forward pass
	precomputation	/	Training of network (+/-2h)	Training of tree-based model (+/-3h)	Training of network (+/-2h)
Generalization	More fascicles (& larger problem size in general)	Prediction time **explodes**	Prediction time increases - New data gen - New training	Constant prediction time - New **data gen** - New training	Constant prediction time - New **data gen** - New training
	Change in properties to estimate	Easy	Easy but new training	Easy but new training	Easy but new training
	Different protocol	OK (easy)	OK (easy)	- New **data gen** - New training	- New **data gen** - New training
	Change in data structure	Robust	+/- Robust	Not robus	Not robust
	Real data	OK	Not good	Not good	Not good
Note		Not possible to use for larger problem sizes → needs an alternative		Not possible to improve	Room for improvement by changing the architecture and taking advantage of the **structure of the data**

Figure 4.5: Comparison of the advantages and drawbacks of the Exhaustive method and of the three alternative solutions proposed in this work

Conclusion and perspectives

The problem was the estimation of microstructural properties of white matter tissues. A general framework has already been proposed in previous work and is based on Monte Carlo simulations, recognized as a reference standard in diffusion-weighted magnetic resonance imaging (Dw-MRI). This approach uses a slow and costly dictionary matching that scales very poorly with an increasing number of parameters, which makes it unusable for too complex models. Hence the need for more efficient alternatives.

This work presents alternative solutions based on machine learning techniques. They increase the efficiency while preserving an acceptable precision. The machine learning models are trained on data that is synthetically generated through Monte Carlo simulations. The described methods span from a biology oriented model to a completely black-box model. With this shift we leave the precomputational phases and the interpretational work step by step to the computer and hope that, with an infinite amount of data, it will do a better work.

The first method described combines an optimization problem, that uses the dictionary of the exhaustive search in a more efficient way, and a neural network with an architecture inspired from the biology. Next a gradient boosting model, based on decision-trees, is used. Each decision tree can be interpreted even if it is not always obvious. Finally a pure neural network is constructed.

The alternative solutions perform quite well, especially the pure deep learning network that has a good precision and shortens the prediction time for a complete MRI scan from several days to less then a minute. The comparison of the models investigated the benefits and drawbacks of this full, end-to-end deep learning solution (compared to solutions with feature extraction stages and decision-tree based models). The analysis also brought to light interesting findings about the influence of the data (quantity, noisy or not) on the performance of the models.

Despite the good results on the synthetic data, it is essential to also look at the generalization of the methods. Indeed a Dw-MRI signal can be based on another protocol, have a high noise or even have missing elements. we could also want to slightly change the model simplifications and hypothesis. The different solutions generalize more to some variations and less to others, but overall it is very challenging to have a generic method that would be robust to all variations. Moreover, as seen in other work, end-to-end deep learning networks trained on synthetic data do not perform well on real human data. This highlights the influence of the synthetic aspect of the data and the importance of its quality to ensure generalization to real data.

Perspectives To go even further than the black-box approach, deep learning networks with complex architectures could be constructed to take advantage of the structure of the data. Another perspective is that the pure deep learning approach gives really good results on data with low noise. In the future, the quality of scanners will improve, which makes deep learning very promising. Finally, the approach developed in this thesis can be applied to more general fingerprinting problems in any domains. The costly dictionary matching could be replaced by an efficient neural network that is trained on synthetically generated data.

Overall, the efficiency of deep learning models will enable larger-scale studies and this way advance our knowledge of psychiatric and neurological disorders.

Bibliography

[1] Rensonnet, G. (2019). *In vivo diffusion magnetic resonance imaging of the white matter microstructure from dictionaries generated by Monte Carlo simulations: development and validation*. (Thesis). EPFL.

[2] Deuschl, G., Beghi, E., Fazekas, F., Varga, T., Christoforidi, K. A., Sipido, E., ... & Feigin, V. L. (2020). The burden of neurological diseases in Europe: an analysis for the Global Burden of Disease Study 2017. *The Lancet Public Health*, 5(10), e551-e567.

[3] Assaf, Y., Blumenfeld-Katzir, T., Yovel, Y., & Basser, P. J. (2008). AxCaliber: a method for measuring axon diameter distribution from diffusion MRI. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 59(6), 1347-1354.

[4] Alexander, D. C., Hubbard, P. L., Hall, M. G., Moore, E. A., Ptito, M., Parker, G. J., & Dyrby, T. B. (2010). Orientationally invariant indices of axon diameter and density from diffusion MRI. *Neuroimage*, 52(4), 1374-1389.

[5] Zhang, H., & Alexander, D. C. (2010, September). Axon diameter mapping in the presence of orientation dispersion with diffusion MRI. *In International Conference on Medical Image Computing and Computer-Assisted Intervention* (pp. 640-647). Springer, Berlin, Heidelberg.

[6] Alexander, D. C. (2008). A general framework for experiment design in diffusion MRI and its application in measuring direct tissue-microstructure features. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 60(2), 439-448.

[7] Zhang, H., Schneider, T., Wheeler-Kingshott, C. A., & Alexander, D. C. (2012). NODDI: practical in vivo neurite orientation dispersion and density imaging of the human brain. *Neuroimage*, 61(4), 1000-1016.

[8] Rensonnet, G., Scherrer, B., Girard, G., Jankovski, A., Warfield, S. K., Macq, B., ... & Taquet, M. (2019). Towards microstructure fingerprinting: Estimation of tissue properties from a dictionary of Monte Carlo diffusion MRI simulations. *NeuroImage*, 184, 964-980.

[9] Ma, D., Gulani, V., Seiberlich, N., Liu, K., Sunshine, J. L., Duerk, J. L., & Griswold, M. A. (2013). Magnetic resonance fingerprinting. *Nature*, 495(7440), 187-192.

[10] Akkus, Z., Galimzianova, A., Hoogi, A., Rubin, D. L., & Erickson, B. J. (2017). Deep learning for brain MRI segmentation: state of the art and future directions. *Journal of digital imaging*, 30(4), 449-459.

[11] Stovall, G., Neurons synapse pictures, `https://pulpbits.net/wp-content/uploads/2013/12/neurons-synapse-pictures.jpg`, [accessed 2021-06-01].

[12] Bonilha, L., Gleichgerrcht, E., Nesland, T., Rorden, C., & Fridriksson, J. (2015). Gray matter axonal connectivity maps. *Frontiers in psychiatry*, 6, 35.

[13] Draganski, B., Gaser, C., Busch, V., Schuierer, G., Bogdahn, U., & May, A. (2004). Changes in grey matter induced by training. *Nature*, 427(6972), 311-312.

[14] Behrens, T. E. J., & Johansen-Berg, H. (2005). Relating connectional architecture to grey matter function using diffusion imaging. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 360(1457), 903-911.

[15] Wozniak, J. R., & Lim, K. O. (2006). Advances in white matter imaging: a review of in vivo magnetic resonance methodologies and their applicability to the study of development and aging. *Neuroscience & Biobehavioral Reviews*, 30(6), 762-774.

[16] Jones, D. K., Knösche, T. R., & Turner, R. (2013). White matter integrity, fiber count, and other fallacies: the do's and don'ts of diffusion MRI. *Neuroimage*, 73, 239-254.

[17] Schwan cell, no labels. `https://www.pngkey.com`. [accessed 2021-06-10].

[18] Avram, Alexandru. (2011). *Diffusion Tensor Imaging of Myelin Water*. (Ph.D. Thesis, National Institute of Health).

[19] Price, W. S. (1997). Pulsed-field gradient nuclear magnetic resonance as a tool for studying translational diffusion: Part 1. Basic theory. *Concepts in Magnetic Resonance: An Educational Journal*, 9(5), 299-336.

[20] Grover, V. P., Tognarelli, J. M., Crossey, M. M., Cox, I. J., Taylor-Robinson, S. D., & McPhail, M. J. (2015). Magnetic resonance imaging: principles and techniques: lessons for clinicians. *Journal of clinical and experimental hepatology*, 5(3), 246-255.

[21] Scarpati, J. (2021). Radio frequency (RF, rf). `https://searchnetworking.techtarget.com/definition/radio-frequency`, [accessed 2021-06-09].

[22] Cavarsan, C., What are the differences among EEG, MRI and fMRI?, `https://www.brainlatam.com`, [accessed 2021-04-20].

[23] Mukherjee, P., Berman, J. I., Chung, S. W., Hess, C. P., & Henry, R. G. (2008). Diffusion tensor MR imaging and fiber tractography: theoretic underpinnings. *American journal of neuroradiology*, 29(4), 632-641.

[24] Drobnjak, I., Zhang, H., Ianuş, A., Kaden, E., & Alexander, D. C. (2016). PGSE, OGSE, and sensitivity to axon diameter in diffusion MRI: insight from a simulation study. *Magnetic resonance in medicine*, 75(2), 688-700.

[25] Diffusion NMR, `http://chem.ch.huji.ac.il/nmr/techniques/other/diff/diff.html`, [accessed 2021-03-10].

[26] Huisman, T. A. G. M. (2010). Diffusion-weighted and diffusion tensor imaging of the brain, made easy. *Cancer Imaging, 10*(1A), S163.

[27] Rensonnet, G., Scherrer, B., Warfield, S. K., Macq, B., & Taquet, M. (2018). Assessing the validity of the approximation of diffusion-weighted-MRI signals from crossing fascicles by sums of signals from single fascicles. *Magnetic resonance in medicine*, 79(4), 2332-2345.

[28] Ginsburger, K., Poupon, F., Beaujoin, J., Estournet, D., Matuschke, F., Mangin, J. F., ... & Poupon, C. (2018). Improving the realism of white matter numerical phantoms: a step toward

a better understanding of the influence of structural disorders in diffusion MRI. *Frontiers in physics*, 6, 12.

[29] Ishimwe, A. N., Benoit, P., Rensonnet, G., & Lee, P. J. (2017). The influence of spherical pores within the white matter on Diffusion-Weighted MRI signals: a numerical study using Monte-Carlo simulations.

[30] Nielsen, M. A. (2015). *Neural networks and deep learning* (Vol. 25). San Francisco, CA: Determination press.

[31] Kurenkov, A. A Brief History of Neural Nets and Deep Learning, `https://www.skynettoday.com/overviews/neural-net-history`, [accessed on 2021-05-29].

[32] Brownlee, J., How to Choose an Activation Function for Deep Learning, `https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/`, [accessed 2021-05-29].

[33] Fallon, E., Murphy, L., Murphy, J., & Miro-Muntean, G. (2012). FRAME—Fixed route adapted media streaming enhanced handover algorithm. *IEEE transactions on broadcasting*, 59(1), 96-115.

[34] Torre, G. (2017), The Brain's Building Blocks: Of Protons and Voxels, `https://knowingneurons.com/2017/09/27/mri-voxels/`, [accessed 2021-05-29].

[35] Tournier, J. D., Calamante, F., & Connelly, A. (2007). Robust determination of the fibre orientation distribution in diffusion MRI: non-negativity constrained super-resolved spherical deconvolution. *Neuroimage*, 35(4), 1459-1472.

[36] DIPY - Diffusion Imaging In Python, `https://dipy.org/`, [accessed 2021-05-26].

[37] Canales-Rodríguez, E. J., Legarreta, J. H., Pizzolato, M., Rensonnet, G., Girard, G., Rafael-Patino, J., ... & Daducci, A. (2019). Sparse wars: A survey and comparative study of spherical deconvolution algorithms for diffusion MRI. *NeuroImage*, 184, 140-160.

[38] Lippmann, R. (1987). An introduction to computing with neural nets. *IEEE Assp magazine*, 4(2), 4-22.

[39] Setsompop, K., Kimmlingen, R., Eberlein, E., Witzel, T., Cohen-Adad, J., McNab, J. A., Keil, B., Tisdall, M. D., Hoecht, P., Dietz, P., et al. (2013). Pushing the limits of in vivo diffusion MRI for the Human Connectome Project. *Neuroimage*, 80:220–233.

[40] Yang, J., Fan, J., Ai, D., Zhou, S., Tang, S., & Wang, Y. (2015). Brain MR image denoising for Rician noise using pre-smooth non-local means filter. *Biomedical engineering online*, 14(1), 1-20.

[41] Gudbjartsson, H., & Patz, S. (1995). The Rician distribution of noisy MRI data. *Magnetic resonance in medicine*, 34(6), 910-914.

[42] Aja-Fernández, S., & Tristán-Vega, A. (2012). Influence of noise correlation in multiple-coil statistical models with sum of squares reconstruction. *Magnetic Resonance in Medicine*, 67(2), 580-585.

[43] Scikit-learn: machine learning in Python — scikit-learn 0.24.2 documentation, `https://scikit-learn.org`, [accessed 2021-06-05].

[44] Bhandari, A. (2020). Feature Scaling for Machine Learning: Understanding the Difference Between Normalization vs. Standardization. *Analytics Vidhya*.

[45] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv*:1412.6980.

[46] Kotsiantis, S. B. (2013). Decision trees: a recent overview. *Artificial Intelligence Review*, 39(4), 261-283.

[47] Breiman, L. (2001). Random forests. *Machine learning*, 45(1), 5-32.

[48] Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4), 367-378.

[49] Zhang, D., Zhou, X., Leung, S. C., & Zheng, J. (2010). Vertical bagging decision trees model for credit scoring. *Expert Systems with Applications*, 37(12), 7838-7843.

[50] Glen, S. (2019), Decision Tree vs Random Forest vs Gradient Boosting Machines: Explained Simply, `https://www.datasciencecentral.com/profiles/blogs/decision-tree-vs-random-forest-vs-boosted-trees-explained`, [accessed 2021-06-05].

[51] Introduction to Boosted Trees - xgboost 1.5.0, `https://xgboost.readthedocs.io`, [accessed 2021-05-23].

[52] Brownlee, J. (2016), How to Configure the Gradient Boosting Algorithm, `https://machinelearningmastery.com`, [accessed 2021-05-23]

[53] Pytorch open source library, `https://pytorch.org`, [accessed 2020-10-10].

[54] Diligenti, M., Roychowdhury, S., Gori, M. (2017, December). Integrating prior knowledge into deep learning. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 920-923). IEEE.

[55] Daniele, A., Serafini, L. (2019, August). Knowledge enhanced neural networks. *In Pacific Rim International Conference on Artificial Intelligence* (pp. 542-554). Springer, Cham.

[56] Cohen, T. S., Geiger, M., Köhler, J., & Welling, M. (2018). Spherical cnns. *arXiv preprint arXiv:1801.10130*.

[57] Perraudin, N., Defferrard, M., Kacprzak, T., & Sgier, R. (2019). DeepSphere: Efficient spherical convolutional neural network with HEALPix sampling for cosmological applications. *Astronomy and Computing*, 27, 130-146.

[58] Taquet, M. (2013). *Multi-Fascicle Models of the Brain Microstructure for Population Studies: Acquisition, Estimation, Registration and Statistical Analysis* (Doctoral dissertation, EPFL, Lausanne, Switzerland).

[59] Hall, M. G., & Alexander, D. C. (2009). Convergence and parameter choice for Monte-Carlo simulations of diffusion MRI. *IEEE transactions on medical imaging*, 28(9), 1354-1364.

[60] Currie, S., Hoggard, N., Craven, I. J., Hadjivassiliou, M., & Wilkinson, I. D. (2013). Understanding MRI: basic MR physics for physicians. *Postgraduate medical journal*, 89(1050), 209-223.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**