July 2019

Master Thesis

Deep Learning in Medical Image Analysis:

A comparative analysis of multi-modal brain-MRI segmentation with 3D deep neural networks

MSc in Biomedical Engineering 2017-2019

Supervisor: Evangelos Dermatas | Student: Adaloglou M. Nikolaos | 1004130

Contents

Acknowledgments

*To my father, **Mihalis Adaloglou**, who was the greatest engineer I had the chance to meet, even for a short time.*

*To my grandparents, **Maria & Savvas Hantzidis**, who were the epitome of hard work, discipline and humbleness.*

To my four nephews, for sharing their wonderful thoughts with me, giving me again the rare ability to think like a child. Through their eyes, I found a bright new view of this world.

Nikolas M. Adaloglou

Thesis Abstract

Volumetric segmentation in magnetic resonance images is mandatory for the diagnosis, monitoring, and treatment planning. Manual practices require anatomical knowledge, are expensive, time consuming and can be inaccurate due to human factor. Automated segmentation can save physicians time and provide an accurate reproducible solution for further analysis. In this thesis, automated brain segmentation from multi-modal 3D magnetic resonance images (MRIs) is studied. An extensive comparative analysis of state-of-the-art 3D deep neural networks for brain sub-region segmentation is performed. We start by describing the fundamentals of MR Imaging because it is crucial to understand your input data to train a deep architecture. Then, we provide the reader with an overview of how deep learning works by extensively analyzing every component (layer) of a deep network. After we study the fields of magnetic resonance and deep learning separately, we attempt give a broader perspective of the intersection of this two fields with a different range of application of deep networks, from MR image reconstruction to medical image generation.

Our work is focused on multi-modal brain segmentation. For our experiments, we used two common benchmark datasets from medical image challenges. Brain MR segmentation challenges aim to evaluate state-of-the-art methods for the segmentation of brain by providing a 3D MRI dataset with ground truth tumor segmentation labels annotated by physicians. In order to evaluate state-of-the-art 3D architectures, we briefly analyze the author's approaches, as well as to provide the reader with an intuition behind the design choices. We perform a comparative analysis of the baseline architectures through extensive evaluations. The implemented networks were based on the specifications of the original papers. Finally, we discuss the reported results and provide future directions for implementing an open-source medical segmentation library in PyTorch along with data loaders of the most common medical MRI datasets. The goal is to produce a 3D deep learning library for medical imaging related tasks. We strongly believe in open and reproducible deep learning research. In order to reproduce our results, the code (alpha release) and materials of this thesis are available in https://github.com/black0017/MedicalZooPytorch

Introduction

The rise of deep networks in the field of computer vision, provided state-of-the-art solutions in problems that classical image processing techniques performed poorly. In the generalized task of image recognition, that includes problems such as object detection, image classification and segmentation, activity recognition, optical flow and pose estimation, we can easily claim that DNN (deep neural networks) have achieved superior performance. Along with this rise in computer vision, there has been a lot of interest in application in field of medical imaging. RBG monocular cameras contain a lot of information that can be capture in nearly no cost. As a consequence, there is abundance of RBG image data. Even though medical imaging data are not so easy to obtain, DNN's seem to be an ideal candidate to model such complex and high dimensional data. As it will be discussed later on, a medical image is often three or four dimensional, containing a lot more than three input channels. Another reason that this field attracts a lot of attention is its direct impact on human lives. Medical errors are the third-leading cause of death, after heart disease and cancer in the USA. Consequently, it is obvious that the first three causes of human deaths are related to medical imaging. That's why it is estimated that AI and deep learning in medical imaging will create a brand new market of more than a billion dollars by 2023.

This work serves as an intersection of these two worlds: Deep neural networks and medical imaging. Although we assume a small background on the general concept of machine learning, an attempt was made in order to assume as less known as possible. The purpose of this work is to provide an engineer with a solid background of the fundamentals of magnetic resonance tomography, as well as the current (2019) state of the art models in the fast-evolving field of deep learning. Even though we are aware that the field of deep learning is considered by most non-machine learning engineers a black box, we will extensively try to prove them wrong. Therefore, every operation performed by a DNN will be analyzed thoroughly and together with the explanation, there will be an intuition related to the underlying principle. Less literally, why we do what we do. Finally, even though the field of medical imaging is close to computer vision problems, solutions are not always directly extensible. From the lack of high-quality annotated data and pretrained feature extractors, to hardware limitations, the medical imaging field needs careful consideration that will be analyzed.

We will tackle the sub-problem of medical image segmentation, focused on MRI, which is one of the most popular and of interest. It is probably the task with the most well-structured datasets that someone can get access to. Since, online medical data collection is not as straightforward as it may sound; a collection of links to start your journey is provided in the corresponding section. Finally, yet importantly, in the context of this work no correlation between deep networks and biological neuron will be provided, because it is considered misleading. Everything regarding deep networks will be analyzed as layer and information processing and learning representations.

The brain is considered as the most well-organized system that processes information from different senses such as sight, hearing, touch, taste, and smell in an efficient and intelligent manner. One of the key mechanisms for information processing in a human brain is that the complicated high-level information is processed by means of the collaboration, i.e., connections of a large number of the structurally simple elements, called neurons. Powerful

data representations drive the performance of many machine learning algorithms. Designing features that generate such representations to effectively capture the information encoded in the given data is a particularly difficult task. In practice, this requires complex data preprocessing pipelines that do not generalize well between different image modalities or learning problems. The reason for that is that most of these systems are manually engineered for specific applications and rely exclusively on human ingenuity to disentangle and understand prior information hidden in the data in order to design the required features and discover intrinsic information about the given task.

In the context of volumetric image parsing, machine learning is used for anatomy detection and organ segmentation. The task of feature engineering becomes increasingly complex since the feature extraction is typically performed under challenging transformations such as arbitrary orientations or scales. Moreover, for robust parameter estimation, scanning the parameter space exhaustively is not feasible given the exponential increase in the size of the space with respect to the space dimensionality, i.e. the number of considered transformation parameters.

Medical image parsing subsumes the robust recognition, detection, and segmentation of objects, which proves to be a particularly difficult task for arbitrary 3D anatomical structures considering the variance in location, the non-rigid nature of the shape, as well as the differences in anatomy among different cases. Nonetheless, in order to achieve an accurate "automatic" segmentation of arbitrary anatomical structures, a robust and efficient solution is required for localizing the object of interest.

The rest of this work is organized as follows: Chapter 1 describes the basic principles of MR imaging. In Chapter 2, the fundamentals of machine and deep learning are briefly described, focused on the basic components of deep networks. In Chapter 3, the most commonly used deep learning architectures are analyzed, starting from conventional monocular images to 3D architectures, focused on dense pixel-wise predictions that will be referenced in the experiments. In Chapter 4, a broader overview of Deep Learning in the field in MRI are described, so as to provide the reader with a broader perspective of the advancements in the field of MR imaging due to the rise of deep learning. In Chapter 5, a detailed intuition of two normal brain MRI datasets are provided, that the experiments were conducted on. In Chapter 6, thorough quantitative and qualitative experimental results are reported and discussed. Finally, in Chapter 7, conclusions are drawn and future directions are provided.

CHAPTER 1 - Fundamentals of Magnetic resonance imaging

Introduction

In this chapter, a basic overview of the fundamentals of Magnetic resonance imaging is provided. An entry-level physics and engineering background is assumed. The aim is to focus on the principles of magnetic resonance imaging and keep things intuitive and simple as much as possible. In the context of this work, the direction of the magnetic field B_0 is defined as the z-axis.

Medical images

Medical imaging is the technique and process of creating visual representations of the interior of a body for clinical analysis and medical intervention, as well as visual representation of the function of some organs or tissues. **Medical imaging seeks to reveal internal structures hidden by the skin and bones, as well as to diagnose and treat diseases**. Medical imaging also establishes a database of normal anatomy and psychology to make it possible to identify abnormalities. Medical image main modalities include computed tomography (CT), magnetic resonance image (MRI) and positron emission tomography (PET).

1.1) The physics of MR imaging and excitation pulses

Medical magnetic resonance (MR) imaging **uses the signal from the nuclei of hydrogen atoms (1H) for image generation**. A hydrogen atom consists of a nucleus containing a single proton and of a single electron orbiting the nucleus. The proton has a positive charge and the electron a negative charge, resulting in zero charge. For the MR analysis proton is what we care about. Apart from its positive charge, the proton possesses spin, an intrinsic property of angular momentum of elementary particles. This means that the proton rotates about its axis like a spinning top. Spin quantum numbers may take only half- integer values. Although the direction of a particle's spin can be changed, it cannot be forced to spin faster or slower. A proton has two important properties: a) as a rotating mass (m), **the proton acts like a spinning top that strives to retain the spatial orientation of its rotation** axis, b) as a rotating mass with an electrical charge, the proton additionally has magnetic moment, denoted as B, and **behaves like a small magnet**. Therefore, it is affected by external magnetic fields and electromagnetic waves and, when it moves, induces a voltage in a receiver coil, as illustrated in Figure 1.

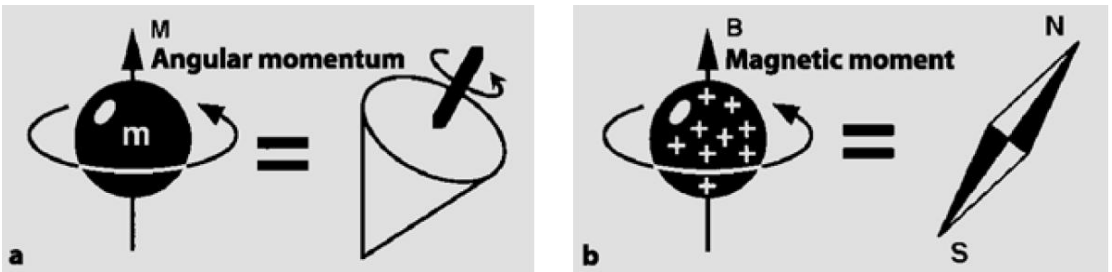


Figure 1, Proton properties

In this exact way, the orientation of its rotation axis from the magnetization vector B can be identified. Thus, when we describe the rotation of a proton, we are not referring to its (invisible) angular momentum, but to the "visible" motion of its magnetic axis, B. This motion can be quantitively measured in a receiver coil, because it generates a magnetic signal, similar

to the generated signal in an electrical generator. However, there is another important difference: a proton's spin always has the same magnitude and can neither be accelerated nor decelerated, precisely because it is a fundamental property of elementary particles. Spin is simply there all the time!

When an external force, typically the earth's gravitational field (G), acts on a spinning top and tries to alter the orientation of its axis, the top begins to wobble, a process called precession. At the same time, friction at the point of contact withdraws energy from the spinning top and slows down its rotation. As a result, its axis becomes more and more inclined, as shown in Figure 2(left).

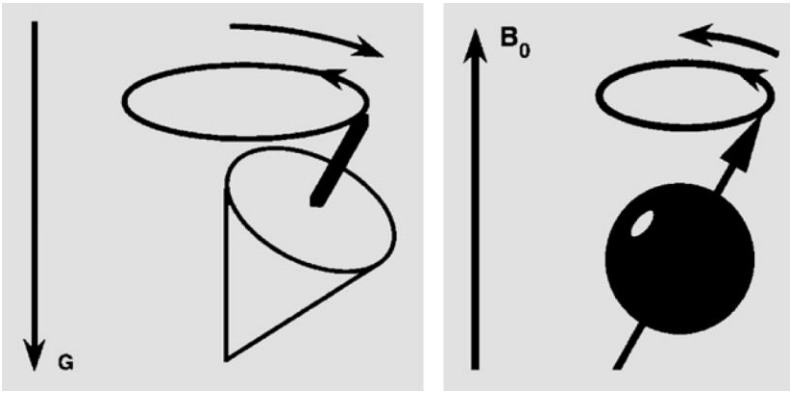


Figure 2, Magnetic field alters spin's orientation

In the case of a hydrogen nuclei: when it is **exposed to an external magnetic field**, denoted as B_0, the magnetic moments, or spins, **align with the direction of the field** like compass needles. The magnetic moments do not only align with the field but, like spinning tops, undergo **recession**, as shown in Figure 2 (right). The direction of orientation can be found using the right-hand rule.

Precession of the nuclei occurs at a characteristic speed, which is proportional to the strength of the applied magnetic field and is called Larmor frequency. Alignment of the spins parallel to the magnetic field is a gradual process and, as with spinning tops, is associated with the dissipation of energy. The Larmor or **precession frequency** is a very important concept that is at the core of MR imaging officially defined, **as is the rate at which spins wobble when placed in a magnetic field**. The Larmor frequency is directly proportional to the strength (B0) of the magnetic field and is given by the Larmor equation:

$$\boldsymbol{\omega_0 = \gamma_0 \cdot B_0 ,}$$

where ω_0 is the Larmor frequency in megahertz [MHz], γ_0 the gyromagnetic ratio, a constant specific to a particular nucleus and B_0 the strength of the magnetic field in tesla [T]. Protons have a gyromagnetic ratio of $\gamma = 42.58$ MHz/T, resulting in a Larmor frequency of 63.9 MHz at 1.5 T. Notation: **in the present work, the direction of the magnetic field B_0 is defined as the z-axis.**

While the spin system relaxes and settles into a stable state, **longitudinal magnetization, defined as M_z**, is building up in the z-direction, because the magnetic vectors representing the

individual magnetic moments add together. This also happens in the earth's magnetic field but the resulting longitudinal magnetization is very weak. The magnetic field B_0 of an MR imager is 60,000 times stronger than earth's, and the resulting longitudinal magnetization is correspondingly larger. Still, the MR signal is relatively weak, so magnetization must be large enough to obtain a signal at all. Actually, things are even a bit more complicated: **the spins tend to align parallel or anti-parallel to the magnetic field**, with parallel alignment being slightly preferred because it is equivalent to spins residing in a more favorable energy state. Hence, under steady-state conditions, a slightly larger fraction aligns parallel to the main magnetic field, as shown in Figure 3. This small difference actually produces the measurable net magnetization M_z and is represented by the net magnetization vector (NMV). Since the energy difference between the two orientations depends on the strength of the external magnetic field, M_z increases with the field strength.

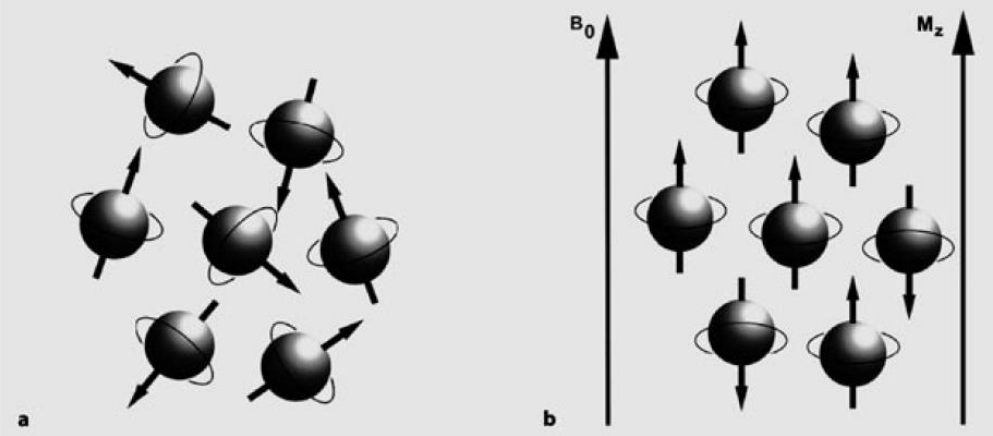


Figure 3, Longitudinal magnetization Mz

Energy can be introduced into such a stable spin system by applying an electromagnetic wave of the same frequency as the Larmor frequency. This is called the resonance condition. The required electromagnetic wave is generated in a powerful radio transmitter and applied to the object to be imaged by means of an antenna coil. The process of energy absorption is known as excitation of the spin system and results in the longitudinal magnetization being more and more tipped away from the z-axis toward the transverse (xy) plane perpendicular to the direction of the main magnetic field.

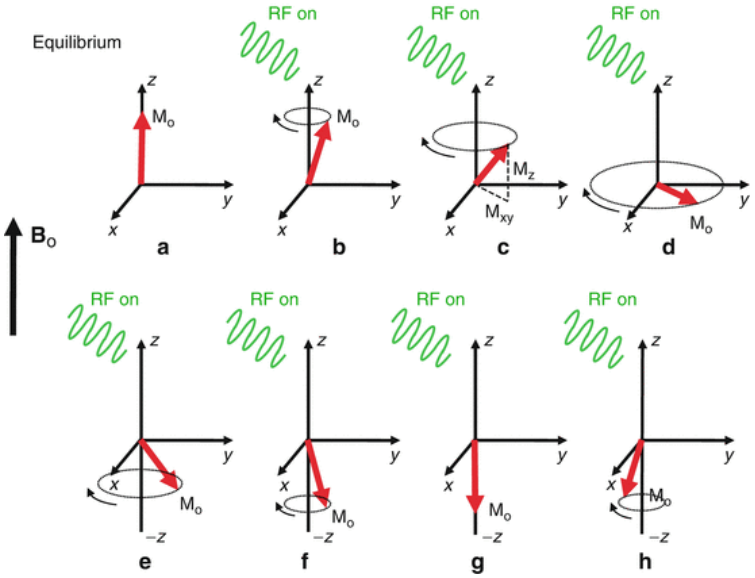


Figure 4, RF pulse applied resulting in transverse magnetization

All of the longitudinal magnetization is rotated into the transverse plane by a radiofrequency (RF) pulse that is strong enough and applied long enough to tip the magnetization by exactly 90° (90° RF pulse), as shown in Figure 4. The resulting magnetization is now denoted by M_{xy} rather than M_z, because it now lies in the xy-plane. Whenever transverse magnetization is present, it rotates or precesses about the z-axis, which has the effect of an electrical generator and induces an alternating voltage of the same frequency as the Larmor frequency in a receiver coil: the MR signal. This signal is collected and processed with sensitive receivers and computers to generate the MR image.

1.2) Relaxation

Immediately after excitation, the magnetization rotates in the xy plane and is now called transverse magnetization or M_{xy}. The rotating transverse magnetization gives rise to the MR signal in the receiver coil. However, the MR signal rapidly fades due to two independent processes that reduce transverse magnetization and thus cause a return to the stable state present before excitation: **spin-lattice interaction and spin-spin interaction**. These two processes cause **T1 relaxation and T2 relaxation**, respectively and will be further analyzed.

T1: Longitudinal Relaxation

Transverse magnetization decays and the magnetic moments gradually realign with the z-axis of the main magnetic field B_0, as discussed previously. The transverse magnetization remaining within the xy-plane – strictly speaking the projection of the magnetization vector onto the xy-plane, as shown in Figure 5, decreases slowly and the MR signal fades in proportion. As transverse magnetization decays, the longitudinal magnetization, Mz – the projection of the magnetization vector onto the z-axis – is slowly restored. This process is known as longitudinal relaxation or T1 recovery**. The nuclei can return to the ground state only by dissipating their excess energy to their surroundings** (the "lattice", which is why this kind of relaxation is also called spin-lattice relaxation). The time constant for this recovery is T1 and is dependent on the strength of the external magnetic field, B_0, and the internal

Brownian motion of the molecules. Biological tissues have T1 values of half a second to several seconds at 1.5 T.

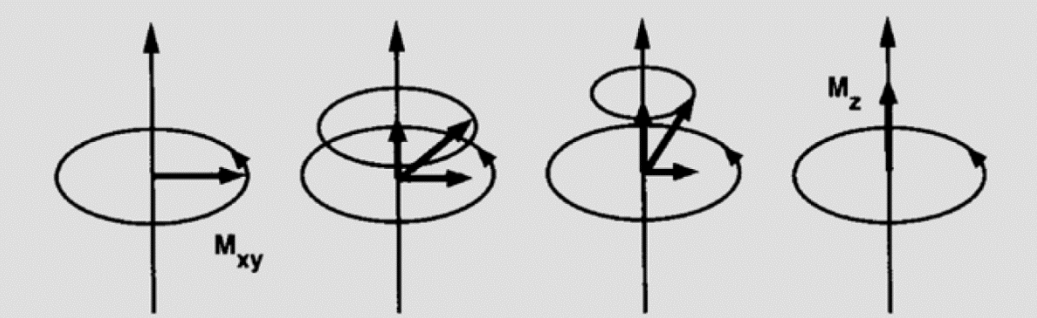


Figure 5, Magnetization relaxation (realignment)

T2/T2: Transverse Relaxation*

To understand transverse relaxation, it is first necessary to know what is meant by "phase". Phase refers to the position of a magnetic moment on its circular precessional path and is expressed as an angle. Consider two spins, A and B, precessing at the same speed in the xy-plane. If B is ahead of A in its angular motion by 10°, then we can say that B has a phase of +10 relative to A. Conversely, a spin C that is behind A by 30° has a phase of –30°, as shown in Figure 6.

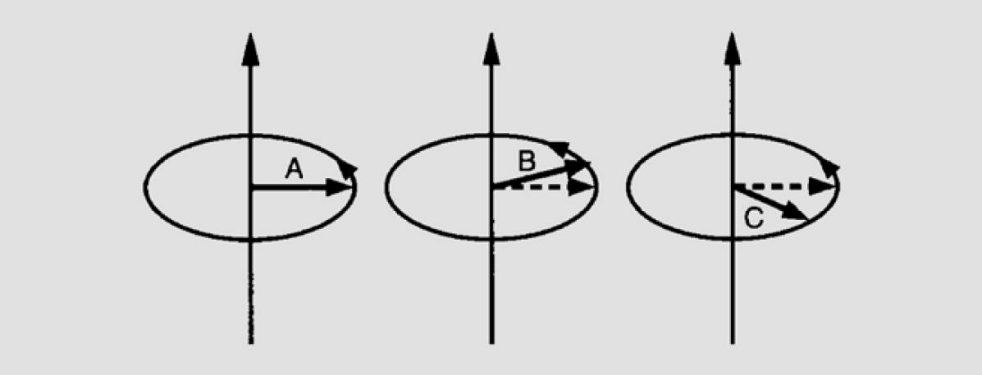


Figure 6, Phase difference

Immediately after excitation, part of the spins precess synchronously. These spins have a phase of 0° and are said to be in phase. This state is called phase coherence. Phase coherence is gradually lost as some spins advance while others fall behind on their precessional paths. **The individual magnetization vectors begin to cancel each other out instead of adding together.** The resulting vector sum, the transverse magnetization, becomes smaller and smaller and finally disappears, and with it the MR signal as shown in Figure 7.

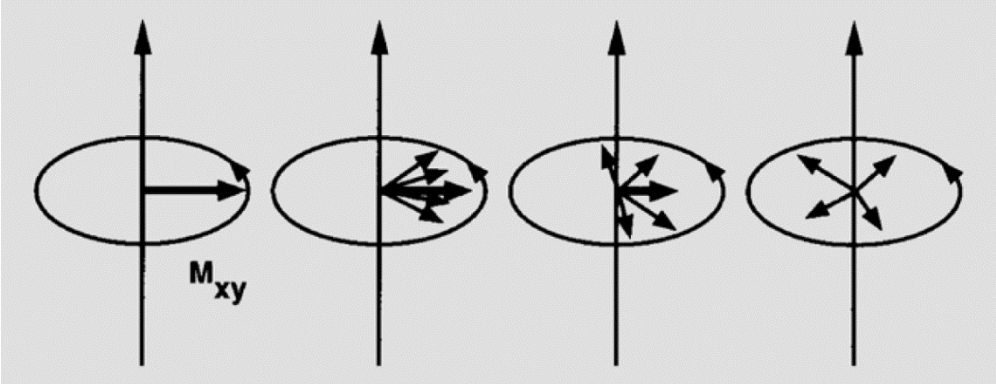


Figure 7, Magnetization cancelation

In other words, transverse relaxation is the decay of transverse magnetization because spins lose coherence (dephasing). Transverse relaxation differs from longitudinal relaxation in that the spins do not dissipate energy to their surroundings but instead exchange energy with each other. To summarize coherence is lost in two ways:

1. **Energy transfer between spins as a result of local changes in the magnetic field.** Such fluctuations are due to the fact that the spins are associated with small magnet fields that randomly interact with each other. Spins precess faster or slower according to the magnetic field variations they experience. **The overall result is a cumulative loss of phase.** It is a process due to pure spin-spin interaction and as such is unaffected by application of a 180° refocusing pulse. Dephasing occurs with the time constant T2 and is more or less independent of the strength of the external magnetic field, B0.
2. **Time-independent inhomogeneities of the external magnetic field B0.** These intrinsic inhomogeneities are caused by the magnetic field generator itself and by the very person being imaged. They contribute to dephasing, resulting in an overall signal decay that is even faster than described by T2. This second type of decay occurs with the time constant T2*, which is typically shorter than T2. **Most of the inhomogeneities that produce the T2* effect occur at tissue borders, particularly at air/tissue interfaces**, or are induced by local magnetic fields (e.g. iron particles). The loss of the MR signal due to T2* effects is called free induction decay (FID). T2* effects can be avoided by using spin echo sequences. **T2 denotes the process of energy transfer between spins, while T2* refers to the effects of additional field inhomogeneities contributing to dephasing.**

To summarize, T1 and T2 relaxation are **completely independent** of each other but occur more or less simultaneously! The decrease in the MR signal due to T2 relaxation occurs within the first 100–300 msec, which is long before there has been complete recovery of longitudinal magnetization Mz due to T1 relaxation (0.5–5 sec).

1.3) Image Contrast

Three intrinsic features of a biological tissue contribute to its signal intensity or brightness on an MR image and hence image contrast:

1. The proton density, i.e. the number of excitable spins per unit volume, determines the maximum signal that can be obtained from a given tissue. Proton density can be

emphasized by minimizing the other two parameters, T1 and T2. Such images are called proton density-weighted or simply proton density images.

2. The **T1** time of a tissue is **the time it takes the excited spins to recover** and be available for the next excitation. T1 affects signal intensity indirectly and can be varied at random. Images with contrast that is mainly determined by T1 are called T1-weighted images (T1w).
3. The **T2** time mostly determines **how quickly an MR signal fades** after excitation. The T2 contrast of an MR image can be controlled by the operator as well. Images with contrast that is mainly determined by T2 are called T2-weighted images (T2w).

Proton density and T1 and T2 times are intrinsic features of biological tissues and may vary widely from one tissue to the next. Depending on which of these parameters is emphasized in an MR sequence, the resulting images differ in their tissue-tissue contrast. This provides the basis for the exquisite soft-tissue discrimination and diagnostic potential of MR imaging: based on their specific differences in terms of these three parameters, tissues that are virtually indistinct on computed tomography (CT) scans can be differentiated by MRI without contrast medium administration.

1.4) Significant Parameters in MR image generation

In order to generate an MR image, a slice must be excited, and the resulting signal recorded many times. **Repetition time (TR) is the interval between two successive excitations of the same slice and is therefore crucial for T1 contrast**. When TR is long, more excited spins rotate back into the z-plane and contribute to the regrowth of longitudinal magnetization. The more longitudinal magnetization can be excited with the next RF pulse, the larger the MR signal that can be collected. If a short repetition time is selected, image contrast is strongly affected by T1.

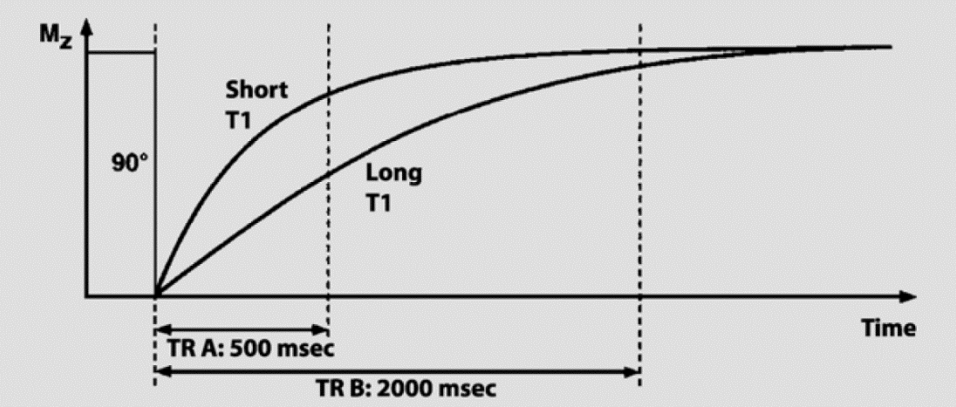


Figure 8, The effect of TR in T1 weighted images

Under this condition, tissues with a short T1 relax quickly and give a large signal after the next RF pulse (and hence appear bright on the image). Tissues with a long T1, on the other hand, undergo only little relaxation between two RF pulses and hence less longitudinal magnetization is available when the next excitation pulse is applied. These tissues therefore emit less signal than tissues with a short T1 and appear dark. **An image acquired with a short TR is T1-weighted because it contains mostly T1 information.** Therefore, by selecting the

repetition time, we can control the degree of T1 weighting of the resulting MR image. Tissues with a short T1 (fat, bone marrow) appear bright because they regain most of their longitudinal magnetization during the TR interval and thus produce a stronger MR signal. Tissues with a long T1 (muscle, cortex) appear dark because they do not regain much of their longitudinal magnetization during the TR interval and thus produce a weaker MR signal.

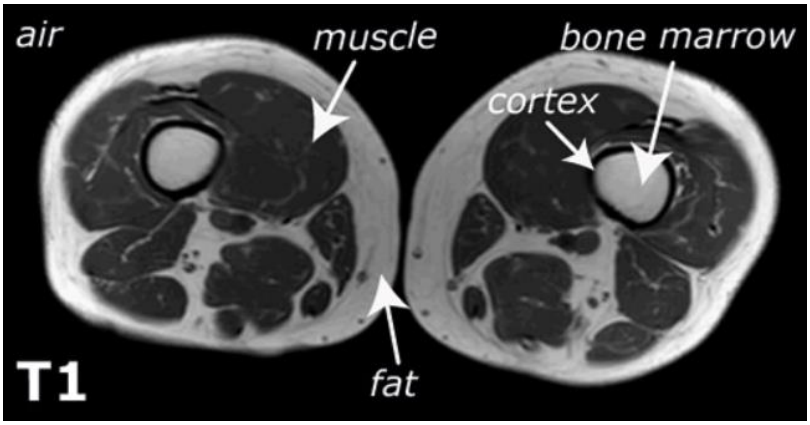


Figure 9, T1 weighted MRI image

Different gradients have to be applied to generate an MR image. Gradients are loops of wire or thin conductive sheets on a cylindrical shell lying just inside the bore of an MR scanner. When current is passed through these coils, a secondary magnetic field is created. This gradient field slightly distorts the main magnetic field in a predictable pattern, **causing the resonance frequency of protons to vary in as a function of position.** The primary function of gradients, therefore, is to allow spatial encoding of the MR signal, as it will be discussed in the next sections. In addition, gradients serve to induce controlled magnetic field inhomogeneities that are needed to encode the spatial origin of the MR signals. However, the gradients also contribute to spin dephasing. These effects must be reversed by applying a refocusing pulse before an adequate MR signal is obtained. The signal induced in the receiver coil after phase coherence has been restored is known as a spin echo and can be measured. **Echo time (TE) is the interval between application of the excitation pulse and collection of the MR signal**. The echo time **determines the influence of T2 on image contrast**. If a short echo time is used (less than about 30 msec), the signal differences between tissues are small, because T2 relaxation has only just started and there has only been little signal decay at the time of echo collection. The resulting image has low T2 weighting. Tissues with a short T2 having lost most of their signal appear dark on the image while tissues with a long T2 still produce a stronger signal and thus appear bright. This is why, for instance, cerebrospinal fluid (CSF) with its longer T2 (like water) is brighter on T2-weighted images compared with brain tissue. By selecting an echo time (TE), the operator can control the degree of T2 weighting of the resulting MR image.

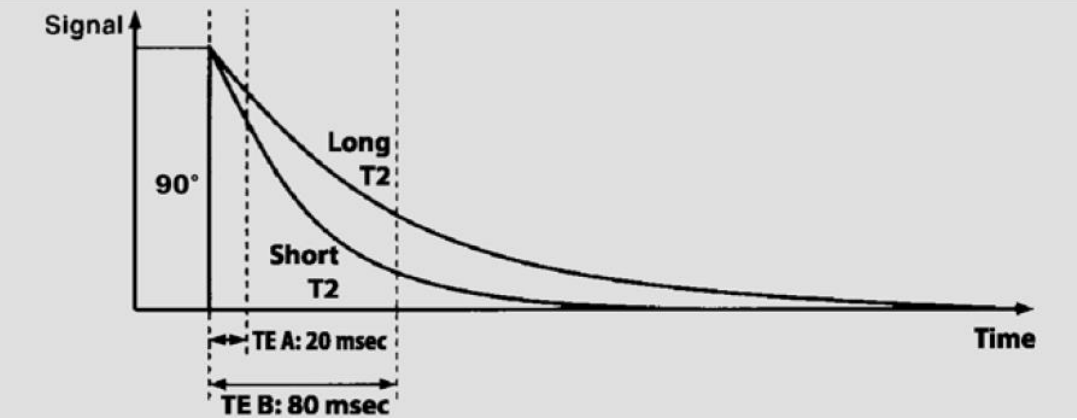


Figure 10, The effect of TE in T2 weighted images

A typical T1-weighted spin echo (SE) sequence is acquired with a TR/TE of 340/13 msec. A T2-weighted fast spin echo (FSE) MR image can be acquired with a TR/TE of 3500/120 msec. MR images that combine T1 and T2 effects are known as proton density-weighted images (PD images), which tend to have a higher signal to noise ratio. PD sequences are especially useful for evaluating structures with low signal intensities such as the bones or connective tissue structures such as ligaments and tendons. PD is often used in high-resolution imaging.

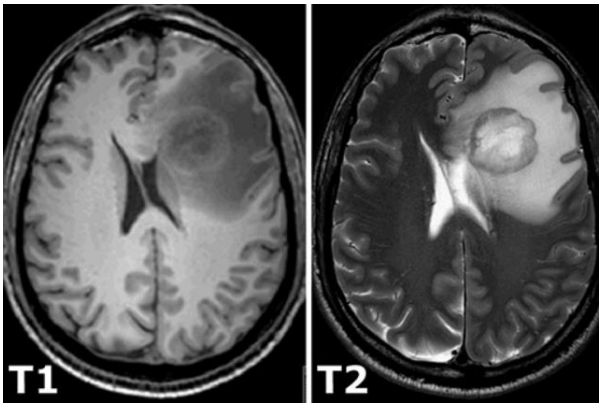


Figure 11, Brain Tumor in T1 and T2 weighted image

1.4) Slice Selection and Spatial Encoding

As a tomographic technique, MR imaging generates cross-sectional images of the human body. The excitation pulse is therefore delivered only to the slice we want to image and not to the whole body. Let us consider a transverse (axial) slice or cross-section through the body. The magnetic field generated by most MR scanners is not directed from top to bottom, but along the body axis of the person being imaged. This is the direction that will be designated by "z" since, as already said, **z stands for the direction of the main magnetic field.** The magnetic field gradients that now come into play are represented by wedges with the thick side indicating the higher field strength and the tip the lower field strength.

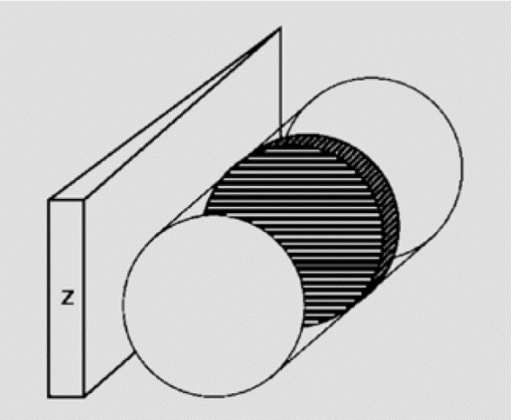


Figure 12, Slice selection in z-axis

Both the excitation of a specific slice and the identification of the site of origin of a signal within the slice rely on the fact that the precessional or **Larmor frequency is proportional to the magnetic field strength**. In addition, recall that protons are excited only by an RF pulse with a frequency roughly equal to their Larmor frequency (resonance condition). If a uniform field of identical strength were generated throughout the body, all protons would have the same Larmor frequency and would be excited simultaneously by a single RF pulse.

To enable selective excitation of a desired slice, **the magnetic field is therefore made inhomogeneous in a linear fashion along the z-direction** by means of a gradient coil. As a result, the magnetic field strength has a smooth gradient so that, for example, it is weakest at the patient's head and strongest at the feet. The Larmor frequencies thus change gradually along the z-axis and each slice now has its unique frequency. Hence, **application of an RF pulse that matches the Larmor frequency of the desired slice will excite only protons within the chosen slice while the rest of the body remains unaffected**, as shown in Figure 13.

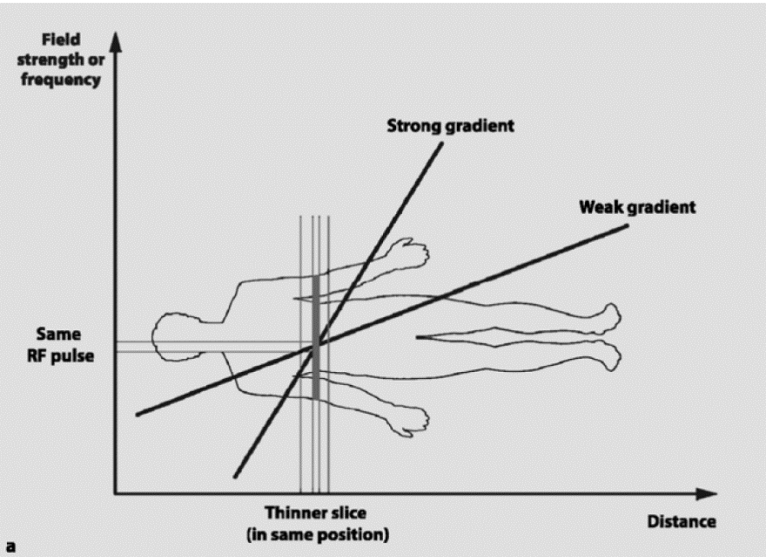


Figure 13, Slice selection with gradient coils

Depending on their position along the gradient, protons are temporarily exposed to magnetic fields of different strength and hence differ in their precessional frequencies. A shallow gradient generates a thicker slice while a steep gradient generates a thinner slice. Slice position is defined by changing the center frequency of the RF pulse applied.

Having selected slice position and thickness by application of an appropriate slice-select gradient, we can now proceed to explain how the spatial position of an MR signal is identified. This is accomplished by spatial encoding, which is the most difficult task in generating an MR image and requires the application of additional gradients that alter the magnetic field strength along the y- and x-axes. **Spatial encoding comprises two steps, phase encoding and frequency encoding.** These two steps are discussed below. Both frequency-encoding gradients and phase-encoding gradients do work in exactly the same way but are used for different purposes. All imaging gradients temporarily change the resonant frequencies of protons while the gradient is being applied.

For **phase encoding, a gradient in the y-direction** (from top to bottom) is switched on after the spins have been excited and precess in the xy-plane. Such a phase-encoding gradient **alters the Larmor frequencies of the spins according to their location along the gradient.** As a result, the excited spins higher up in the scanner experience a stronger magnetic field and thus gain phase relative to the somewhat slower spins further down. The result is a phase shift of the spins relative to each other, as shown in Figure 14. The degree of phase shift is determined by the duration and amplitude of the phase-encoding gradient and by the physical location of the precessing nuclei along its length. **The phase gain is higher for nuclei closer to the top of the scanner**. When the gradient is switched off after some time, all spins return to their initial rate of precession yet are now ahead or behind in phase relative to their previous state. Phase now **varies along the y-axis in a linear fashion** and each line within the slice can thus be identified by its unique phase.

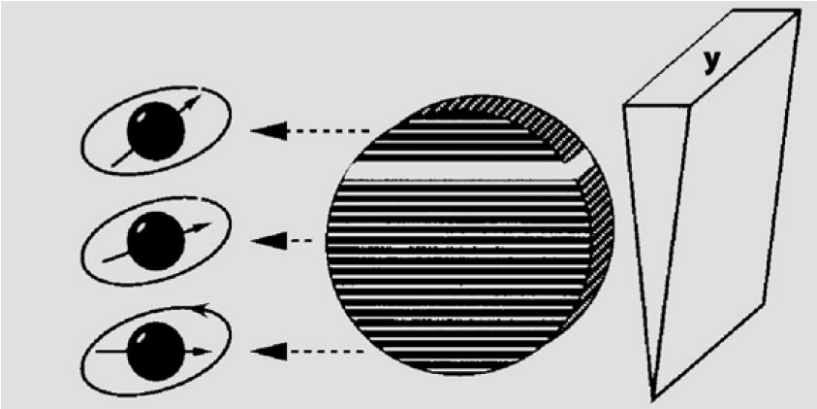


Figure 14, Phase encoding

The second spatial dimension of the MR signal that needs to be identified is encoded by changes in frequency along the x-direction. To this end, a frequency-encoding gradient is applied – in our example along the x-axis. This gradient generates a magnetic field that increases in strength from right to left. The corresponding changes in Larmor frequencies make spins on the left side precess slower than the ones on the right side. When we collect the MR signal while the frequency-encoding gradient is switched on, we do not obtain a single

frequency but a whole frequency spectrum, as shown in Figure 15, comprising high frequencies from the right edge of the slice and low frequencies from the left edge. Each column of the slice is thus characterized by a specific frequency. Frequency and phase together enable unique spatial identification of each volume element (voxel).

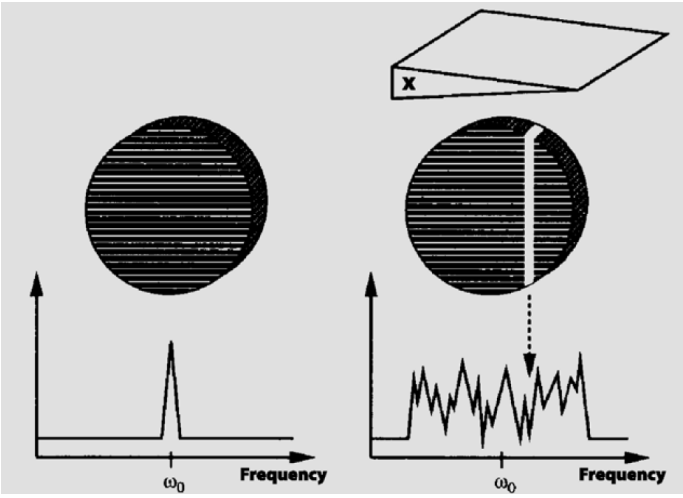


Figure 15, Frequency encoding via spectrum

The MR signal measured in this way contains two pieces of information. **The frequency locates the signal along the x-axis**. This information can be extracted directly by applying a Fourier transform to decompose the signal into its component frequencies along the frequency encoding direction. This mathematical operation serves to identify the individual frequencies that make up a signal. The phase distribution within each frequency provides information on the place of origin of the corresponding signal component along the y-axis. How do we get the second piece of information when we merely have the sum of all spins with the same frequency but different phases is another thing to consider.

The phases of the individual spins cannot be derived from a single signal but only from a set of signals. In this respect, the MR signal is comparable to a mathematical equation with many unknowns. To deal with this issue of multiple unknowns, **we must repeat the sequence many times with increasing or decreasing gradient strengths.** The set of echoes acquired with different phase encodings allows us to derive the required phase-encoded spatial information by applying a second Fourier transform, this time along the y-axis. Hence, **for spatial encoding in two dimensions, the Fourier transform has to be applied twice**, which is why this technique is called two-dimensional Fourier transform (2D-FT). To perform such complex calculations, an MR scanner is equipped with a dedicated array processor. **Repeated measurements are performed with a specific temporal delay, the repetition time previously mentioned (TR)**. The number of phase-encoding steps performed depends on the desired image quality. More phase-encoding steps improve resolution and image quality but also prolong scan time.

1.5) K-Space

The k-space formalism is a technique that proved invaluable in unifying different MR imaging techniques. They showed that **the demodulated MR signal generated by freely precessing nuclear spins in the presence of a linear magnetic field gradient equals the Fourier transform**

of the effective spin density. Raw data in MRI are the transversal components of the magnetization in the imaging object after excitation, sampled from the receiver coil signal and stored as a function of time during the data acquisition of an MR imaging sequence. In a transverse slice, the horizontal axis is usually set as the frequency encoding direction, shown as k_x while the vertical axis is the phase encoding direction of excited protons k_y (Figure 17). This is also known as **k-space data**. Thus, **k-space** is an abstract concept and refers to a **data matrix containing the raw MRI data before reconstruction.** From the basic k-space formula, it follows immediately that we can reconstruct an image simply by taking the inverse Fourier transform of the sampled data. **A single line of k-space is scanned per RF excitation.**

The k-space is an extension of the concept of Fourier space well known in MR imaging. **It represents the spatial frequency information in two or three dimensions of an object.** The k-space is defined by the space covered by the phase and frequency encoding data. The relationship between k-space data and image data is the Fourier transformation. The data acquisition matrix contains raw data before image processing. The position in k-space is directly related to the gradient across the object being imaged. By changing the gradient over time, the k-space data are sampled in a trajectory through Fourier space. Every point in the raw data matrix contains part of the information for the complete image. A point in the raw data matrix does not correspond to a point in the image matrix. **The outer rows of the raw data matrix, the high spatial frequencies, provide information regarding the borders and contours of the image, the detail of the structures. The inner rows of the matrix, the low spatial frequencies, provide information on the general contrast of the image.** The brightness of each point represents the relative contribution of that point's unique spatial frequency to the final image, as illustrated in Figure 16.

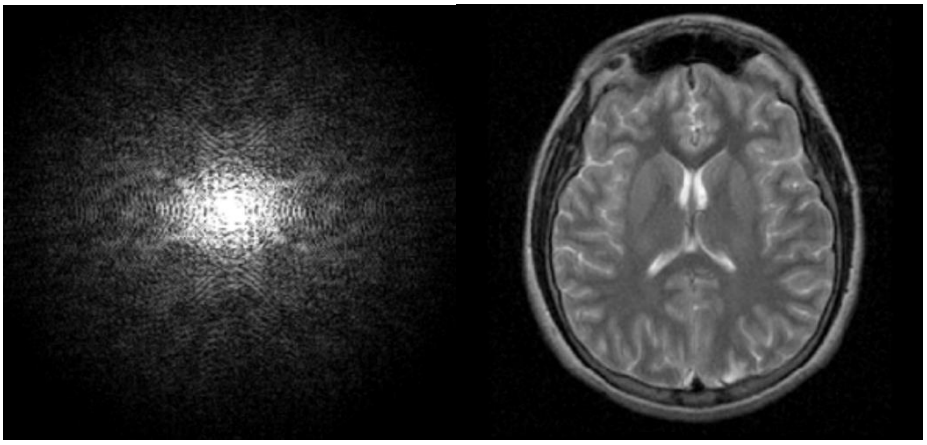


Figure 16, K-space(left) & Fourier Transformed (right)

Each data point in k-space is derived directly from the MR signal. A single slice corresponds to a k space plane acquired in real-time. Each point on the k-space contains specific frequency, phase (x, y, coordinates) and signal intensity information (brightness). Inverse FT is applied after k-space acquisition to derive the final image. **Each pixel in the resultant image is the weighted sum of all the individual points in the k-space**.

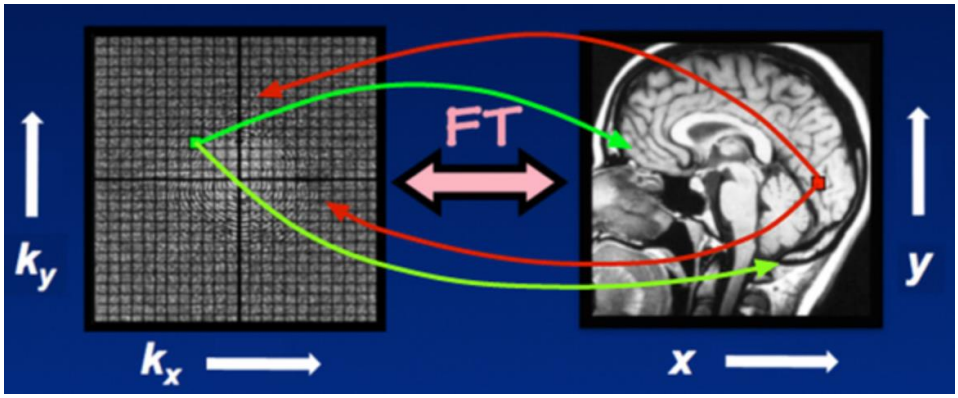


Figure 17, Each point in k-space maps to every point in the image, and vice-versa.

Hence, disruption of any point in the k-space translates into some form of final image distortion, determined by the frequency- and phase-related data stored in that particular point. **In general, central regions of the k space encode contrast information and peripheral regions of the k space encode spatial resolution.** Although the k-space and the MR image appear quite different, they contain identical information about the scanned object. The two representations may be converted to one another using an integral Fourier transformation. A discrete Fourier or fast Fourier transform is generally used.

1.6) Additional Factors affecting Image Quality

The relationship between the MR signal and the amount of image noise present is expressed as the signal-to-noise ratio (SNR). Mathematically, the SNR is the quotient of the signal intensity measured in a region of interest (ROI) and the standard deviation of the signal intensity in a region outside the anatomy or object being imaged (i.e. a region from which no tissue signal is obtained). **A high SNR is desirable in MRI**. The SNR is dependent on the following parameters:

- **Slice thickness and receiver bandwidth**. Thinner slices are associated with more noise, and so the SNR decreases with the slice thickness. Conversely, thicker slices are associated with other problems such as an increase in partial volume effects. The receiver bandwidth is the range of frequencies collected by an MR system during frequency encoding. A wide receiver bandwidth enables faster data acquisition but also reduces SNR as more noise is included. With a narrow bandwidth, on the other hand, there will be more motion artifacts and the number of slices that can be acquired for a given TR is limited.
- **Field of view (FOV)**. In simple terms, the FOV determines the size of the pixels. A smaller FOV results in a smaller pixel size as long as the matrix is unchanged. Pixel size is crucial for the spatial resolution of the MR image. With the same FOV, a finer matrix results in an improved spatial resolution. Conversely, a coarser matrix results in a poorer spatial resolution when the FOV is held constant.
- **Size of the (image) matrix**. A matrix is a two-dimensional grid of rows and columns. Each square of the grid is a pixel, which is assigned a value that corresponds to a signal intensity. Each pixel of an MR image provides information on a corresponding three-dimensional volume element, termed a voxel. The voxel size determines the spatial resolution of an MR image.

- **Number of excitations**. The number of excitations (NEX) denotes how many times a signal from a given slice is measured. The SNR, which is proportional to the square root of the NEX, improves as the NEX increases, but scan time also increases linearly with the NEX.
- Imaging parameters (**TR, TE, flip angle**). Other parameters affecting the SNR are the sequence used, echo time (TE), repetition time (TR), and the flip angle. The SNR increases with the TR but the T1 effect is also lost at longer TRs. Conversely, the SNR decreases as the TE increases. With a short TE, the T2 contrast is lost. For this reason, the option of shortening TE to improve SNR is available only for T1-weighted sequences.
- **Magnetic field strength**. Applying a higher magnetic field strength increases longitudinal magnetization because more protons align along the main axis of the magnetic field, resulting in an increase in SNR. The improved SNR achieved with high-field systems can be utilized to generate images with an improved spatial resolution or to perform fast imaging.
- Selection of the **transmit and receive coil** (RF coil). An effective means to improve SNR, without increasing voxel size or lengthening scan time, is selecting an appropriate radiofrequency (RF) coil. In general, an RF coil should be as close as possible to the anatomy being imaged and surround the target organ. The nearer the coil can be placed to the organ under examination, the better the resulting signal. RF coils can be used either to transmit RF and receive the MR signal or to act as receiver coils only. In the latter case, the body coil delivers excitation pulses.

1.7) The Magnet, the gradient system and radiofrequency system

The main magnetic field generated by the magnet must have the following features:

1. An **adequate strength**, which typically ranges from 0.1 to 3.0 T in medical MR imaging.
2. A high **stability** without fluctuations in field strength.
3. The best **homogeneity** possible with a **uniform strength** throughout the entire field and without “holes”. Field homogeneity is usually expressed in ppm relative to the main field over a certain distance.

Furthermore, three common types of magnets are distinguished:

- **Resistive magnets** are conventional electromagnets that depend on a high and constant power supply to create a magnetic field. The maximum field strength generated by resistive magnets is about 0.3 T. Their major disadvantages are the high operating costs due to the large amounts of power required and a field homogeneity that is often poor. An advantage is the safety of the system as the field can be turned off instantly in an emergency.
- **Permanent magnets** consist of ferromagnetic substances and create a magnetic field that is maintained without an external power supply. However, permanent magnets are very heavy, can generate a field with a maximum strength of only 0.5 T, and rely on a constant external temperature.
- **Superconducting magnets** consist of a coil made of a niobium-titanium (Nb-Ti) alloy whose resistance to current flow is virtually eliminated when cooled to near absolute

zero (about 4°Kelvin or –269°C). In this superconducting state, which is achieved using coolants known as cryogens (usually liquid helium), a current once induced flows practically forever. Once the magnetic field has been established, it is maintained without additional power input. Very strong and highly homogeneous magnetic fields of up to 18T can be generated using superconducting magnets. However, liquid helium evaporates and must be resupplied regularly. In an emergency, it is not possible to simply switch off the magnet. About 95% of all MR systems used today have superconducting magnets. A quench refers to a magnet's sudden loss of superconductivity with subsequent breakdown of the magnetic field and may be induced by very minute movements of the coil. Due to the frictional energy released by this process, the coil temperature rises above the superconductivity threshold and the coils suddenly develop resistance.

Magnetic field gradients are applied for slice selection and spatial encoding. **A set of three separate gradient coils, each with its own amplifier, is needed to alter the magnetic field strength along the x-, y-, and z-axes.** These are switched on separately or in combination, e.g. to define an oblique slice. The isocenter is the geometric center of the main magnetic field, where the field strength is not affected by any of the three gradients. The gradient coils generate magnetic fields that are small compared with the main field but need a current of several hundred amperes. The changing magnetic fields generated when the gradients are switched lead to the typical banging sound heard during an MR scan. Similar to a loudspeaker, which is nothing but a coil inside a magnetic field, the gradient coils "try to move" when the current is switched on and off, which causes a noisy clanging. Despite the high currents, the gradient fields must be extremely stable in order to prevent image distortions.

The radiofrequency (RF) system comprises a powerful RF generator (the Larmor frequency at 1.5 T is 63.8 MHz, which is in the range of FM transmitters) and a highly sensitive receiver**. The stability of these two components is crucial**: as both the frequency and the phase of the signal are needed for spatial encoding, any distortions, e.g. by phase rotation introduced by the receiver, would result in a blurred image. Moreover, to adequately detect the weak MR signal, effective RF shielding of the scanner room is necessary to prevent interference from external sources. This can be achieved by housing the magnet in a closed conductive structure known as a **Faraday cage**. The RF subsystem also includes the transmit and receive coils. These may be combined coils acting as both transmitters and receivers such as the body coil, which is integrated into the scanner. It is not visible from the outside and consists of a "cage" of copper windings encircling the patient. The RF transmitter serves to deliver pulses that correspond to the resonant frequency of hydrogen atoms. Careful coil selection according to the anatomy being imaged is important for optimizing image quality.

1.8) Inversion recovery T1 and T2-FLAIR MR imaging

Inversion recovery (IR) is a conventional spin echo sequence preceded by a 180° inverting pulse. The time between the 180° inverting pulse and the 90° pulse is called the inversion time (TI). The function of the inverting pulse is to flip the initial longitudinal magnetization (M_o) of all tissues in the imaged slice or volume to point opposite to the direction of the main magnetic field ($\mathbf{B_o}$). During the TI interval, these inverted tissues undergo T1 relaxation as they variably seek to re-establish magnetization along the +*z*-direction. When spin echo signal generation begins (at the 90°-pulse), the initial longitudinal magnetizations of different tissues are now separated based on their different intrinsic T1 relaxation times. The degree of separation (and hence image contrast) is controlled by varying the *TI* parameter in the pulse sequence. Additional contrast effects are also obtained by manipulation of *TR* and *TE*.

In addition to the timing parameters (*TR*/*TI*/*TE*), how the MR signal is reconstructed has a significant effect on image contrast and appearance. Historically, IR techniques were widely used in the "early days" of MRI (1980-1985). They produced excellent image contrast, especially for T1-weighted imaging, but were time-consuming (15-20minute sequences typical). Today, IR techniques are widely used in all branches of MRI, but especially for neuroradiology and cardiac imaging applications.

T2-FLAIR stands for T2-weighted-Fluid-Attenuated Inversion Recovery. Originally just called "FLAIR", this technique was developed in the early 1990's by the Hammersmith research team led by Graeme Bydder, Joseph Hajnal, and Ian Young. Their original sequences used TI values of 2000-2500 to null signal from CSF, coupled with very long TRs (8000) and TEs (140) to create strong T2-weighting. Very long imaging times (15-20 min typical), the T2-FLAIR technique repeatedly proved itself by revealing a wide range of lesions, including cortical, periventricular, and meningeal diseases that were difficult to see on conventional images. By the late 1990s, the use of fast spin echo signal generation significantly reduced imaging times and T2-FLAIR became a standard protocol for routine imaging. Today, the venerable old spin-density sequences (long TR/Short TE spin echo) have been nearly completely replaced by T2-FLAIR imaging, at least in neuroimaging.

Chapter Summary

In this chapter, we introduced the concepts of MR imaging, in order to proceed with our analysis. We briefly described how magnetic fields change the direction of water molecules in the human body and different tissues respond differently to RF pulses. We understood the difference between the two basic phenomena that produce T1 and T2 weighted MR images and described the spatial encoding that produces the k-space data (Fourier space).

CHAPTER 2 - Machine learning & Deep learning principles

Introduction

Machine learning is a form of AI that enables a system to learn from data rather than through explicit programming. Machine learning uses a variety of algorithms that iteratively learn from data to improve, describe data, and predict outcomes. As the algorithms ingest training data, it is then possible to produce more precise models based on that data. The model is the output generated when you train your machine learning algorithm with data. After training, when you provide a model with an input, you will be given an output. Then, when you provide the model with data, you will receive a prediction based on the data that trained the model.

In supervised learning, the idea is to find a model that can predict the answers when we don't know them (future or incomplete data). We give the machine learning algorithm the labeled data, with inputs and their corresponding outputs, and let it learn from the relationships between them. Machine learning models are provided experiences in the form of training data but are tuned to produce accurate predictions for the training data by an optimization algorithm. The main goal of the models is to be able to generalize their learned expertise, and deliver correct predictions for new, unseen data. A model's generalization ability is typically estimated during training using a separate data set, the validation set, and used as feedback for further tuning of the model. After several iterations of training and tuning, the final model is evaluated on a test set, used to simulate how the model will perform when faced with new, unseen data.

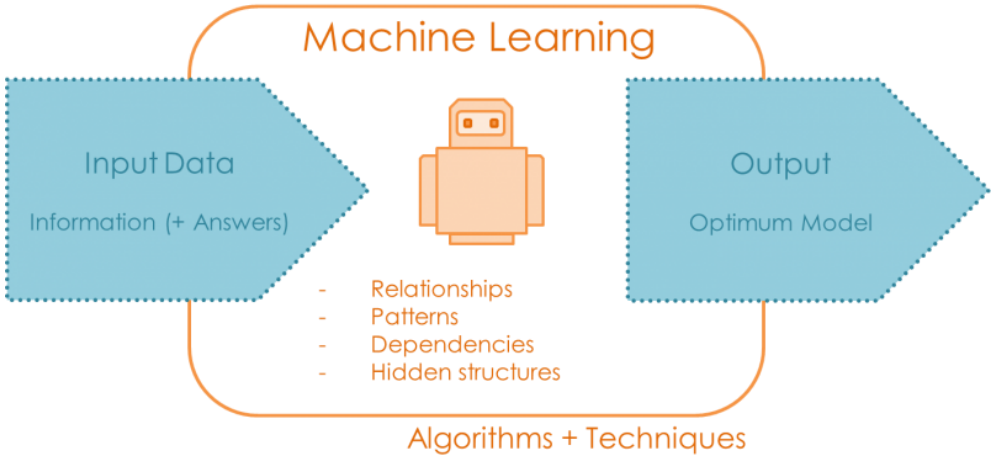


Figure 18, Machine Learning as a black box

2.1) Generalization of Machine Learning Models

In machine learning, generalization refers to how well the concepts learned by a machine learning model apply to specific examples not seen by the model during the learning process. **The goal of a well-trained machine learning model is to generalize well from the training data to any data from the problem domain**. This allows us to make predictions in the future on data the model has never seen. There is a terminology used in machine learning when we talk about how well a machine learning model learns and generalizes to new data, namely **overfitting** and **underfitting**. Overfitting and underfitting are the two biggest causes for poor performance of machine learning algorithms.

Overfitting refers to a model that 'learns' the training data too well. Overfitting happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data. This means that the noise or random fluctuations in the training data is picked up and learned as concepts by the model. The problem is that these concepts do not apply to new data and negatively impact the model's ability to generalize. Overfitting is more likely with nonparametric and nonlinear models that have more flexibility when learning a target function. As such, many nonparametric machine learning algorithms also include parameters or techniques to limit and constrain how much detail the model learns.

Underfitting refers to a model that can neither model the training data nor generalize to new data. An underfit machine learning model is not a suitable model and will be obvious, as it will have poor performance on the training data. Underfitting is often not discussed, as it is easy to detect given a good performance metric. The remedy is to move on and try alternate machine learning algorithms. Nevertheless, it does provide a good contrast to the problem of overfitting.

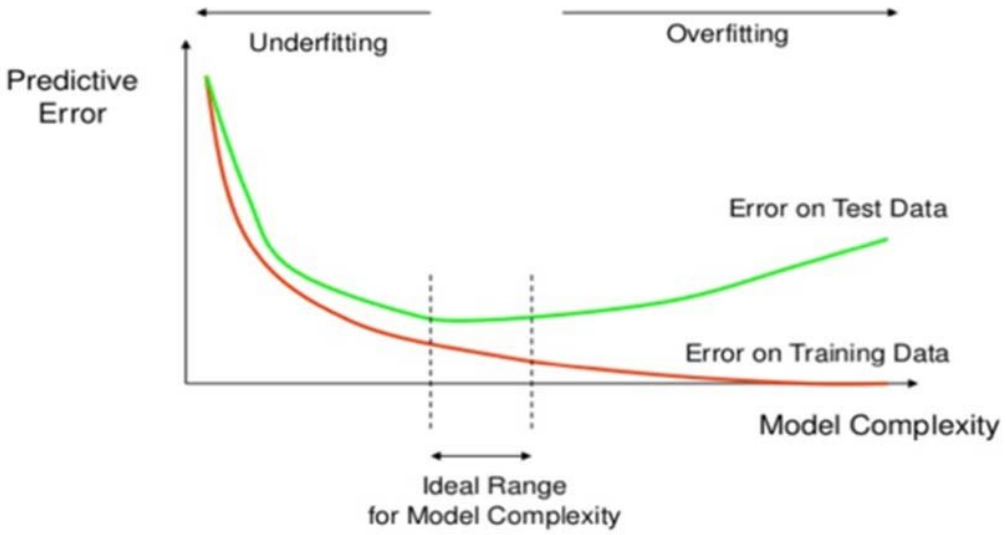


Figure 19 Underfitting and overfitting

Ideally, you want to select a model at the sweet spot between underfitting and overfitting, as depicted in Figure 19. **The aforementioned goal is very difficult to achieve in practice.** To understand this goal, we can look at the performance of a machine learning algorithm over time as it is learning a training data. We can plot, as in Figure 19, both the performance on the

training data and on a test set we have held back from the training process. Over time, as the algorithm learns, the error for the model on the training data goes down and so does the error on the test set. If we train for too long, the performance on the training dataset may continue to decrease because the model is overfitting and learning the irrelevant detail and noise in the training dataset. At the same time, the error for the test set starts to rise again as the model's ability to generalize decreases. The desired spot is the point just before the error on the test dataset starts to increase where the model has good performance on both the training dataset and the unseen test set. This is often not useful technique in practice, because by choosing the stopping point for training using the performance on the test dataset it means that the test set is no longer "unseen" or a standalone objective measure. Some knowledge about that data has leaked into the training procedure. There are two additional techniques you can use to help find the sweet spot in practice: **resampling methods and a validation dataset.**

The most popular resampling technique is **k-fold cross validation**. It allows you to train and test your model k-times on different subsets of training data and build up an estimate of the performance of a machine learning model on unseen data. A validation dataset is simply a subset of your training data that you hold back from your machine learning algorithms until the very end of your project. After you have selected and tuned your machine learning algorithms on your training dataset you can evaluate the learned models on the validation dataset to get a final objective idea of how the models might perform on unseen data.

2.2) A brief overview of gradient descent & backpropagation

The provided overview is by no means a complete description of gradient descent & backpropagation. For an in-depth explanation, the reader is encouraged to visit [25]-[28]. Machine learning models typically have parameters and a loss function to evaluate how good a particular set of parameters are. Many machine learning problems reduce to finding a set of weights for the model which minimizes the cost function. Consider the simple following example: if the prediction is **p**, the target is **t**, and error metric is mean squared error, then the loss function:

$$J(\Theta) = \frac{1}{N}\sum_{1}^{N}(\boldsymbol{p} - \boldsymbol{t})^2, \quad where\ \boldsymbol{p} = f(input)$$

Note that the prediction **p** depends on the input, as well as the model the parameters **Θ**. During training, our aim is to find a set of values for **Θ** that minimize the loss function J. This means our prediction **p** will be close to the target **t** in the end of the training process. Gradient descent is an iterative method that performs such optimization. We start with some set of values for our model parameters - here **Θ** - and improve them slowly.

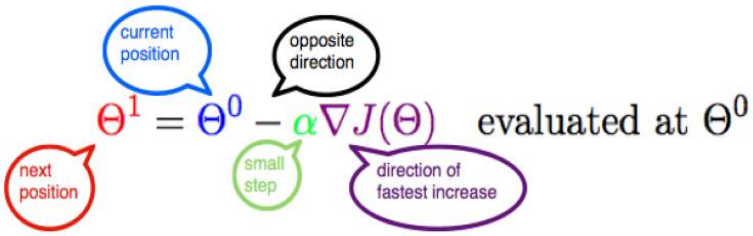


Figure 20, The update rule

To improve a given set of weights, we try to get a sense of the value of the cost function for weights similar to the current weights (by calculating the gradient). Then we move in the direction, which reduces the cost function, which is called the **update rule, as shown**. By repeating this step many times, we will continually minimize the cost function J until it stops reducing, or some other predefined termination criteria is met.

Backpropagation is about understanding how changing the weights (parameters) in a network changes the loss function by computing the partial derivatives, using the chain rule. So, it is an algorithm that uses gradient descent to minimize error in the predictions. The backpropagation is all about calculating the gradient of the error function of any given loss function and a neural network, while considering the different weights within that neural network, **which is nothing more than calculating the partial derivatives of the loss function with respect to model parameters.** As the field of deep learning gained popularity, there has been a huge demand for different alterations in deep networks. In order to deal with this demand synchronous deep learning frameworks such as TensorFlow and Pytorch created an environment where you define the forward pass of your deep architecture and given the scalar loss function(s), gradients of the network's parameters are calculated automatically. This is usually referred in bibliography as automatic differentiation. The most significant advantage of backpropagation is it's scaling capability from small to large models.

2.3) Expressive Power of Deep models

Deep neural networks have an extremely large number of parameters compared to the traditional statistical models. The model description can easily grow out of control. However, having numerous parameters is necessary for a neural network to obtain high expressivity power. Because of its great capability to capture any flexible data representation, deep neural networks have achieved great success in many applications. The **Universal Approximation Theorem [28] states that a feedforward network with 1) a linear output layer, 2) at least one hidden layer containing a finite number of neurons and 3) some activation function can approximate any continuous function on a compact subset of R^n to arbitrary accuracy**. The theorem was first proved for sigmoid activation function. Later it was shown that the universal approximation property is not specific to the choice of activation, but the multilayer feedforward architecture. Although a feedforward network with a single layer is sufficient to represent any function, the width has to be exponentially large. The universal approximation theorem does not guarantee whether the model can be learned or generalized properly. Often, adding more layers helps to reduce the number of hidden neurons needed in a shallow (small) network. To take advantage of the universal approximation theorem, we can always find a neural network to represent the target function with error under any desired threshold, but we need to pay the price — the network might grow super large.

Having discussed that, it is less surprising to see that **DNNs are able to learn unstructured random noise perfectly**. If labels of image classification dataset are randomly shuffled, the high expressivity power of deep neural networks can still empower them to achieve near-zero training loss. These results do not change with regularization terms added. In addition, deep learning models are **heavily over-parameterized** and can often get to perfect results on training data. In the traditional view, like bias-variance trade-offs, this could be a disaster that nothing may generalize to the unseen test data. However, as is often the case, such "overfitted" (training error = 0) deep learning models still present a decent performance on out-of-sample test data.

Considering a neural network with a great number of parameters, forming a high-dimensional parameter space, **the learning happens on this high-dimensional objective landscape.** The shape of the parameter space manifold is critical. For example, a smoother manifold is beneficial for optimization by providing more predictive gradients and allowing for larger learning rates, this was claimed to be the reason why batch normalization has succeeded in stabilizing training. Even though the parameter space is huge, fortunately we do not have to worry too much about the optimization process getting stuck in local optima, as it has been shown that local optimal points in the objective landscape almost always lay in saddle-points rather than valleys, as depicted in Figure 21. In other words, there is always a subset of dimensions containing paths to leave local optima and keep on exploring.

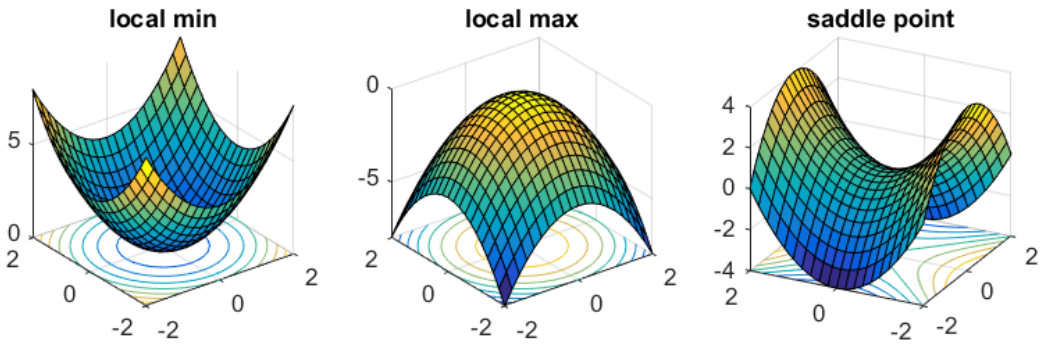


Figure 21, parameter space

Since the parameter space has such high dimensionality, it is probably not necessary to exploit all the dimensions to learn efficiently. If we only travel through a slice of objective landscape and still can learn a good solution, the complexity of the resulting model is likely lower than what it appears to be by parameter counting.

2.4) The building blocks of Deep Neural Networks (DNNs)

Deep learning techniques have become the de facto standard for a wide variety of computer vision problems. They are, however, not limited to image processing and analysis, but are outperforming other approaches in areas like natural language processing, speech recognition and synthesis, and in the analysis of unstructured, tabular-type data using entity embeddings. The sudden progress and wide scope of deep learning, and the resulting surge of attention and multi-billion-dollar investment, has led to a virtuous cycle of improvements and investments in the entire field of machine learning. Healthcare providers generate and capture enormous amounts of data containing extremely valuable signals and information, at a pace far surpassing what traditional methods of analysis can process.

Linear Layers (Fully connected layers or Dense layers)

In order to understand the so-called Linear layers let's consider its simplest form of a single unit, usually called as perceptron. It is the simplest learnable artificial neural model and it is represented with an input vector, denoted as **x**, a vector of trainable connection weights **w**, a bias scalar term b and an output unit **y** as shown in the equation below. Since the perceptron model has a single layer of an output unit, it is also called a single-layer neural network. Given an observation $\mathbf{x} \in R^D$, the scalar value of the output unit y is obtained from a nonlinear activation function **f** (·) by taking the weighted sum of the inputs, which is the inner product, as follows:

$$y(\boldsymbol{x}\,; \boldsymbol{w}, b) = f\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} + b\right) = \boldsymbol{f}\left(\sum_{i=1}^{\boldsymbol{D}} \boldsymbol{x_i}\,\boldsymbol{w_i} + b\right)$$

If we take multiple linear combinations of vector x with different coefficients this would result in the generalized linear layer, commonly referred as dense or fully connected layer, as shown.

$$\boldsymbol{Y} = f(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$$

Convolutional Layers

Convolution is a widely used technique in signal processing, image processing, and other engineering fields. In Deep Learning, a kind of layer architecture, Convolutional Neural Network (CNN), is named after this technique. However, convolution in deep learning is essentially the cross-correlation in signal processing. In signal processing, convolution is defined as:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau$$

We essentially perform element-wise multiplication and addition. If **f** and **g** are two functions defined in the time domain (1-dimensional), convolution is defined as the integral of the product of the two functions after one is reversed and shifted. Function **g** is usually the filter. It is reversed, and then it is slided along the axis. For every position, we calculate the area of the intersection between *f* and reversed *g*. That intersection area is the convolution value at that specific position. On the other hand, cross-correlation is known as sliding dot product or sliding inner product of two functions. **The filter in cross-correlation is not reversed**. It directly slides through the function **f**. **The intersection area between *f* and *g* is the cross-correlation.** There is a subtle difference between these two operations. Although, everyone refers to this operation in the neural network context as convolution, because the only change is that we do not flip the one of the two signals, as it would result in the addition of unnecessary complexity. The weights of filters are learned during training. If the reversed function **g** in the example above is the right function, then after training the learned filter would look like the reversed function **g**. So, since flipping does not make a difference, as it will be learnt through training, **cross-correlation is equivalent to convolution in this context.** In Deep Learning, the filters in convolution are not reversed.

One may argue in why use the convolution operation in the first place. First, convolutional layers are a great way to deal with raw pixel inputs into a neural network. Each convolutional layer has a set of filters, which work by performing convolution. In simple terms, it is just taking the sum of the element-wise product between a portion of the image, called patch, and the filter (also called a kernel). It focuses on a patch of the image because we need the two

matrices to be the same size. The size of the filter is critical to the task and the input, because it defines the receptive field in the input grid. **We do not preset the filter weights to perform a specific task - instead the neural network will learn the weights that it deems the best with backpropagation**, resulting in each filter to learn different useful information called **features**.

Therefore, the purpose of doing convolution is to extract useful features from the input. In image processing, there is a wide range of different filters one could choose for convolution. Each type of filters helps to extract different aspects or features from the input image, such as edges. Similarly, in Convolutional Neural Networks (CNNs), different features are extracted through convolution using filters whose weights are automatically learned during training. All these extracted features then are 'combined' to make decisions.

There are a few advantages of doing convolution, such as **weights sharing and translation invariance**. Convolution also takes into account the spatial relationship of pixels, because the lie on a grid. These could be very helpful especially in many computer vision tasks, since those tasks often involve identifying objects where certain components have certain spatially relationship with other components. This also applies and proves to be helpful in medical image analysis, because data lie in a 3D grid. Less literally, **the voxel space lies on a 3D grid**.

When dealing with high-dimensional inputs such as images, it is impractical to connect all neurons (weights) with the input. Instead, we will connect each neuron to only a local region of the input volume. The spatial extent of this connectivity is a hyperparameter called the receptive field of the neuron (equivalently this is the filter size). The extent of the connectivity along the depth axis is always equal to the depth of the input volume.

Before describing the process, it is important to clarify a few terminologies: layers, channels, feature maps, filters, and kernels. From a hierarchical point of view, the concepts of layers and filters are at the same level, while channels and kernels are at one level below. Channels and feature maps are the same thing. A layer could have multiple channels (or feature maps): an input layer has three channels if the inputs are RGB images. "channel" is usually used to describe the structure of a "layer". Similarly, "kernel" is used to describe the structure of a "filter". However, the difference between filter and kernel is a bit tricky. Sometimes, they are used interchangeably, which could create confusions. Essentially, these two terms have subtle difference. A "Kernel" refers to a 2D array of weights. The term "filter" is for 3D structures of multiple kernels stacked together. For a 2D filter, filter is same as kernel. Nevertheless, for a 3D filter and most convolutions in deep learning, **a filter is a collection of kernels**. Each kernel is unique, emphasizing different aspects of the input channel.

Convolutional over 2D grids (single channels convolutions)

Having said all the above, let us see convolution for an image with one channel, such as a specific slice of an MRI image, as illustrated in the figure below. Let's suppose that the red rectangle to be the only part we care about, with the patch being a 5x5 matrix with pixel intensities and the kernel to be a 3x3, as shown. The kernel is slided across the 2D grid and every time we perform cross correlation. The result of a 5x5 patch is stored in a 3x3 matrix and is propagated in the next layer of the network.

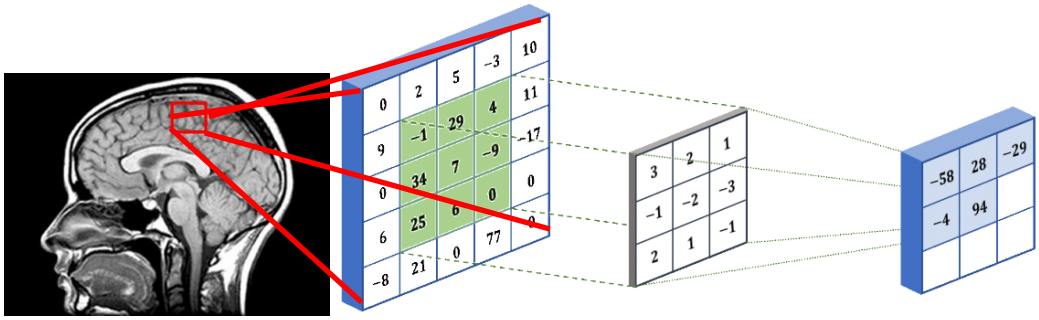


Figure 22, 2D Convolution over a 2D grid

Convolution over 3D grids (multi-channel 2D convolutions)

In many applications, we are dealing with images with multiple channels. A typical example is the RGB image. Each RGB channel emphasizes different aspects of the original image. Another example of multi-channel data is medical images, as illustrated in the Figure.

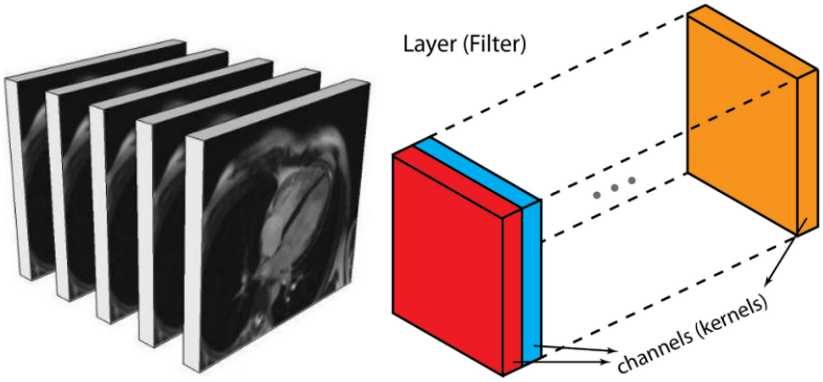


Figure 23, MRI slices is a multi-channel 3D grid input in a CNN

In addition, a convolutional-net layer usually consists of multiple channels (typically hundreds of channels). Each channel describes different aspects of the previous layer. Therefore, we make transitions between multi channels layers with different depths. However, how do we transform a layer with depth N to the following layer with depth M? The answer is simple: by using M convolutional filters in an N channel input. We can think of multi-channel convolution as sliding a 3D filter matrix through the input layer. Notice that the input layer and the filter have the same depth (channel number = kernel number). **The 3D filter moves only in 2-direction, height & width of the image. That is why such operation is called as 2D convolution although a 3D filter is used to process 3D volumetric data.** At each sliding position, we perform element-wise multiplication and addition, which results in a single number. In the example shown below, the sliding is performed at 5 positions horizontally and 5 positions vertically. Overall, we get a single output channel, as illustrated below. Now we can see how one can make transitions between layers with different depths. Let us say the input layer has N channels, and we want the output layer has *M* channels. What we need to do is to just apply *M* filters to the input layer. Each filter has *N* kernels.

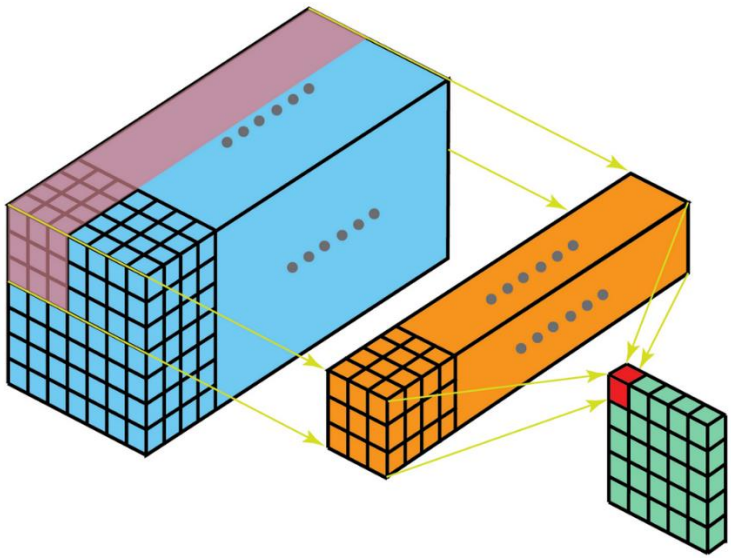


Figure 24, 2D Convolution over 3D a grid

For an input image with size of i, kernel size of k, padding of p, and stride of s, the output image from convolution has size o:

$$o = lower\_bound\left(\frac{i + 2p - k}{s}\right) + 1$$

3D Convolution over 3D grids (multi-channel 3D convolutions)

In the last illustration of the previous section, we see that we were actually perform convolution to a 3D volume. However, typically, we still call that operation as 2D convolution in Deep Learning. It is a 2D convolution on a 3D volumetric data. The filter depth is same as the input layer depth. The 3D filter moves only in 2-direction (height & width of the image). The output of such operation is a 2D image (with one channel only). Naturally, there are 3D convolutions. They are the generalization of the 2D convolution. Here in 3D convolution, the filter depth is smaller than the input layer depth (kernel size < channel size). As a result, the 3D filter can move in all 3-direction (height, width, channel of the image). At each position, the element-wise multiplication and addition provide one number. Since the filter slides through a 3D space, the output numbers are arranged in a 3D space as well. The output is a 3D volume.

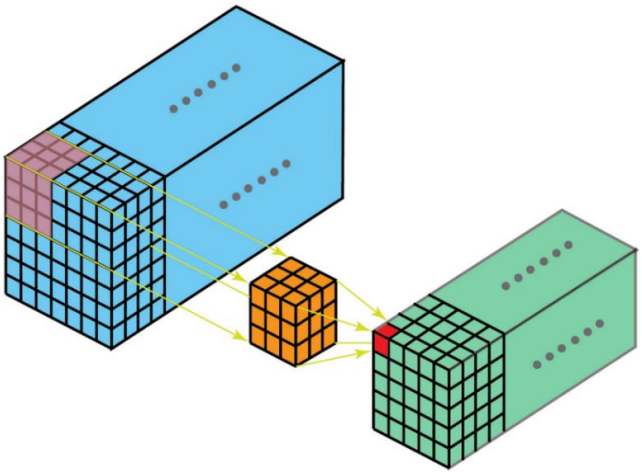


Figure 25, 3D Convolution over a volume

Similar to the 2D convolutions, which encode spatial relationships of objects in a 2D domain, **3D convolutions can describe the spatial relationships of objects in the 3D space**. Such 3D relationship is important for some applications, such as in 3D segmentations or reconstructions of biomedical imagining. Nevertheless, the additional dimension is not restricted to 3D spatial space. There are many computer vision applications where the extra dimension is the time domain or depth sensor data, for tasks as activity recognition. In this way, we can process monocular RGB videos or RGB+D images, which renders the 3D convolution a far more abstract operation.

Finally, let us look at another interesting operation, which is used in state-of-the-art deep networks. That is **1 x 1 kernel convolution**. You may wonder why this is helpful. The operation is trivial for layers with only one channel. There, we multiply every element by a number. Although, things become interesting if the input layer has multiple channels. The following picture illustrates how **1 x 1 convolution** works for an input layer with dimension H x W x D. After 1 x 1 convolution with filter size 1 x 1 x D, the output channel is with dimension H x W x 1. If we apply N such 1 x 1 convolutions and then concatenate results together, we could have a output layer with dimension H x W x N.

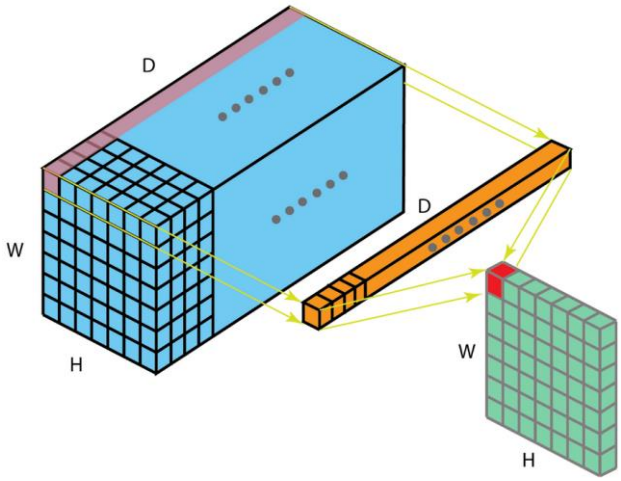


Figure 26, 1x1 kernel convolution

A few advantages of 1 x 1 convolutions are the dimensionality reduction for efficient computations, the efficient low dimensional embeddings they produce, or feature pooling and the fact that they apply nonlinearity again after convolution.

Transpose Convolutional Layers (deconvolutional layers)

For many applications and in many network architectures, we often want to do transformations going in the opposite direction of a normal convolution, i.e. we would like to perform up-sampling. A few examples include generating high-resolution images and mapping low dimensional feature map to high dimensional space such as in semantic segmentation. (In the later example, semantic segmentation first extracts feature maps in the encoder and then restores the original image size in the decoder so that it can classify every pixel in the original image.) Traditionally, one could achieve up-sampling by applying interpolation schemes or manually creating rules. Although Modern CNNs, on the other hand, tend to let the network itself learn the proper transformation automatically, without human intervention. To achieve that, we can use the transposed convolution. The transposed convolution is also known as deconvolution, or fractionally strided convolution in the literature. However, it is worth noting that the name “deconvolution” is less appropriate, since transposed convolution is not the real deconvolution as defined in signal processing domain. Technically speaking, deconvolution in signal processing reverses the convolution operation. That is not the case here. Because of that, some authors are strongly against calling transposed convolution as deconvolution.

It is always possible to implement a transposed convolution with a direct convolution. For an example in the image below, we apply transposed convolution with a 3 x 3 kernel, shown as a light shadow, over a 2 x 2 input padded with a 2 x 2 border of zeros using unit strides. The up-sampled output has 4 x 4 size. It is worth noticing that one can map the same input to a different output size, by applying fancy padding & stride.

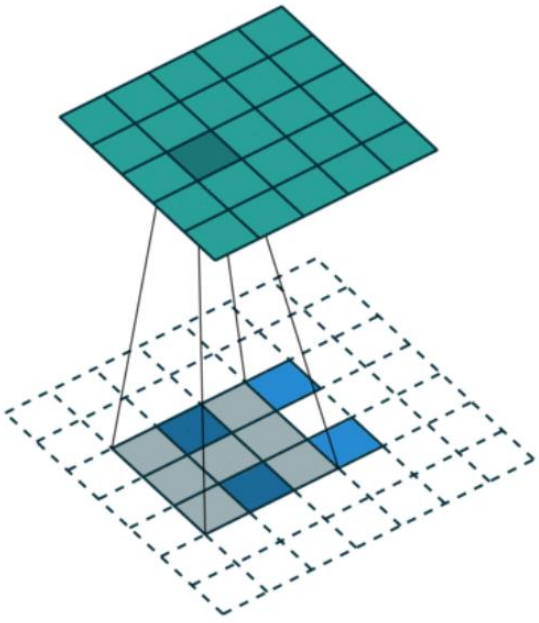


Figure 27, A transpose convolution example of a 2x2 input to a 5x5 output

Viewing transposed convolution in the examples above could help us build up some intuitions. If we look at how it is implemented through matrix multiplication for optimization purposes, the name of transpose convolution is clearly demonstrated. The transposed convolution has uneven overlap when the filter size is not divisible by the stride. This "uneven overlap" puts more of the paint in some places than others, thus creates the checkerboard effects. In fact, the unevenly overlapped region tends to be extreme in two dimensions. There, two patterns are multiplied together, the unevenness is squared. Two things one could do to reduce such artifacts, while applying transposed convolution. First, **make sure you use a filer size that is divided by your stride**, avoiding the overlap issue. Secondly, one can **use transposed convolution with stride = 1**, which helps to reduce the checkerboard effects. However, artifacts can still leak through, as seen in many recent models.

Loss Functions

Loss function is an important part in neural networks, which is used to measure the inconsistency between predicted value and actual label. Loss is a non-negative scalar value, where the robustness of model increases along with the decrease of the value of the loss function. Loss function is the hard core of empirical risk function as well as a significant component of structural risk function. Many functions could be used to estimate the error of a set of weights in a neural network. We prefer a function where the space of candidate solutions maps onto a smooth (but high-dimensional) landscape that the optimization algorithm can reasonably navigate via iterative updates to the model weights.

When modeling a classification problem where we are interested in mapping input variables to a class label, we can model the problem as predicting the probability of an example belonging to each class. In a binary classification problem, there would be two classes, so we may predict the probability of the example belonging to the first class. In the case of multiple-class classification, we can predict a probability for the example belonging to each of the classes.

In the context of this work, two loss functions will be used and will be presented: a) cross-entropy loss and b) the dice loss. The most commonly used loss function for the task of image segmentation is a **pixel-wise cross entropy loss**. This loss examines *each pixel individually*, comparing the class predictions (depth-wise pixel vector) to our one-hot encoded target vector. Because the cross-entropy loss evaluates the class predictions for each pixel vector individually and then averages over all pixels, we are essentially asserting equal learning to each pixel in the image. This can be a problem if your various classes have unbalanced representation in the image, as the most prevalent class can dominate training. Some recent works, FCN included, discuss weighting this loss for each **output channel** in order to counteract a class imbalance present in the dataset. As described in Unet a loss weighting scheme for each **pixel** such that there is a higher weight at the border of segmented objects is proposed. This loss-weighting scheme helped their model segment cells in biomedical images in a *discontinuous* fashion such that individual cells may be easily identified within the binary segmentation map.

Another popular loss function for image segmentation tasks is based on the Dice coefficient, which is essentially a measure of overlap between two samples. This measure ranges from 0

to 1 where a Dice coefficient of 1 denotes perfect and complete overlap. The Dice coefficient was originally developed for binary data, and can be calculated as:

$$Dice = \frac{2\,|A \cap B|}{|A| + |B|}$$

As illustrated in Figure 28 and 29, cross entropy evaluates the loss along the pixel channels, while the dice loss between output channels.

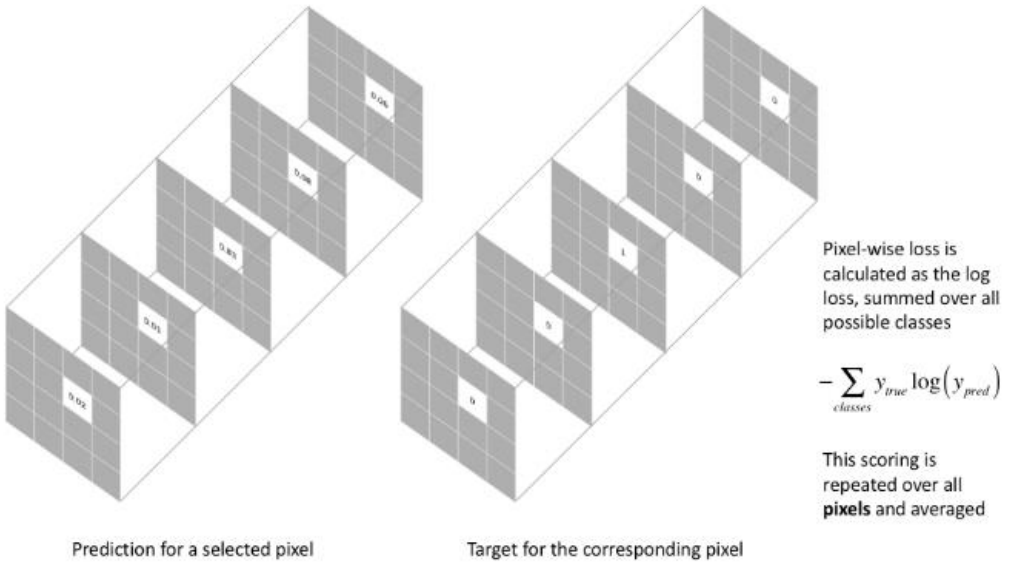


Figure 28, Cross-entropy loss

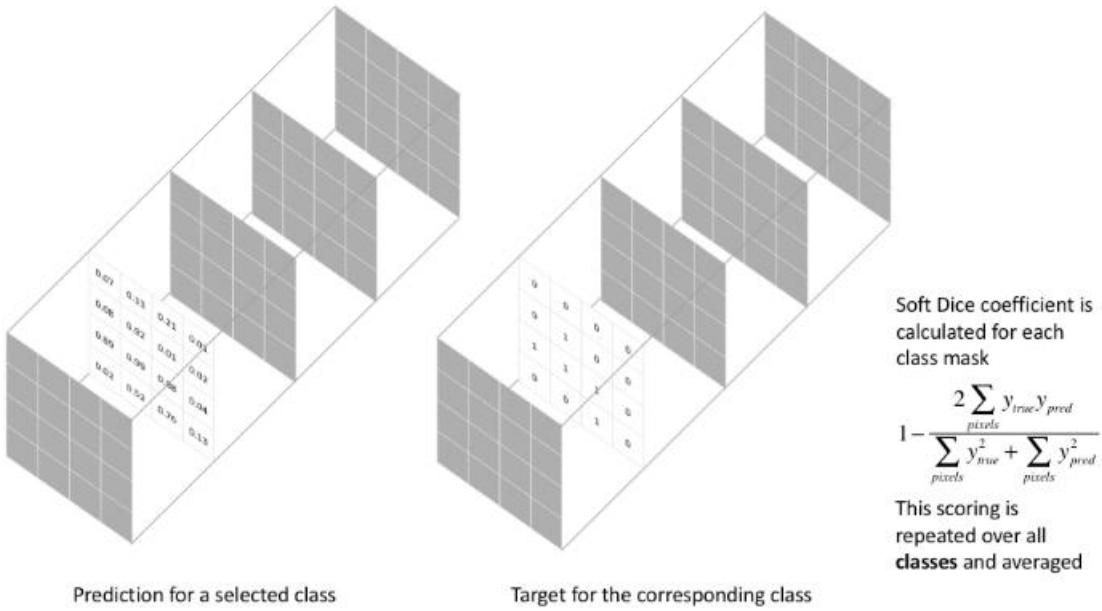


Figure 29, Dice loss

Batch normalization

Batch Normalization [10] was first introduced by two researchers at Google, Sergey Ioffe and Christian Szegedy in their paper [10]: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift in 2015. The authors showed that batch normalization improved the top result of ImageNet (2014) by a significant margin using only 7% of the training steps. Now, Batch Normalization is used in almost all CNN architectures.

In order to intuitively understand the reasoning behind batch normalization, it's necessary to grasp the concept of covariance shift. Covariance is a measure of the joint variability of two random variables. If the greater values of one variable mainly correspond with the greater values of the other variable, and the same holds for the lesser values, (i.e., the variables tend to show similar behavior), the covariance is positive. The sign of the covariance therefore shows the tendency in the linear relationship between the variables. The magnitude of the covariance is not easy to interpret because it is not normalized and hence depends on the magnitudes of the variables. In our minds, it makes sense that every mini-batch used in the training process should have the same distribution. However, most of the times this is not the case. In other words, a mini-batch should not have only images from one of the two subsets above. Ideally, it should have images randomly selected from both subsets in each mini-batch. Remember that each min-batch is used in the forward information in order to calculate the accumulated gradients of the training step.

The same intuition is graphically depicted in Figure 30. The last column of Figure 30 shows the two classes (roses and non-roses) in the feature space (shown in two dimensions for ease of visualization). The blue curve shows the decision boundary. We can see the two subsets lie in different regions of the feature space. This difference in distribution is called the covariate shift**. When the mini-batches have images uniformly sampled from the entire distribution, there is negligible covariate shift.** However, when the mini-batches are sampled from only one of the two subsets shown in Figure 30 (right), there is a significant covariate shift. This makes the training procedure significantly slow.

Figure 30, The two subsets have very different distributions. The last column shows the distribution of the two classes in the feature space using red and green dots. The blue line show the decision boundary between the two classes.

Just as it made intuitive sense to have a uniform distribution for the input layer, it is advantageous to have the same input distribution for each hidden unit over time while training. But in a neural network, each hidden unit's input distribution changes every time there is a parameter update in the previous layer. This is called **internal covariate shift**. This makes training slow and requires a very small learning rate and a good parameter initialization. This problem is solved by normalizing the layer's inputs over a mini-batch and this process is therefore called **Batch Normalization**. To increase the stability of a neural network, batch normalization normalizes the output of a previous activation layer by subtracting the batch mean and dividing by the batch standard deviation. However, after this shift/scale of activation outputs by some randomly initialized parameters, the weights in the next layer are no longer optimal. SGD (Stochastic gradient descent) undoes this normalization if it's a way for it to minimize the loss function. Consequently, batch normalization adds two trainable parameters to each layer, so the normalized output is multiplied by a "standard deviation" parameter and add a "mean" parameter. In other words, batch normalization lets SGD do the denormalization by changing only these two weights for each activation, instead of losing the stability of the network by changing all the weights.

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Figure 31, Batch Normalization, applied to activation x over a mini-batch

The four equations shown do the following:

1. Calculate mean (μ_B) of the mini-batch.
2. Calculate variance (σ^2_B) of the mini-batch.
3. Calculate z-normalized input, by subtracting mean and subsequently dividing by standard deviation. A small number, epsilon (ϵ), is added to the denominator to prevent divide by zero.
4. Calculate y_i by multiplying normalized xi with a scale (γ) and adding a shift (β) and use normalized y_i in place of y_i. The extra two parameters β and γ are learned during the training process.

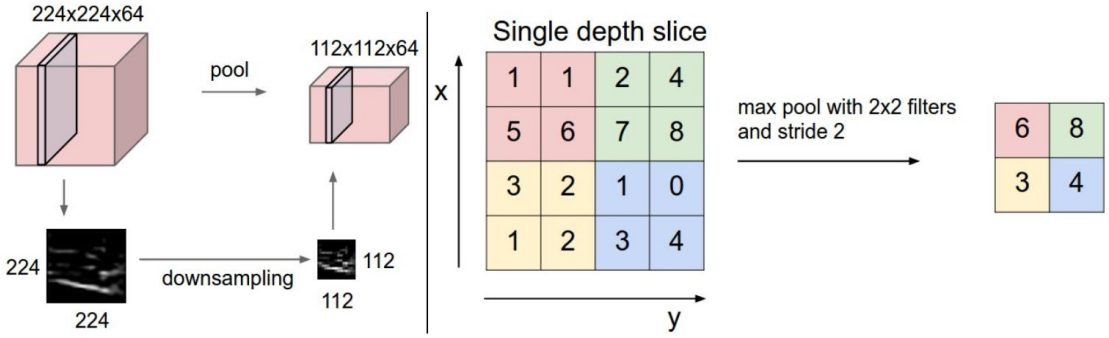
With batch normalization, small changes in parameter to one layer are not propagated to other layers. This makes it possible to use higher learning rates for the optimizers, which otherwise would not have been possible. It also makes gradient propagation in the network

more stable. Finally, the normalization step sees all the training examples in the mini-batch together; it brings in a regularization effect with it.

Pooling Layers

It is common to periodically insert a Pooling layer in-between successive Convolutional layers in a CNN. Its function is to progressively reduce the spatial size of the representation to reduce the number of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every channel of the input and resizes it spatially, most of the times by using the max operation. The most common form is a pooling layer with filters of size 2x2 applied with a stride of two that downsamples every slice in the input by 2 along both width and height, discarding 75% of the activations. Every MAX operation would in this case be taking a max over four numbers (little 2x2 region in some depth slice). The depth dimension remains unchanged. It introduces zero parameters since it computes a fixed function of the input and it induces non-linearity in the network. In addition to max pooling, the pooling units can also perform other functions, such as *average pooling* or even *L2-norm pooling*. Average pooling was often used historically but has recently fallen out of favor compared to the max pooling operation, which has been shown to work better in practice.

Finally, it is worth mentioning the intuition behind a pooling operation. If the output of a single filter in a convolutional layer produces a slice performing cross correlation between the signal and the kernel, by taking the max operation we choose to keep the pixel value of the feature map that locally maximizes the cross-correlation between the signal and the kernel. We assume that the local peak values that we select are of greater importance, which seems to be the case for many computer vision tasks. Therefore, we do not only perform a kind of downsampling, but we also increase the semantic information.



Many people dislike the pooling operation and think that we can get away without it. Instead, to reduce the size of the representation modern approaches suggest using larger stride in the convolutional layers occasionally. Discarding pooling layers has also been found to be important in training good generative models, such as variational autoencoders (VAEs) or generative adversarial networks (GANs). It seems likely that future architectures will feature very few to no pooling layers.

Unpooling Layers

To perform unpooling, we need to remember the position of each maximum activation value when doing max pooling. Then, the remembered position is used for unpooling as shown. Pooling consists in taking local maxima on a map and discarding the rest. Unpooling puts back these maxima into their map location and set the rest to 0.

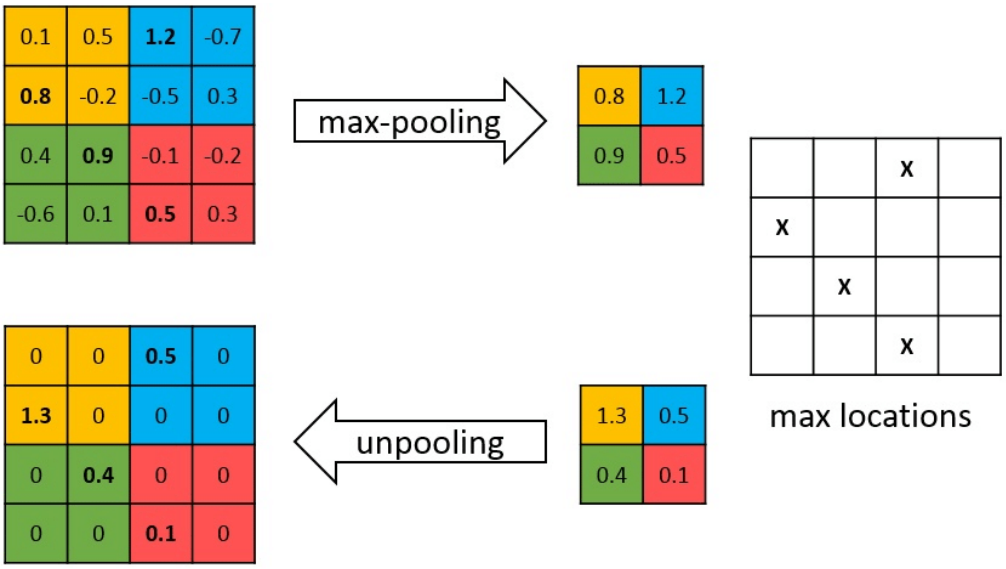


Figure 32, Pooling and unpooling layer

This figure shows the pooling operation with a kernel of size 2 and a stride of 2. A kernel of two means that you look for the local maxima on a block of 2x2. A stride of two means that you take a block every two blocks. Given the pooled maps and indices, we can also perform an unpooling operation, which inserts the pooled values in the appropriate locations in the feature maps, with the remaining elements being set to zero. This step **significantly increases sparsity.**

Skip Connections

Skip architecture as the name suggests skips some layer in the neural network and feeds the output of one layer as the input to the next layers (instead of only the next layer). Usually we do that because there was some information that was captured in the initial layers and we would like to allow the later layers to also learn simple features that are captured in the earlier layers. If we had used the skip connection that information would have turned too abstract for it to be used further. Therefore, the information that we had in the primary layers can be fed explicitly to the later layers using the skip architecture. The intuition behind this type of skip connection is that they have uninterrupted gradient flow from the first layer to the last layer, which tackles the vanishing gradient problem. Less literally, **skip connections enable feature reusability.**

Dropout Layer

Dropout layers [9] have a very specific function in neural networks. We have already discussed the problem of overfitting, where after training, the weights of the network are so tuned to the training examples they are given that the network does not perform well when given new examples. The idea of dropout is simplistic in nature. **This layer "drops out" a random set of activations in that layer by setting them to zero.** Now, what are the benefits of such a simple and seemingly unnecessary and counterintuitive process? It forces the network to be redundant. This results in the network to be able to provide the right classification or output for a specific example even if some of the activations are dropped out. It makes sure that the network is not being too "fitted" to the training data and thus helps alleviate the overfitting problem. An important note is that this layer is only used during training, and not during test time.

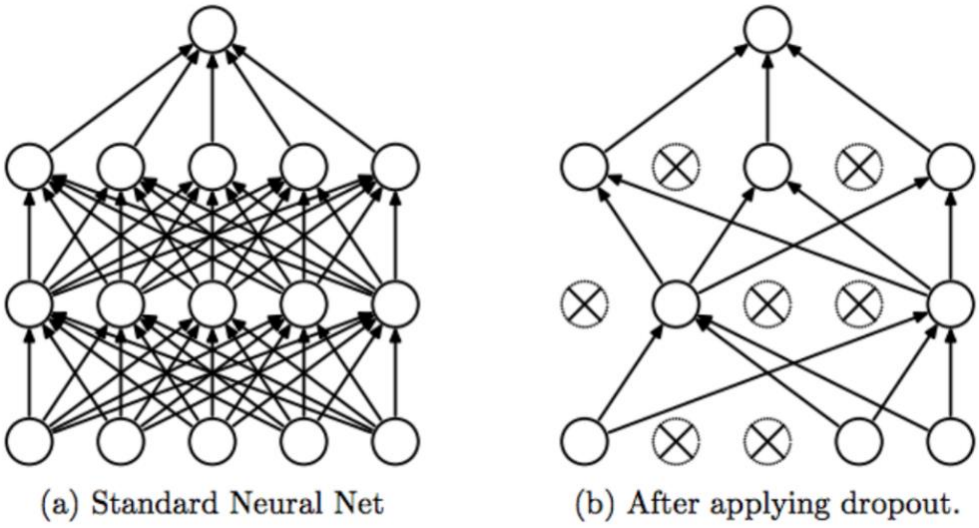


Figure 33, Dropout visualization

Putting it all together

Putting all the components that were described together, an DNN architecture is designed. An illustration of a relatively simple architecture is depicted in Figure 33. The Convolutional layers are capturing multiple correlation in the image. Convolutional layer is usually followed by batch normalization and non-linear activations. After that pooling operation is usually applied. This results in the reduction spatial dimension and the increase of the channel dimension. The final vector is passed in a Linear layer for classification.

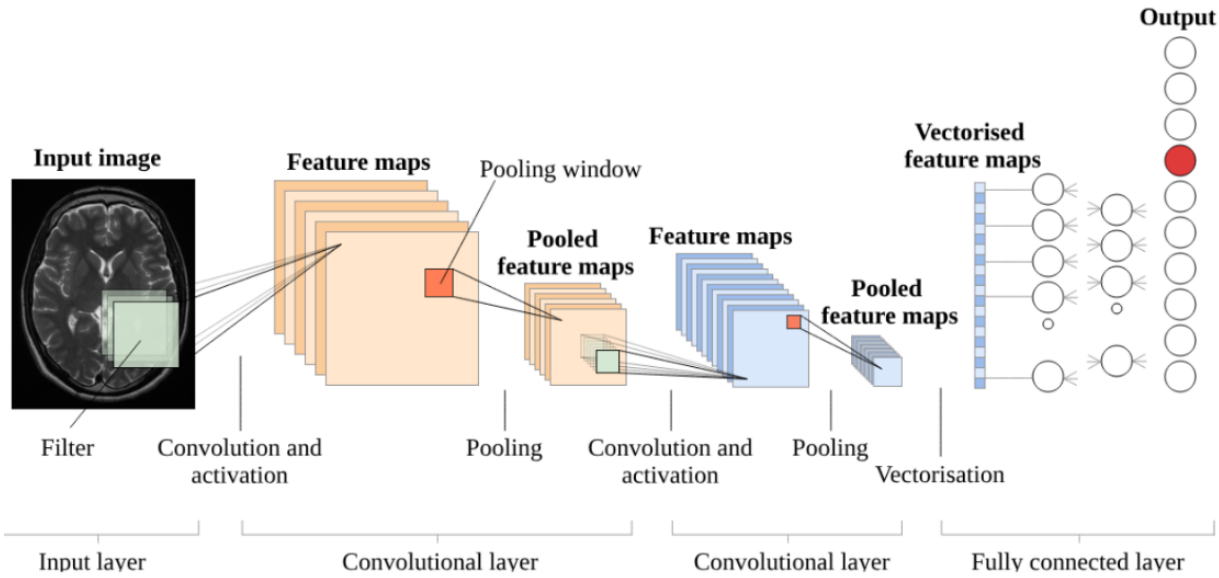


Figure 34, Putting it all together

Chapter summary

In this chapter, we gave the reader with a broad overview of machine learning principles. We understood the expressive power of DNN's and common model training behavior, as well as the goal of generalizing in unseen data. A detailed overview of the basic components that will be extensively used in the next chapters is given. It is significant in the understanding of deep architectures to understand the intuition behind the mathematic operations in order to reason the validation results. The process of training a DNN requires careful observations in order to minimize the tiring process of trial and error, especially when training such a model requires many hours.

CHAPTER 3 - Common DNN Architectures

Introduction

The DNNs that will be presented have dominated the image recognition challenges the past years. There are many variants of these networks, but we will discuss the basic concepts and intuitions of the first-published papers. All these networks can be found pretrained in image classification datasets in the most common deep learning frameworks. Since they have a huge number of parameters, it is obvious that they are hard to trained from scratch. The most adopted strategy is to fine-tune the last layers for the specific task. These families of networks are usually called feature extractors or backbone networks. They extract meaningful and abstract feature representations for RGB images. The problem that will be discussed again is that in medical imaging there are no such common feature extractor networks because the input image consists of many channels. Finally, image segmentation networks will be discussed.

3.1) GoogLeNet (InceptionNet)

GoogLeNet [37] won the image classification challenge in 2014 and won again in 2016 after they iterated on it (Inception v4). The real novelty of this network is the **Inception Modules**. The network itself simply stacked these inception modules together. The motivation behind these inception modules was around the issue of selecting the right filter or kernel size. We always prefer to use smaller filters, like 3×3 or 5×5 or 7×7, but which ones of these works the best? Depending on how deep we make our network, each convolutional layer has a choice between 3 different filter sizes. Instead of choosing just a single filter size, choose all of them and concatenate the results. Taking the activation maps of features from the previous layer, we apply 3 separate convolutions and one pooling on top of that single input and concatenate the resulting feature maps together.

There is one issue with this approach: computational complexity. An inception module is slow for high-dimensionality input. The solution is to convert that high-dimensionality input to a lower dimensionality using 1×1 convolutions. The improved inception modules use these dimensionality-reducing 1×1 convolutions before applying the 3×3 and 5×5 convolutional layers. We also perform dimensionality reduction after the max pooling layer as well, for the same reason.

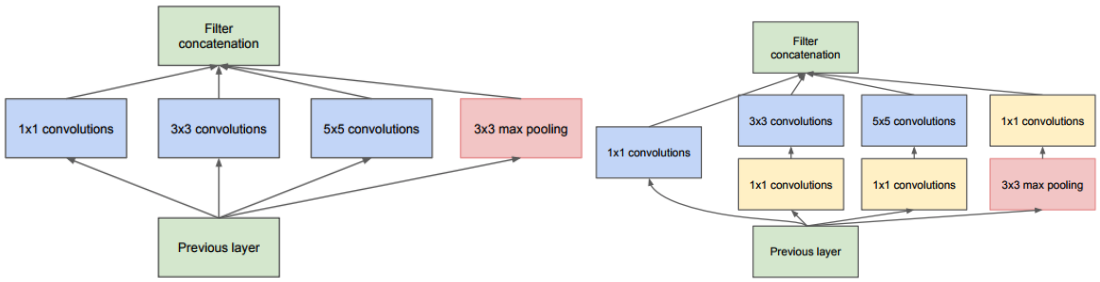


Figure 35, Left: Naïve version , Right: Inception with channel dimensionality reduction

3.2) ResNet

The network that won image classification challenge in 2015 is also the deepest network to this day: ResNet [6], or **residual networks**. The major issue with taking something like GoogLeNet and extending it many layers is the **vanishing gradient problem**. With any network, we compute the error gradient at the end of the network and use backpropagation to propagate our error gradient backwards through the network. Using the chain rule, we must keep multiplying terms with the error gradient as we go backwards. However, in the long chain of multiplication, if we multiply many things together that are less than one, then the result will be very small. This applies to the gradient as well: the gradient becomes very small as we approach the earlier layers in a deep architecture. This small gradient is an issue because then we cannot update the network parameters by a large enough amount and training is very slow. In some cases, the gradient becomes zero, meaning we do not update the earlier parameters at all! Whenever we backpropagate through an operation, we must use the chain rule and multiply, but what if we were to backpropagate through the identity function? Then the gradient would simply be multiplied by one and nothing would happen to it.

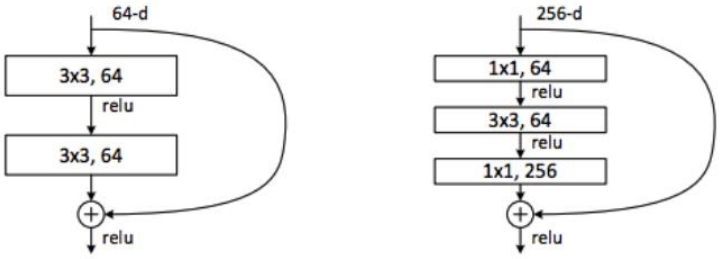


Figure 36, The building block of ResNet

This is the idea behind ResNet: it stacks these **residual blocks** together where we use an identity function to preserve the gradient. Residual connections directly refer to the skip connections we described earlier. The beauty of residual blocks is the simplicity behind it. We take our input, apply some function to it and add it to our original input. Then, when we take the gradient, it is simply one. Mathematically, we can represent the residual block, and calculate its partial derivative (gradient), given the cost function C, like this:

$$H(x) = F(x) + x$$

$$\begin{aligned}\frac{\partial C}{\partial x} &= \frac{\partial C}{\partial H}\frac{\partial H}{\partial x} \\ &= \frac{\partial C}{\partial H}\left(\frac{\partial F}{\partial x} + 1\right) \\ &= \frac{\partial C}{\partial H}\frac{\partial F}{\partial x} + \frac{\partial C}{\partial H}\end{aligned}$$

Figure 37, Residual connections and the vanish gradient problem

With that addition, the gradient is less likely to go to zero and we simply propagate the complete gradient backwards. These residual connections act as a “gradient highway” since

the gradient distributes evenly at sums in a computation graph. This allows us to preserve the gradient as we go backwards. Additionally, we can have bottleneck residual blocks, as shown on the right-side of the figure. We use those 1×1 convolutions again to reduce dimensionality before and after the middle convolutional layer. It turns out that these residual blocks are so powerful that we can stack many of these to produce networks that are over 5 times deeper than before! The deepest variant of ResNet was ResNet 151. ResNets revolutionized deep architectures with such simple ideas.

3.3) DenseNet

The most recent new architecture is from Facebook AI Research (FAIR) and won best paper at the most prestigious computer vision conference: Computer Vision and Pattern Recognition (CVPR) in 2017. Their architecture was titled **DenseNet** [7]. Like GoogLeNet and ResNet before it, DenseNet introduced a new block called a **Dense Block** and stacked these blocks on top of each other, with some layers in between, to build a deep network.

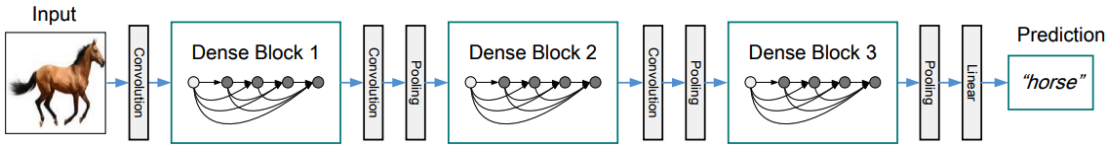


Figure 38, The Dense block

These dense blocks take the concept of residual networks a step further and connect every layer to every other layer. In other words, for a dense block, we consider all dense block before it as an input and we produce an output that we feed into all subsequent dense blocks. To make the layers compatible with each other, we apply convolutions and batch normalizations. The benefit of doing this is **feature reusability**, which resolves the vanishing gradient problem. This idea, counter-intuitively, reduces the overall number of trainable parameters.

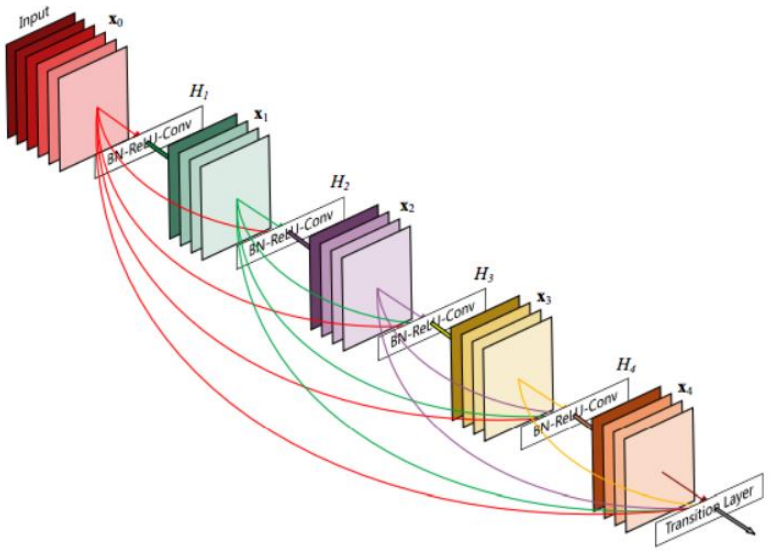


Figure 39, Dense net architecture

3.4) Fully Convolutional Network (FCN)

In this work [15] authors propose a fully convolutional network (FCN) trained end-to-end, on semantic segmentation. This was the first work to train FCNs end-to-end (1) for pixelwise prediction. Both learning and inference were performed in the whole image by dense feedforward computation and backpropagation. The usage of in-network upsampling (transpose convolutional) layers enable pixel-wise prediction and learning in nets with subsampled pooling.

Semantic segmentation faces an inherent tension between semantics and location: global information resolves **what** while local information resolves **where**. Deep feature hierarchies encode location and semantics in a nonlinear local-to-global pyramid. A skip architecture is defined to take advantage of this feature spectrum that **combines deep, coarse, semantic information and shallow, fine, appearance information.**

In classification problems, conventionally, an input image is spatially downsized and goes through the convolution layers and fully connected (FC) layers, and output one predicted label for the input image. If the image is not downsized, the output will not be a single label. Instead, the output has a size smaller than the input image (due to the max pooling). But, if we upsample the output above, then we can calculate the pixelwise output (label map) as below:

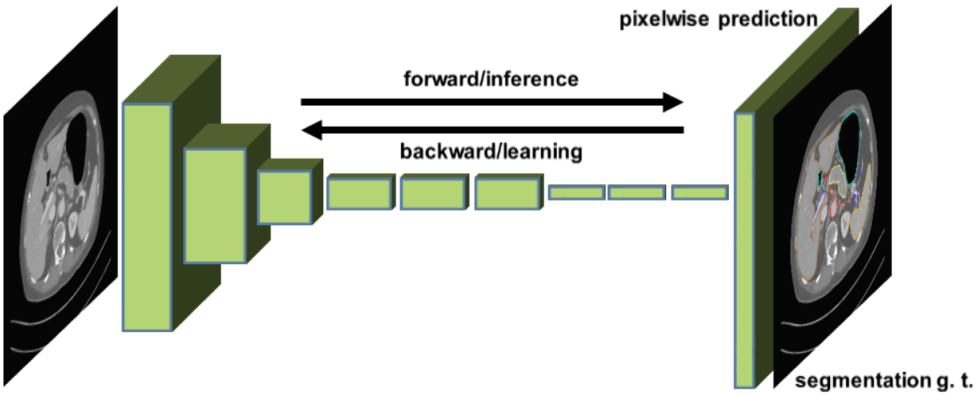


Figure 40, Fully convolutional networks can efficiently learn to make dense per pixel predictions

Upsampling is performed with one or more transposed convolutions. Such an operation is trivial to implement, since it simply reverses the forward and backward passes of convolution. Thus, upsampling is performed in-network for end-to-end learning by backpropagation from the pixel-wise loss. Note that the transpose convolution filter in such a layer need not be fixed (e.g., to bilinear upsampling), but can be learned. A stack of transpose convolution layers and activation functions can even learn a nonlinear upsampling that generated pixel-wise predictions. It has been proven that in-network upsampling is fast and effective for learning **dense prediction**. The best segmentation architecture uses these layers to learn to up-sample for refined predictions.

3.5) U-net: Convolutional Networks for Biomedical Image Segmentation

In this paper [1], authors build upon the "Fully Convolutional Network" (FCN) [15], that was previously described. The goal was to modify and extend FCN such that it works with very few training images. The main idea of FCN, as we saw, is to supplement a usual contracting network(also call encoder network- left half of Figure 42) by successive layers, where pooling operators are replaced by transpose convolution layers. Hence, these layers increase the spatial resolution of the output. In order to provide a localization, high resolution features from the contracting path are combined with the up-sampled outputs, via skip connections. The decoding part of the network (also called expansive path- right half of Figure 42) has many feature channels, which allows the network **to propagate context information to higher resolution layers**. As in FCN, U-net does not have any fully connected layers and **the segmentation map only contains the pixels**, for which the full context is available in the input image. This is the so-called patch-overlap strategy, which allows the segmentation of arbitrarily large images. In order to predict the pixels in the border region of the image, as shown in the yellow region, the missing context is extrapolated by mirroring the input image. Using this strategy, it was possible to apply the network to large images, since otherwise the resolution would be limited by the GPU memory.

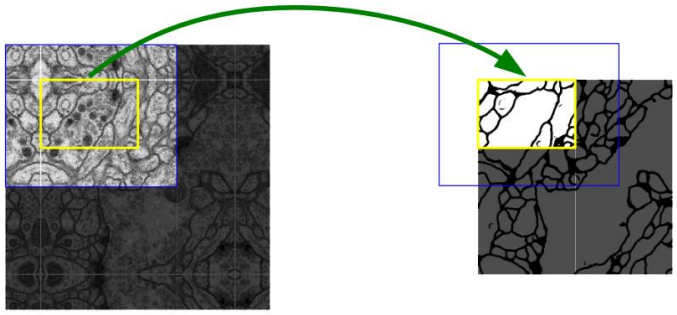


Figure 41, Patch overlap strategy

Because this network was used for the task of medical image segmentation, where the size of the available training data is relatively small, authors used excessive data augmentation by applying elastic deformations to the available training data. Since in such application there are significant elastic deformations, this data augmentation approach provides invariance. Finally, the known problem of segmentation bordering authors proposed the use of a modified weighted loss, where the separating background labels between touching cells obtain a large weight in the loss function, defined as:

$$w(\mathbf{x}) = w_c(\mathbf{x}) + w_0 \cdot \exp\left(-\frac{(d_1(\mathbf{x}) + d_2(\mathbf{x}))^2}{2\sigma^2}\right)$$

Where x is the position of the pixel, w_0 and σ^2 are constants and d_1 and d_2 are the distances of the first and second closest borders respectively. w_c is the weighted map of x that acts as label smoothing. This is implemented for all x pixels that belong to the image segmentation map.

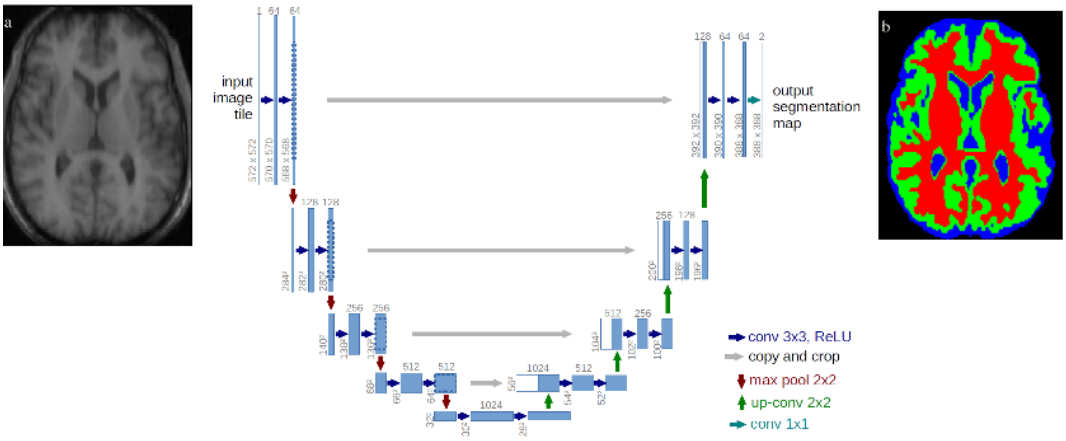


Figure 42, Overview of U-net architecture (example for 32x32 pixels in the lowest resolution).

The Contraction path (encoder) consists of a repeated application of a 3x3 convolutions(unpadded) each followed by a ReLU and a 2x2 max pooling operation with stride 2 for downsampling. At each downsampling step, we double the number of feature channels. This captures **context** via a compact feature map. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations. Every step in the **expansive path(decoder)** consists of a 2x2 transpose convolution of the feature maps, that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU. The cropping is necessary due to the loss of border pixels in every convolution. At the final layer, a 1x1 convolution is used to map each feature channels to the desired number of classes. However, this architecture was used for slice segmentations tasks. That means, in order to process images sizes of 256x256 or more that you only use one slice per time from the MRI scan.

3.6) V-net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation

In this work [2], taking the idea of U-net one step further, authors present a novel approach to medical image segmentation task. As in U-net [1] this architecture leverages the expressive power of a fully convolutional neural network, similar to FCN and is trained in an end-to-end manner. The difference is that this network **processes MRI volumes**, differently from most segmentation approaches that process the input volumes slice-wise. To achieve that, volumetric convolutions are used instead. At the end of each 3D convolution layer, spatial resolution is reduced by using appropriate stride. This is performed through convolution with 2×2×2 voxels wide kernels applied with stride 2, instead of max pooling layers. Similar to the ResNet approach, in each stage, comprising of one to three convolutional layers a residual skip connection is formed.

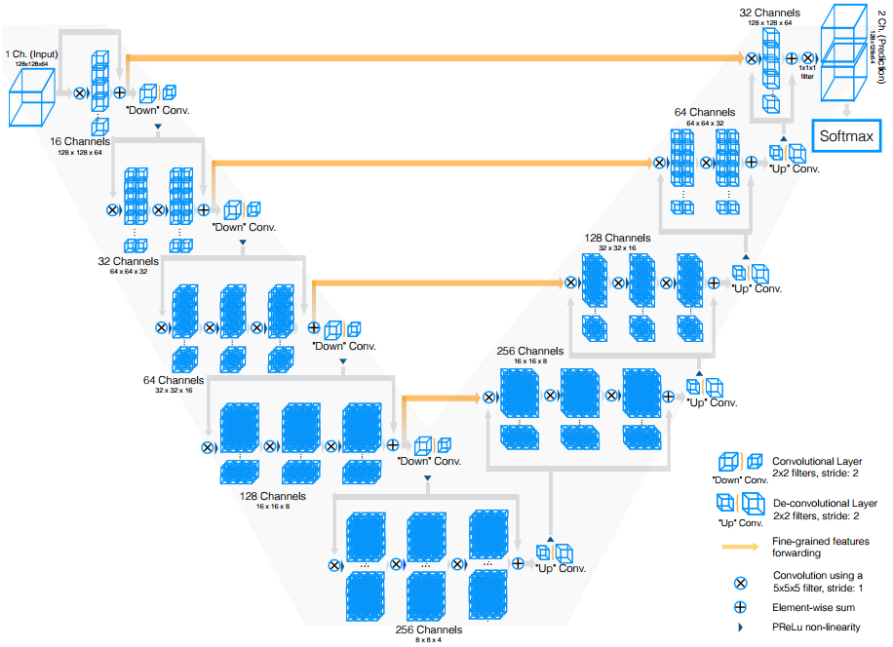


Figure 43, V-net architecture

In the direction of the majority of CNNs, the left half of the picture allows to reduce the spatial size of the input, while increasing the number of features by a factor of two. On the other hand, the right part of the network extracts features and expands the spatial resolution in order to gather and assemble the necessary information to output an N-channel volumetric segmentation (where N is the number of classes). Like U-net, the produced outputs are of the same size as the input volume. Output volumes are converted to probabilistic segmentations by applying soft max but voxel-wise, instead of pixel-wise. Furthermore, a new loss function based on Dice coefficient maximization is introduced.

$$D = \frac{2\sum_i^N p_i g_i}{\sum_i^N p_i^2 + \sum_i^N g_i^2}$$

3.7) 3D-Unet

Like the standard 2D U-net, 3D-Unet [45] has an analysis and a synthesis path each with four resolution steps. In the analysis path, each layer contains two 3 ×3 ×3 convolutions each followed by a rectified linear unit (ReLu), and then a 2 ×2 ×2 max pooling with strides of two in each dimension [45]. In the synthesis path, each layer consists of a transpose convolution of 2 ×2 ×2 by strides of two in each dimension, followed by two 3 ×3 ×3 convolutions each followed by a ReLu. Shortcut connections from layers of equal resolution in the analysis path provide the essential high-resolution features to the synthesis path. In the last layer, a 1×1×1 convolution reduces the number of output channels to the number of labels. Bottlenecks are avoided by doubling the number of channels already before max pooling. 3D batch normalization is introduced before each ReLU. Each batch is normalized during training with its mean and standard deviation and global statistics are updated using these values. This is followed by a layer **to learn scale and bias explicitly**. At test time, normalization is done via these computed global statistics and the learned scale and bias. The Fig. below illustrates the network architecture.

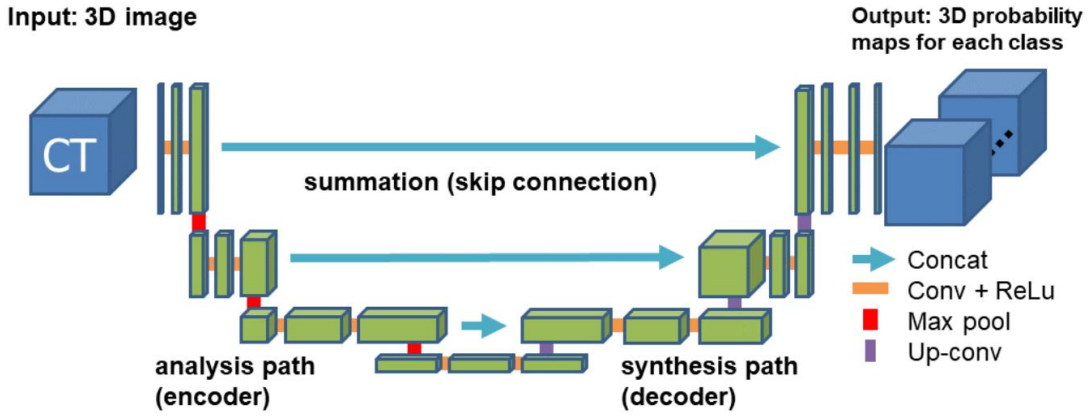


Figure 44, The architecture of 3D U-Net

3.8) 3D-DenseNet

Dense connections have attracted substantial attention in computer vision because they facilitate gradient flow and implicit deep supervision during training. Particularly, DenseNet, which connects each layer to every other layer in a feed-forward fashion, has shown impressive performances in natural image classification tasks. A 3D fully convolutional neural network that extends the definition of dense connectivity to multi-modal segmentation problems [5].

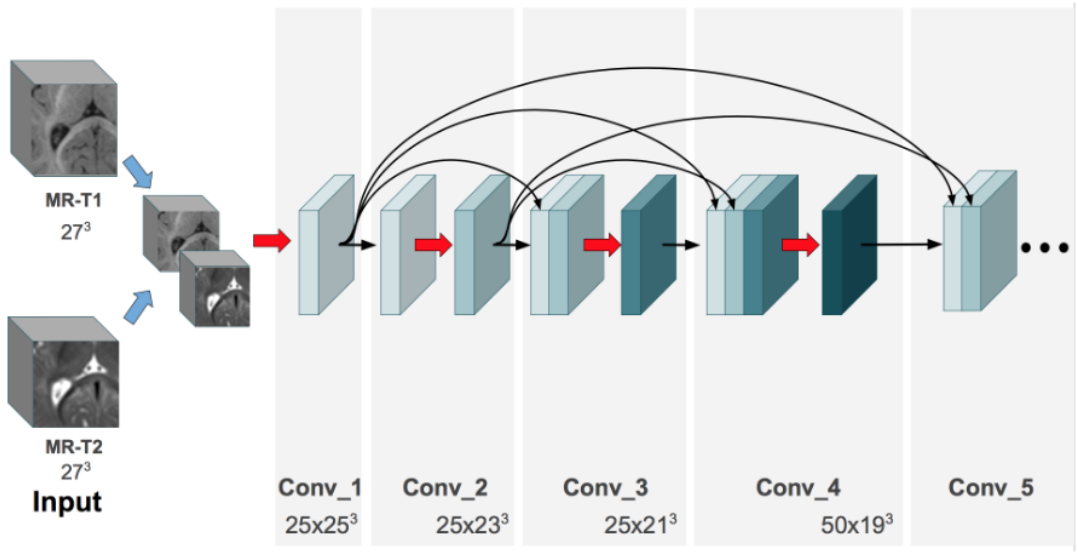


Figure 45, 3D-Densenet

The concept of "the deeper the better" is considered as a key principle in deep learning. Nevertheless, one obstacle when dealing with deep architectures is the problem of vanishing/exploding gradients, which hampers convergence during training. To address these limitations in very deep architectures. DenseNets are built on the idea that adding direct connections from any layer to all the subsequent layers in a feed-forward manner makes training easier and more accurate [5]. This is motivated by three observations. First, there is an implicit deep supervision thanks to the short paths to all feature maps in the architecture. Second, direct connections between all layers help improving the flow of information and gradients throughout the entire network. Third, dense connections have a regularizing effect, which reduces the risk of over-fitting on tasks with smaller training sets.

	Conv. kernel	# kernels	Output Size	Dropout
conv_1	3×3×3	25	25×25×25	No
conv_2	3×3×3	25	23×23×23	No
conv_3	3×3×3	25	21×21×21	No
conv_4	3×3×3	50	19×19×19	No
conv_5	3×3×3	50	17×17×17	No
conv_6	3×3×3	50	15×15×15	No
conv_7	3×3×3	75	13×13×13	No
conv_8	3×3×3	75	11×11×11	No
conv_9	3×3×3	75	9×9×9	No
fully_conv_1	1×1×1	400	9×9×9	Yes
fully_conv_2	1×1×1	200	9×9×9	Yes
fully_conv_3	1×1×1	150	9×9×9	Yes
Classification	1×1×1	4	9×9×9	No

Figure 46, 3D DenseNet architecture

3.9) 2-stream 3D-DenseNet (early and late fusion)

Late fusion 3D-DenseNet

Each imaging modality has a unique path, as illustrated in Figure 47. The proposed networks have the freedom to learn more complex combinations between the modalities, between all the levels of abstraction, which increases significantly the learning representation. Features maps of all previous layers are concatenated. Before the final classification layer, features are further concatenated and fed to the 1x1x1 classifier that outputs volumes as the number of classes.

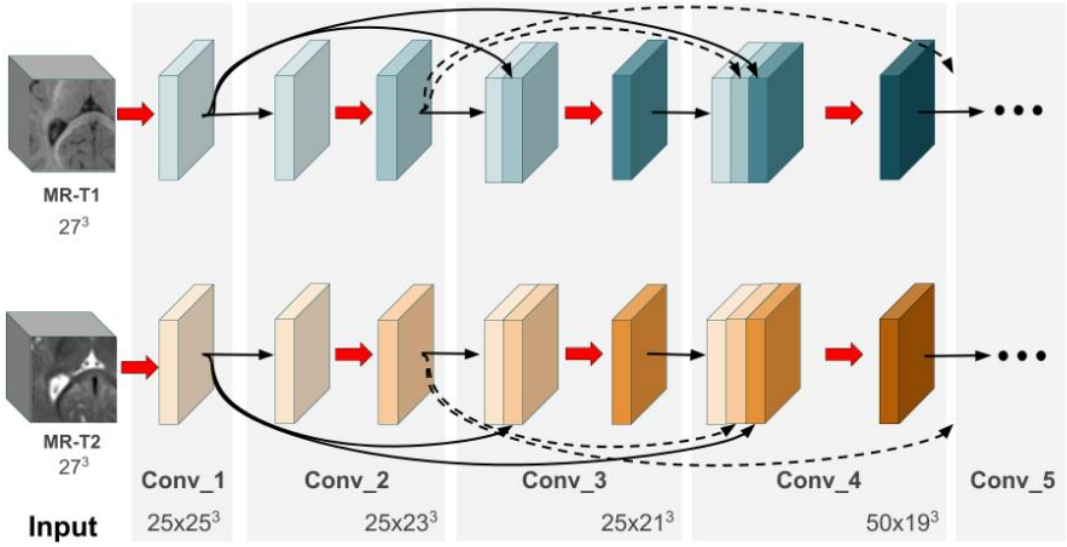


Figure 47, Late Fusion 2-stream architecture

Early fusion 3D-DenseNet

The significant difference in the early fusion setup is that instead of concatenating the abstract features of the last layers, we fuse simpler features of the first 3D convolutional networks.

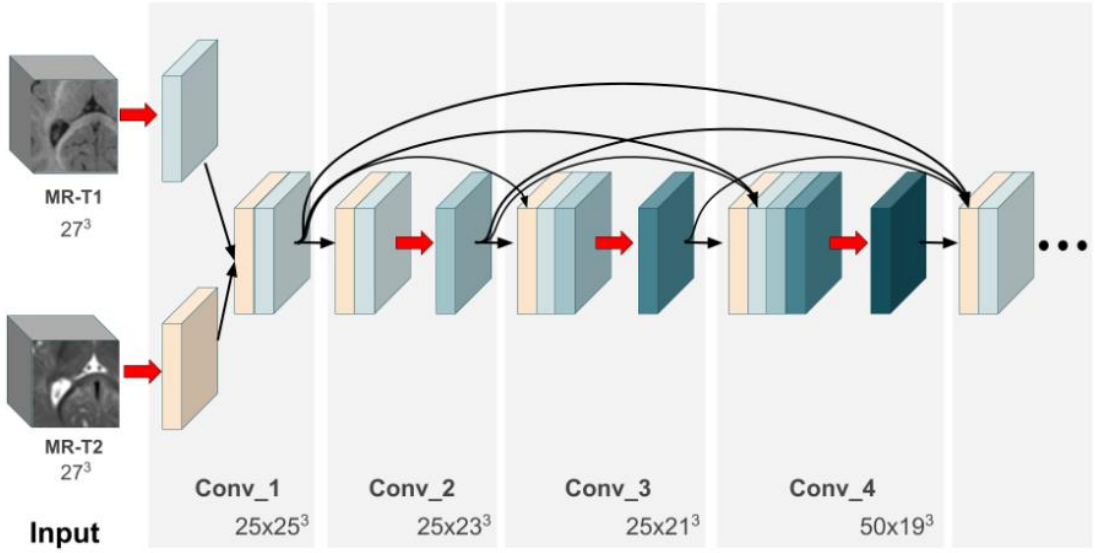


Figure 48, Early Fusion 2 stream 3D-Densenet

Authors [5] results indicate that processing multi-modal data in separate paths, while allowing dense connectivity between all the paths, increases performance over early and late fusion, as well as over early fusion performed after the first convolutional block.

The strong reuse of early features from both paths occurred across all tested configurations on both datasets. The same pattern is observed when using **three modalities in MrBrains dataset**, with a strong reuse of shallow features. This reflects the importance of giving deep layers access to early-extracted features. Finally, it suggests that learning how and where to fuse information from multiple sources is more effective than combining these sources in early or late stages.

Chapter Summary

In this chapter, we provide an overview of common deep architectures starting from conventional 2D networks. Then, we focus on 3D architectures, as they are well suited for MRI analysis. More specifically, we describe the semantic segmentation architectures that will be utilized in the comparative experimental evaluations.

CHAPTER 4 - Deep learning in Medical Imaging & MRI

Introduction

This chapter refers briefly in the wide application areas of DNN in MRI. Many image diagnosis tasks require initial search to identify abnormalities, quantify measurement and changes over time. Deep learning methods are increasingly used to improve clinical practice. In the field of MRI, deep learning has seen applications at each step of entire workflows. From acquisition to image retrieval, from segmentation to disease prediction. In the context of this thesis, we will divide the aspects of deep learning in MRI into two parts: **(i) the signal processing chain close to the physics of MRI**, including image restoration and multimodal image registration and **(ii) the use of deep learning in MR reconstructed images**, such as medical image segmentation. Deep learning in MRI has typically been focused on segmentation and classification of reconstructed magnitude images, as we will also do in this work.

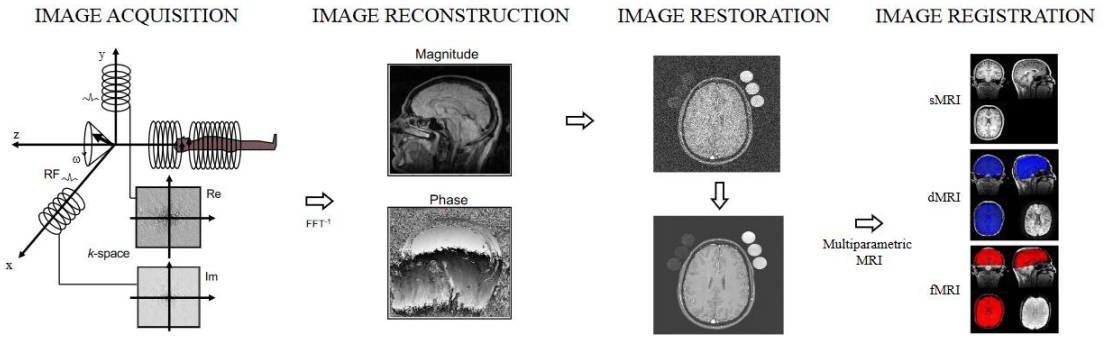


Figure 49, Aspects of Deep Learning applications in the signal processing chain of MRI

We will attempt to provide the reader with a brief overview of three works of Deep Learning applications in MR imaging, apart from medical image segmentation: a) Medical Image Reconstruction, b) Medical Image Synthesis and Denoising, and c) Medical Image Super-resolution.

4.1) Medical Image Reconstruction

One of the first works in the field was by [31]. Authors of this work propose a framework for reconstructing dynamic sequences of 2D cardiac magnetic resonance (MR) images from under-sampled data using a deep cascade of convolutional neural networks (CNNs) to accelerate the data acquisition process. Each 2D image frame was reconstructed independently. The proposed method outperforms 2D compressed sensing approaches in terms of reconstruction error and reconstruction speed. Reconstructing the frames of the sequences jointly, results in learning spatio-temporal correlations efficiently by combining convolution and data sharing approaches. We show that the proposed method consistently outperforms state-of-the-art methods and can preserve anatomical structure.

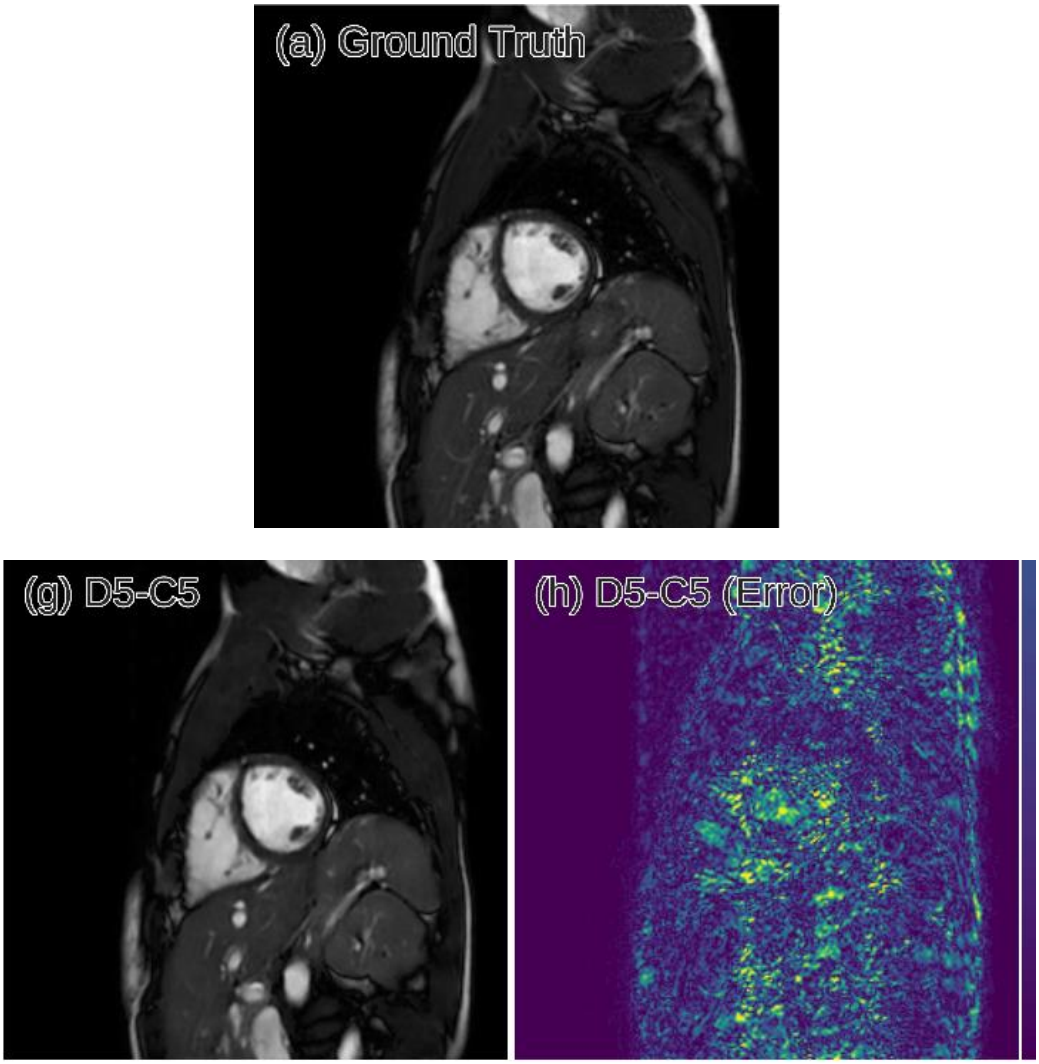


Figure 50, a) ground truth, g) DNN output, h) relative error

4.2) Medical Image Synthesis and Denoising

We will briefly describe the work proposed by [32], which was done to extract quantitative information from the acquired images in order to make observations about the presence of disease or markers of development in populations. Having a low-dimensional manifold of an image allows for easier statistical comparisons between groups and the synthesis of group representatives. Many studies have sought to identify the best mapping of brain MRI to allow multi-dimensional manifold. Authors used deep learning techniques to investigate **implicit manifolds of normal brains** and generate new, high-quality images. Implicit manifolds by addressing the problems of image synthesis were explored. They also tackled image denoising with deep learning networks. They produced T1-weighted brain MRI images using a Generative Adversarial Network (GAN) by learning from 528 examples of 2D axial slices of brain MRI.

Synthesized images were first shown to be unique by performing a cross-correlation with the training set. Real and synthesized images were then assessed in a blinded manner by two imaging experts providing an image quality score of 1-5. The quality score of the synthetic image showed substantial overlap with that of the real images. Moreover, an autoencoder

with skip connections for image denoising was used, showing that the proposed method in order to denoise the image. This was one of the first works in medical imaging that showed the power of deep networks to synthesize realistic medical imaging data, which could be used to improve image processing techniques and provide a quantitative framework to structural changes in the brain.

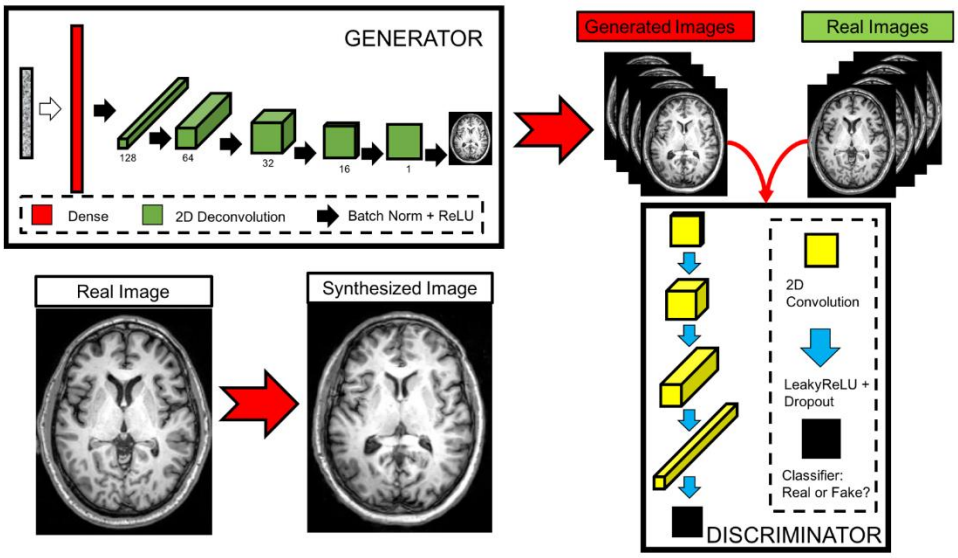


Figure 51, Using GANs to generate realistic normal brain MRI images

Of note, the first rater, a neuro-radiologist, mentioned that despite comparable quality, **the synthetic images were immediately given away by anatomic abnormalities such as largely asymmetric left and right caudate**. Similarly, the second rater, a neuroimaging expert, noticed brighter intensities near the center of the image compared to the boundaries in the synthetic images. This would immediately be noticed after acquisition by the technician and reported as a hardware issue. These comments represent challenges in image synthesis: anatomic accuracy and signal quality.

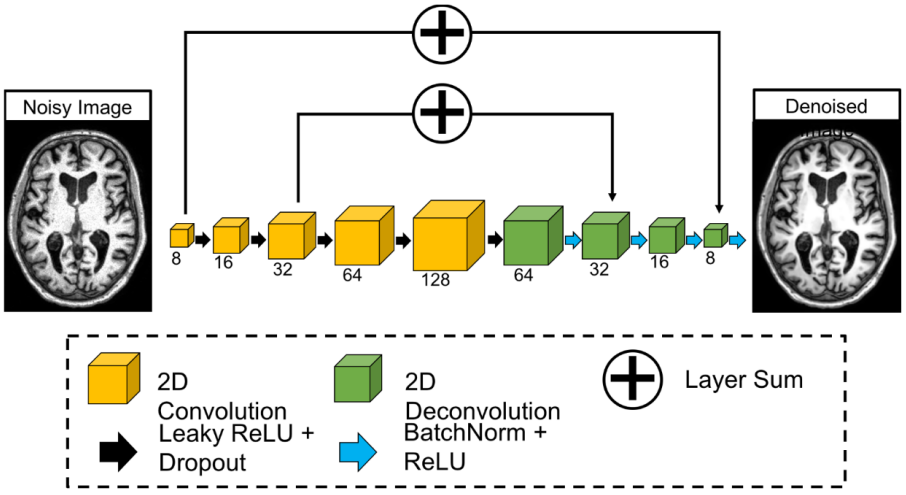


Figure 52, Using autoencoders for image denoising

To summarize, further work is needed to refine the subset that truly represents possible MRI images. However, **exploration of these unrealistic synthesized images may shed a light on possible structural and functional variants in brain anatomy found in healthy individuals or disease**. Secondly, the problem of signal quality is essential to manifold learning, since it allows for better distinction between subjects. Physical restrictions can again be applied to make the image intensity homogenous and realistic.

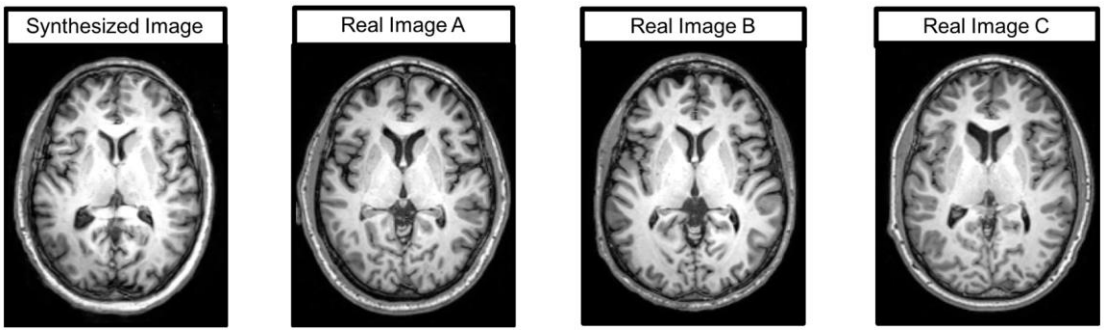


Figure 53, A representative synthesized image, as well as three real images with highest correlation values

4.3) Medical Image Super-resolution

A significant work in medical image super resolution is performed by [33]. The proposed network is able to learn end-to-end mapping from low/high-resolution images. Simultaneously, due to the fusion of different paths, the network can extract multi-scale information to recover detailed information and accelerate the convergence speed. The contributions of this work [33] include the following three aspects:

•It is illustrated that different convolution responses using different convolution kernel sizes, demonstrated that fusing different responses was beneficial for recovering detailed information from a low-resolution image. Conventional CNNs may learn different scale information from different convolution layers, but they are unable to integrate different scale information and decrease the error during the back-propagation procedure.

•To integrate multi-scale information induced by different convolution layers, they stacked a multi-scale fusion unit, which is a full convolution network that is capable of learning end-to-end mapping between low and high-resolution images. In this way, this unit makes full use of prior knowledge from high-resolution images and uses multi-scale information to infer missed details in low-resolution images. This network exhibits an outstanding performance in MRI reconstruction. The proposed network also has a faster convergence speed than the traditional convolution network. This network is capable of learning feature maps and provides exact guidance for the design of network architecture.

• They found that a larger kernel size, an increased number of kernels, and a deeper structure are beneficial for improving the reconstruction performance. However, these features increase the computational burden and converge more slowly. Considering the ideal trade-off between performance and speed, the adopted network structure has achieved superior performance.

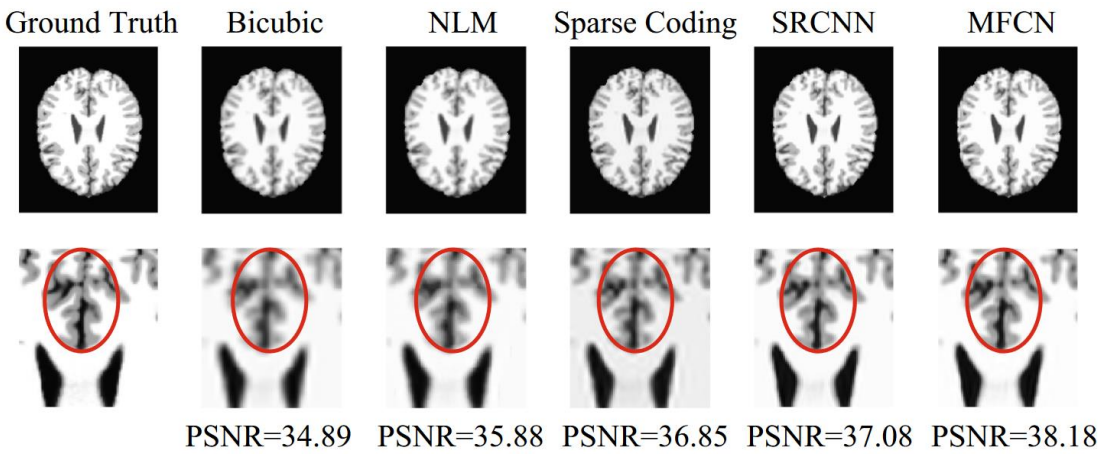


Figure 54, Visual comparison of different methods using Brain Data

4.4) Medical Image segmentation

Now that we saw some application of Deep Learning in MRI, we will focus on MRI segmentation focused on brain MRI. Image segmentation, a significant section of quantitative image analysis, **is the process of partitioning an image into multiple regions that share similar attributes, enabling localization and quantification**. It has an enormous history and has become the biggest target for deep learning approaches in medical imaging. An impressive range of segmentation methods and approaches has been reported, especially for brain segmentation. MR image segmentation using deep learning approaches, are penetrating the whole field of applications.

Brain tissue classification or segmentation **is used for detection and diagnosis of normal and pathological tissues such as multiple sclerosis tissue abnormalities and tumors.** These abnormalities could be identified by tracking of changes in volume, shape and regional distribution of brain tissue during follow-up examinations of patients. Also, some of the neurological and psychiatric disorders such as Alzheimer's, Parkinson's and Huntington's disease, depression, autism, can be diagnosed with detection of changes in the morphology of subcortical nuclei and the cerebellum [34]. Furthermore, brain image segmentation plays an important role in clinical diagnostic tools and treatment procedures such as diagnosis and follow-up and 3D brain visualization for measuring the volume of different tissues in brain such as Gray and White Matter, Thalamus, Amygdala, Hippocampus etc. [35]. However, some authors try to change the problem to a three-type tissue classification, and they assume multiple gray matter structures as one class. Hence, they label brain volumes into a three main classes like WM, GM, Cerebrospinal fluid (CSF) [36].

Brain images basically contain a lot of artifacts including Partial Volume Effect (PVE), Intensity Non-Uniformity (INU) and some noises and deviations. PVE is happen when multiple tissues are placed into a voxel and a mixing value is laid into each voxel, thus a wrong value is assigned to each voxel. INU happens because of Radio Frequency coil. Hence, an accurate segmentation of brain images is a challenging task. On the other hand, in most cases **an accurate and precise segmentation is crucial for a correct diagnosis by clinical tools.** Moreover, **manual**

segmentation of brain MRI images is a time-consuming and labor-intensive procedure. Therefore, automatic image segmentation is widely used for this purpose.

A significant research has been conducted in the field of automatic segmentation with supervised methods. Two main categories include machine learning-based methods and atlas-based methods. Machine learning based methods use an annotated database which are divided into training and test set, as described in Chapter 2. The learning phase is performed on the training set and the evaluation is done on the test set. Support Vector Machine (SVM) [37] was the most common machine learning approach mostly used in this area, until the rise of DNN's.

Chapter Summary

In this chapter, we provide the reader with a broader overview of the intersection of MR and deep learning. The aim is to give a new perspective, because deep learning learns meaningful representation that is not limited to the high-level processing of final MR 3D reconstructed images. Finally, this chapter serve as a proof for the future collaborations between biomedical engineers, deep learning specialists and radiologist in an interdisciplinary environment.

CHAPTER 5 - Datasets and intuitions

Introduction

DNNs requiring large number of training samples before they can produce anything useful generalized representation and labeled training data is typically both expensive and difficult to produce. In addition, the training data must be representative of the data the network will meet in the future. If the training samples are drawn from a data distribution that is different from the one would meet in the real world, then the network's generalization performance will be lower than expected. Considering the large difference between the high-quality images one typically works with when doing research and the messiness of the real, clinical world, this can be a major obstacle when putting deep learning systems into production.

Luckily there are ways to alleviate these problems. A widely used technique is **transfer learning** [47], also called fine-tuning or pre-training: First you train a network to perform a task where there is an abundance of data, and then you copy weights from this network to a network designed for the task at hand. For two-dimensional images one will almost always use a network that has been pre-trained on the ImageNet data set. The basic features in the earlier layers of the neural network found from this data set typically retain their usefulness in any other image-related task. Starting from weights tuned on a larger training data set can also make the network more robust. Focusing the weight updates during training on later layers requires less data than having to do significant updates throughout the entire network. Consequently, availability of data is an important obstacle for DNNs, especially in medical data analysis, because medical imaging data are limited and difficult to acquire. When deploying DNNs, or any other machine learning model, one is instantly faced with challenges related to data access, privacy issues, data protection, and more. Most works on deep learning for medical data analysis use open & anonymized data. In this Chapter we will introduce the 3 most common brain MRI segmentation datasets as well as an overview of the tissues that we attempt to distinguish. Finally, it is worth mentioning that these medical imaging datasets were created for the MICCAI medical image computing conference, as challenges. Therefore, a lot of researchers from the computer vision industry participated in medical image segmentation challenges.

5.1) Brain tissues

Since we are focusing on brain MRI automatic segmentation it important to briefly describe the basic structures of the brain that DNN's are trying to distinguish **a) White matter, b) Grey matter, c) Cerebrospinal fluid.** The following figure illustrates the segmented tissues in brain MRI slice.

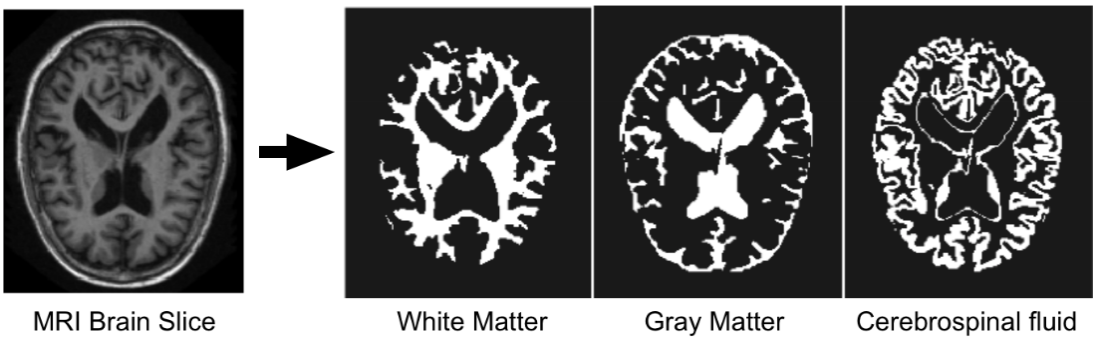


Figure 55 , Left image — middle slice from the brain MRI volume. Right images — segmented images (ground truth data)

- **White matter** refers to areas of the central nervous system that are mainly made up of myelinated axons, also called tracts. White matter is composed of bundles, which connect various gray matter areas (the locations of nerve cell bodies) of the brain to each other and carry nerve impulses between neurons. Myelin acts as an insulator, which allows electrical signals to jump, rather than coursing through the axon, increasing the speed of transmission of all nerve signals. While in the past it was thought to be passive tissue, white matter affects learning and brain functions, modulating the distribution of action potentials and coordinating communication between different brain regions. White matter is named for its relatively light appearance resulting from the lipid content of myelin. However, the tissue of the freshly cut brain appears pinkish white to the naked eye because myelin is composed largely of lipid tissue veined with capillaries.
- **Grey matter** is a major component of the central nervous system, consisting of neuronal cell bodies, dendrites and myelinated as well as unmyelinated axons, glial cells, synapses, and capillaries. Grey matter is distinguished from white matter in that it contains numerous cell bodies and relatively few myelinated axons, while white matter contains relatively few cell bodies and is composed chiefly of long-range myelinated axon tracts. The color difference arises mainly from the whiteness of myelin. In living tissue, grey matter has a very light grey color with yellowish or pinkish hues, which come from capillary blood vessels and neuronal cell bodies. Grey matter is distributed at the surface of the cerebral hemispheres (cerebral cortex) and of the cerebellum (cerebellar cortex), as well as in the depths of the cerebrum.
- **Cerebrospinal fluid** (CSF) is a clear, colorless body fluid found in the brain. It is produced by the specialized ependymal cells in the choroid plexuses of the ventricles of the brain and absorbed in the arachnoid granulations. There is about 125mL of CSF at any one time, and about 500 mL is generated every day. CSF acts as a cushion or buffer for the brain, providing basic mechanical and immunological protection to the brain inside the skull. CSF also serves a vital function in cerebral autoregulation of cerebral blood flow.

5.2) I-Seg challenge 2017

Accurate segmentation of infant brain MR images into white matter (WM), gray matter (GM), and cerebrospinal fluid (CSF) in this critical period is of fundamental importance in studying both normal and abnormal early brain development [39]. The first year of life is the most dynamic phase of the postnatal human brain development, along with rapid tissue growth and development of a wide range of cognitive and motor functions. This early period is critical in many neurodevelopmental and neuropsychiatric disorders, such as schizophrenia and autism. More and more attention has been paid to this critical period. Of note, there are three distinct phases in the first-year brain MRI, including:

1. infantile phase (≤5 months)
2. **isointense phase (6-8 months)**
3. early adult-like phase (≥9 months)

In the isointense phase, the intensity range of voxels in GM and WM are largely overlapping (especially in the cortical regions), thus leading to the lowest tissue contrast and creating the most significant challenge for tissue segmentation, in comparison to images acquired at other phases of brain development. For example, we depict in the Fig. below, longitudinal MR images for an infant scanned every 3 months during the first year, starting from the second week after birth.

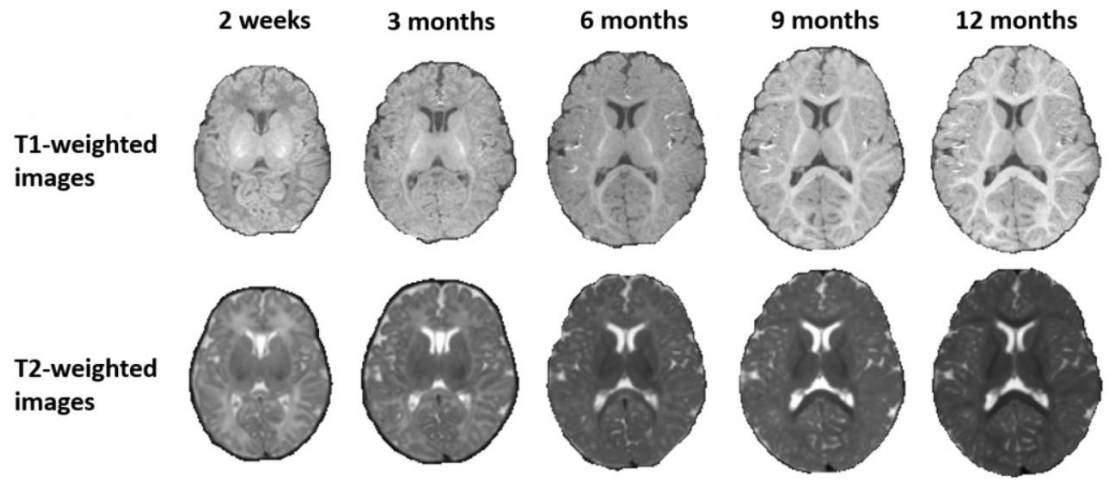


Figure 56, T1- and T2-weighted MR images of an infant scanned.

In this medical image challenge [39], researchers are invited to propose and evaluate their automatic algorithms to segment WM, GM and CSF on isointense (6-month) infant brain MRI scans. The aim of this challenge is to promote automatic segmentation algorithms on 6-month infant brain MRI. This challenge was carried out in conjunction with MICCAI 2017, with a total of 21 international teams. The dataset contains 10 densely annotated images from experts and 13 imaging for testing. Test labels are not provided, and you can only see your score after uploading the results in the official website. For each subject there is T1 weighted and T2 weighted image.

Class	Region
0	Background (everything outside the brain)
1	Cerebrospinal fluid (CSF)
2	Gray matter (GM)
3	White matter (WM)

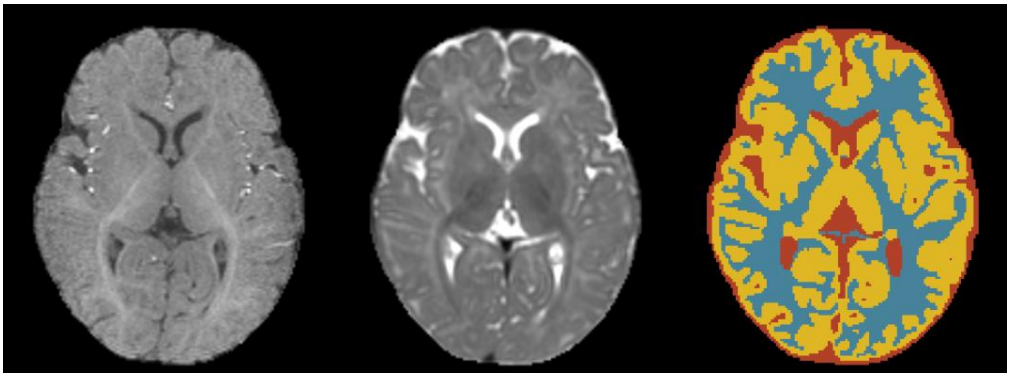


Figure 57, Example of data from a training subject.6-month infant brain images from a mid-axial T1w slice(left), the corresponding T2w slice (middle), and the ground-truth labels (right).

5.3) MR Brains 2018 dataset

The purpose of this dataset [42], which was released in the form of another MICCAI challenge, is to directly compare methods for segmentation of gray matter, white matter, cerebrospinal fluid, and other structures on 3T MRI scans of the adult brain, and to assess the effect of (large) pathologies on segmentation and volumetry. The dataset consists of seven sets of brain MR images (T1, T1 inversion recovery, and T2-FLAIR) with manual segmentations of ten brain structures. These manual segmentations have been made by experts in brain segmentation. The training- and test-data include images with (large) pathologies.

SEQUENCE	DETAILS	TR (MS)	TE (MS)	TI (MS)
T1	3D T1-weighted sequence	7.9	4.5	
T1-IR	Multi-slice T1-weighted inversion recovery sequence	4416	15	400
T2-FLAIR	Multi-slice T2 FLAIR sequence	11000	125	2800

Figure 58, Acquisition information

Image data used in this challenge were acquired on a 3T scanner at the UMC Utrecht (the Netherlands). For each of the 30 subjects, fully annotated multi-sequence (T1-weighted, T1-weighted inversion recovery and T2-FLAIR) scans are available. The 30 subjects include patients with diabetes, dementia and Alzheimer's, and matched controls (with increased cardiovascular risk) with varying degrees of atrophy and white matter lesions (age > 50).

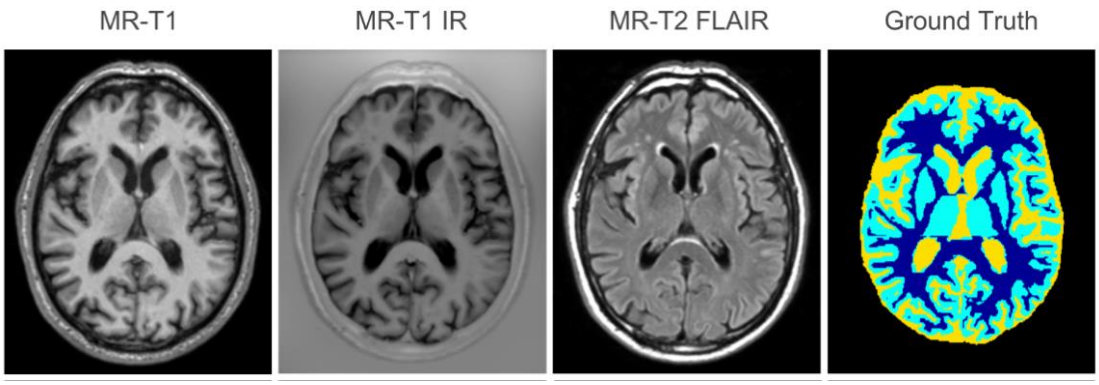


Figure 59, T1 (1st), T1-IR (2nd), T2-FLAIR (3rd), G.T.(4th)

All scans have a voxel size of 0.958mm × 0.958mm × 3.0mm and all are aligned. The scans are bias field corrected using the N4ITK algorithm, but the original data is also available. Some of the T1-IR images contain artifacts at the bottom of the scan. These artifacts occur often in clinical scans, and it is therefore interesting to know how automatic segmentation methods that use these scans perform under these circumstances. The objective of this challenge is to automatically segment labels 1 to 8. Ground-truth labels 9 and 10, i.e. infarctions and 'other' lesions, will be excluded in the evaluation. This means that those voxels will have no effect on the scores; you may label these voxels as gray matter, white matter, or any other label. Some extra notes on the manual segmentations:

- White matter lesions were segmented on the FLAIR scan.

- The outer border of the CSF was segmented using both the T1-weighted scan and the T1-weighted inversion recovery scan.
- All other structures were segmented on the T1-weighted scan (0.958mm × 0.958mm × 3.0mm).
- Vessels were not segmented separately. The CSF segmentation therefore also includes the superior sagittal sinus and transverse sinuses.

LABEL	DESCRIPTION
0	Background
1	Cortical gray matter
2	Basal ganglia
3	White matter
4	White matter lesions
5	Cerebrospinal fluid in the extracerebral space
6	Ventricles
7	Cerebellum
8	Brain stem
9	Infarction
10	Other

Figure 60, Dataset classes

5.4) Dataset preprocessing & augmentation

Mean/std normalization (standardization)

Normalization is a handy technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. For machine learning, every dataset does not require normalization. However, it is usually required only when features have different ranges such as different MRI modalities. The most common approach that we also adopt is to transform input MRI data to have a mean of zero and a standard deviation of one. This standardization is called a **z score**, and data points can be standardized with the following formula:

$$z_i = \frac{x_i - \bar{x}}{s}$$

Bias field correction

Magnetic resonance images often exhibit image intensity non-uniformities that are the result of magnetic field variations rather than anatomical differences. These artifacts, often described as shading or bias, can be produced by imperfections in the field coils used in the systems or by magnetic susceptibility changes at the boundaries between anatomical tissue and air. These variations are often seen as a signal gain change that varies slowly spatially. This can result in white matter measurements in one part of the image with the same intensity as grey matter measurements elsewhere; an ideal T1-weighted image would display brighter white matter. This problem can confound tissue classifiers, which often assume that the image intensities of a given type of tissue are relatively uniform throughout the image.

Bias field signal is a low frequency and very smooth signal that corrupts MRI images specially those produced by old MRI machines [43]. Image processing algorithms such as segmentation, texture analysis or classification that use the grey-level values of image pixels will not produce satisfactory results. This pre-processing step is needed to correct for the bias field signal before submitting corrupted MRI images to such algorithms or the algorithms should be modified.

A well-known intensity inhomogeneity correction method, known as the N4 (nonparametric non-uniformity normalization), was proposed in [46]. The method is iterative and seeks the smooth multiplicative field that maximizes the high frequency content of the distribution of tissue intensity. The method is fully automatic, requires no a priori knowledge and can be applied to almost any MR image. Furthermore, this method attempts to improve the previous baseline algorithm, called N3, by replacing the B-spline smoothing strategy used in the original N3 framework with an advantageous alternative, which addresses major issues.

Image Registration

Image registration is the process of transforming different image datasets into one coordinate system with matched imaging contents, which has significant applications in medicine. Registration may be necessary when analysing a pair of images that were acquired from different viewpoints, at different times, or using different sensors/modalities. Until recently, image registration was mostly performed manually by clinicians. However, many registration

tasks can be quite challenging, and the quality of manual alignments are highly dependent upon the expertise of the user, which can be clinically disadvantageous. To address the potential shortcomings of manual registration, automatic registration has been developed. Although other methods for automatic image registration have been extensively explored, prior to the deep learning renaissance, deep learning has changed the landscape of image registration research.

Generally, the inverse of the affine gives the mapping from scanner to voxel coordinates in the image data. Now imagine we have affine array A for modality A, and affine array B for modality B. A gives the mapping from voxels in the image data array of modality A to millimetres in scanner. B gives the mapping from voxels in image data array of modality B to *the same* scanner. Now let's say we have a particular voxel coordinate (i, j, k) in the data array and we want to find the voxel in another modality that is in the same spatial position. Given that this matching voxel coordinate is called (i', j', k'), we first apply the transform from A voxel space to scanner space and then apply the transform from scanner to the other voxel space in B^{-1}. This geometric transformation is illustrated in Fig. 61.

$$\begin{bmatrix} i' \\ j' \\ k' \\ 1 \end{bmatrix} = B^{-1} A \begin{bmatrix} i \\ j \\ k \\ 1 \end{bmatrix}$$

Figure 61, Medical Image registration through affine transformations

Data augmentation

To have a large receptive field, FCNs like 3D-Unet and Vnet typically use full images as input. The number of parameters is then limited via pooling/unpooling layers. A problem with this approach is the loss of resolution from repeated down-sampling operations. In DenseNet-based approaches, sub-volumes are used as input, **avoiding pooling layers**. While sub-volumes of size 32x32x32 are considered for training, we used 35×35×35 non-overlapping sub-volumes during inference. This strategy offers two considerable benefits. First, it reduces the memory requirements of DenseNet-based networks, thereby removing the need for spatial pooling. More importantly, it substantially increases the number of training examples and, therefore, does **not need data augmentation technique's to be applied**. For the same reasons sub-volumes of 128x128x64 were used in the 3D-Unet and Vnet models. Since sub-volumes were used it is obvious that a sampling technique had to be used. A uniform sampling technique to draw non-empty volumes were performed, because empty volumes lead to training instabilities.

Chapter Summary

In this chapter, we provide information about the used dataset to perform our comparative analysis. Starting from understanding the multiple MR modalities, we end up describing segmentation map regions and data preprocessing pipelines that we used in our analysis.

CHAPTER 6 - Experimental evaluation

6.1) Experimental Setup & Implementation Details

I-SEG 2017 and MrBrains 2018

The first subject of both datasets was used for testing. The original MR volumes are of size 256x192x144 and 240x240x48 for I-SEG and MrBrains. In Vnet & 3D-Unet, the sampled sub-volumes that were used are of size 128x128x64 and 128x128x32, for I-SEG and MrBrains accordingly. The training dataset that was generated consisted of 500 sub-volumes of from K-1 MR images for Vnet & 3D-Unet. For the validation set, 10 random samples from one subject was used. For all the variations of 3D-Denseset sub-volumes of 28^3 were sampled. The training consisted of 2000 **non-empty** samples to have more representative sampling. For validation set, 50 such sub-volumes from the test subject were generated. We normalize all input images to have zero mean and unit standard deviation (**based on non-zero voxels only**). The training epochs for these datasets varied from 120 to 400 based on the models and the dataset.

Implementation Details

We developed all models in the PyTorch [44] framework. Stochastic gradient descend with single batch size with learning rate $1e^{-3}$ and weight decay $1e^{-8}$ was used for all experiments. While further hyperparameter tuning was tested, this setup produces the best results across models. The metric of the average dice coefficient as well as per brain region in range 0-100 were measured in the testing set. All the models were designed in order to meet the choices of the original authors. For the qualitative evaluations, non-overlapping volumes were used for inference. For all datasets, the experiments were conducted in a NVIDIA GeForce GTX-1080 GPU with 12GB of memory and 16GB of RAM. A complete summary of the networks significant properties for someone to consider are summarized in Table 1.

Table 1. An overview of the used models

	#Parameters (in millions)	# Layers	Volume size (ISEG/MrBrains)	Input Channels (MRI modalities)	GPU Training Memory (ISEG/MrBrains) in GB	Total train time in hours
Vnet	~45M	113	128x128x (64/32)	1	4.1/3.9 GB	6h
Vnet light	~25M	84	128x128x (64/32)	1	2.5/2.0 GB	5h
3D-Unet	~10M	95	128x128x (64/32)	1, 2, 3	3.7/1.9 GB	4h
3D-Densenet	**~2.7M**	34	28^3	1, 2, 3	**1.6 GB**	10h
2stream-3D-Densenet (late fusion)	~5.2M	67	28^3	2	2 GB	12h
2stream-3D-Densenet (early fusion)	~2.9M	40	28^3	2	1.7 GB	12h
3stream-3D-Densenet (early fusion)	~3.4M	43	28^3	3	2.5 GB	16h

6.2) Experimental Results – Quantitative evaluations

Validation Results on ISEG 2017, 2 modalities, 4 classes

	Average DSC (%)	CSF (%)	GM (%)	WM (%)
Vnet	87.20	92.53	71.42	84.88
Vnet light	83.79	87.51	69.07	80.04
3D-Unet	**93.84**	**94.99**	**90.84**	**89.51**
3D-Densenet	75.51	79.40	74.90	63.92
2stream-3D-Densenet (late fusion)	76.65	79.51	72.43	63.01
2stream-3D-Densenet (early fusion)	78.63	86.23	75.72	70.71

Validation Results on MR-Brains 2018, 3 modalities, 4 classes

	Average DSC (%)	CSF (%)	GM (%)	WM (%)
Vnet	83.04	79.88	83.04	77.34
Vnet light	82.63	77.80	84.85	76.15
3D-Unet	**86.64**	**80.57**	**85.01**	**82.83**
3D-Densenet	65.11	59.96	49.93	67.02
2stream-3D-Densenet (late fusion)	68.21	54.00	60.42	71.75
2stream-3D-Densenet (early fusion)	70.02	64.07	61.01	71.97
3 stream 3D Densenet – early fusion	76.89	72.06	69.43	80.08

Model summary

DSC in %	I-Seg 2017 4 classes	MrBrains 2018 4 classes
Vnet	87.20	83.04
Vnet light	83.79	82.63
3D-Unet	**93.84**	**86.64**
3D-Densenet	75.51	65.11
2 stream-3D-Densenet (late fusion)	76.65	68.21
2 stream-3D-Densenet (early fusion)	78.63	70.02
3 stream 3D-Densenet (early fusion)	-	76.89

It is important to note the DenseNet variations, as implemented from the original publication [5] are evaluated in smaller sub-volumes of 28^3 and in a bigger sample from the original image – 50 instead of 10. For a fair comparison of the predicted results, a visual comparison is necessary. Finally, authors [5] do not provide with all the implementation details to reproduce their results. As we observed from the train and validation curves on these datasets a more intense regularization strategy is necessary (i.e. higher dropout probability).

Our results validate that in such biomedical applications, only very few images are adequate to train a network that generalizes reasonably well in a few classes. **This is because each image already comprises repetitive structures with corresponding variation**. The problem of volumetric segmentation works adequately well in normal brain MRI. However, in order to train deep learning models to find mappings in more **heterogeneous** data such as **brain tumors** of other complex pathologies with high variability, more data will be necessary.

6.3) Experimental Results – Qualitative evaluations

In the section, we present train-validation curves for the best models. Our results clearly illustrate that the models produce generalized representations. In all plots we observe the validation set to vary.

Validation Results on ISEG 2017 – UNET3D

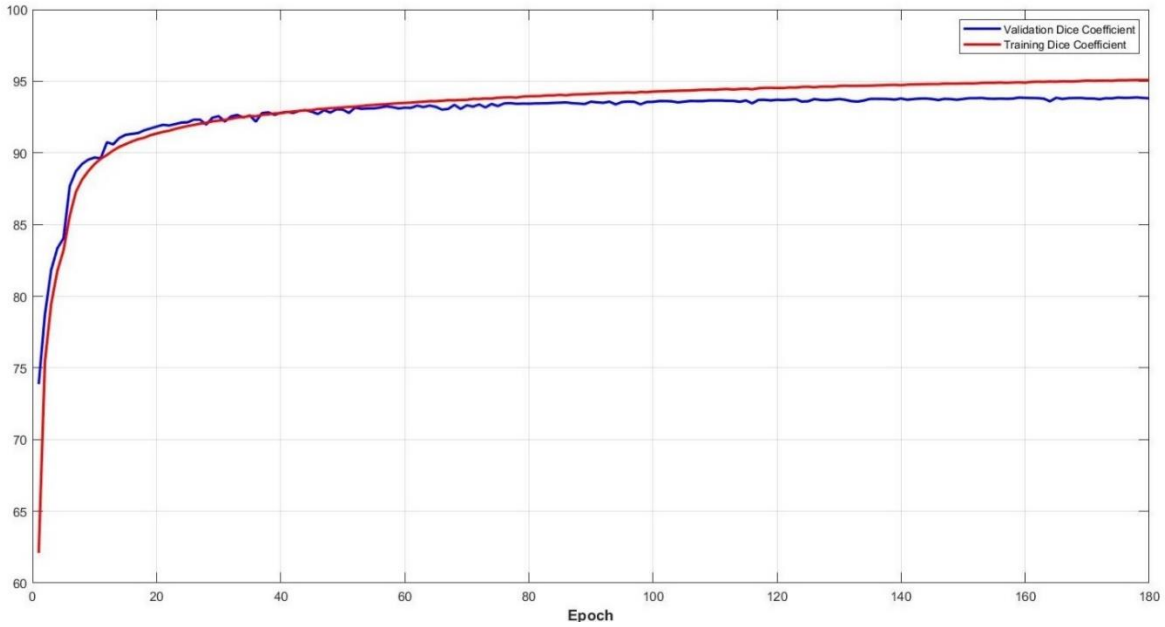


Figure 62, Learning curve DSC coefficient – UNET3D

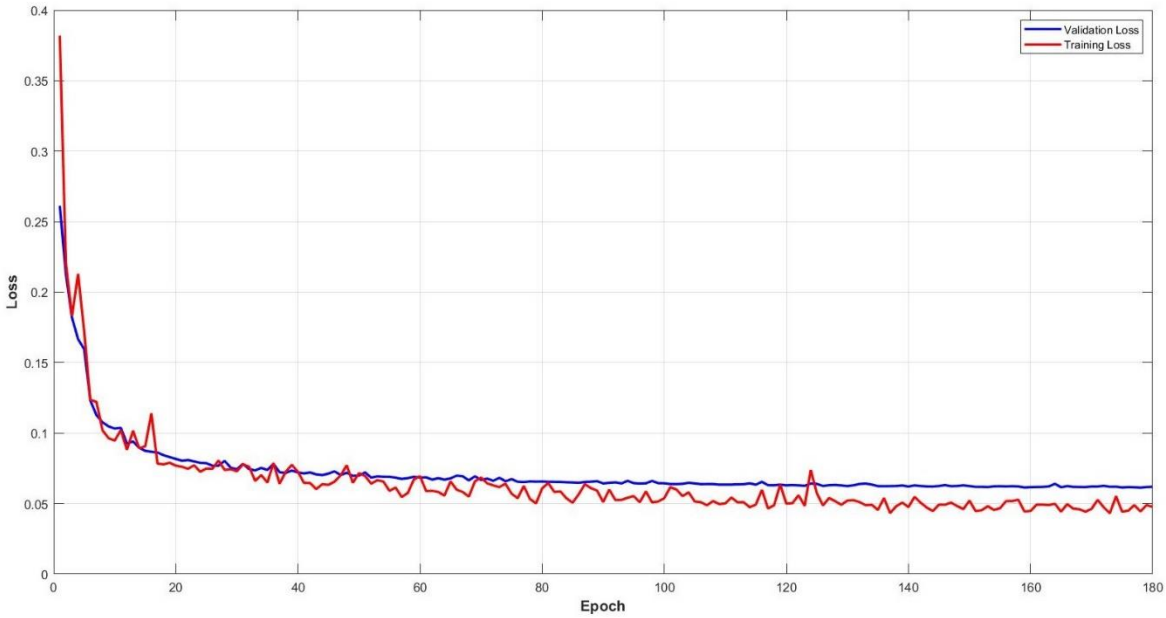


Figure 63, Train & Val loss – UNET3D

Validation Results on ISEG 2017 – VNET

We observe that VNET Dice score is not so stable even though it has more trainable parameters. However, VNET is designed to have only 1 input channel, so only T1 weighted MRI was used.

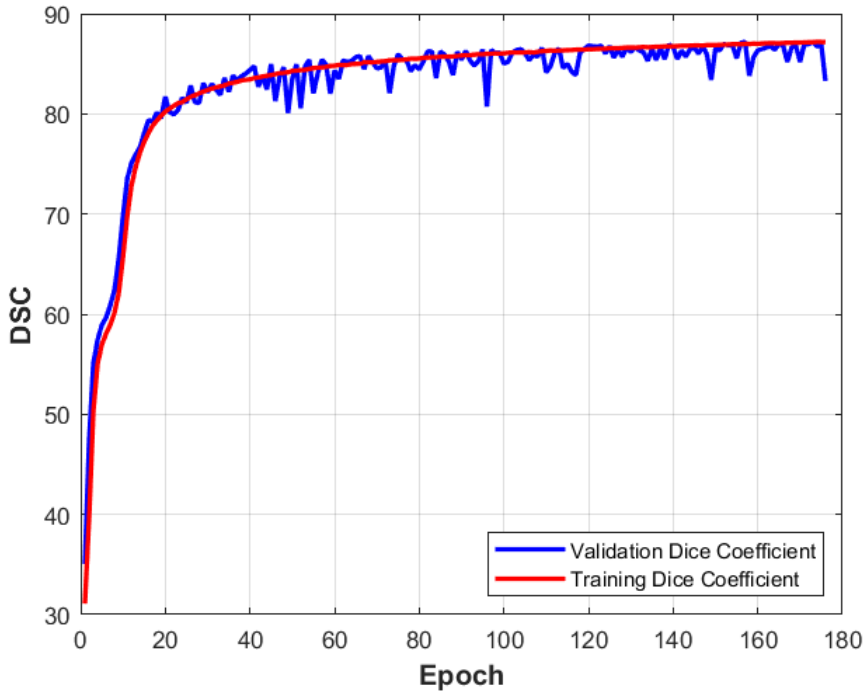


Figure 64, Learning curve, dice coefficient – VNET

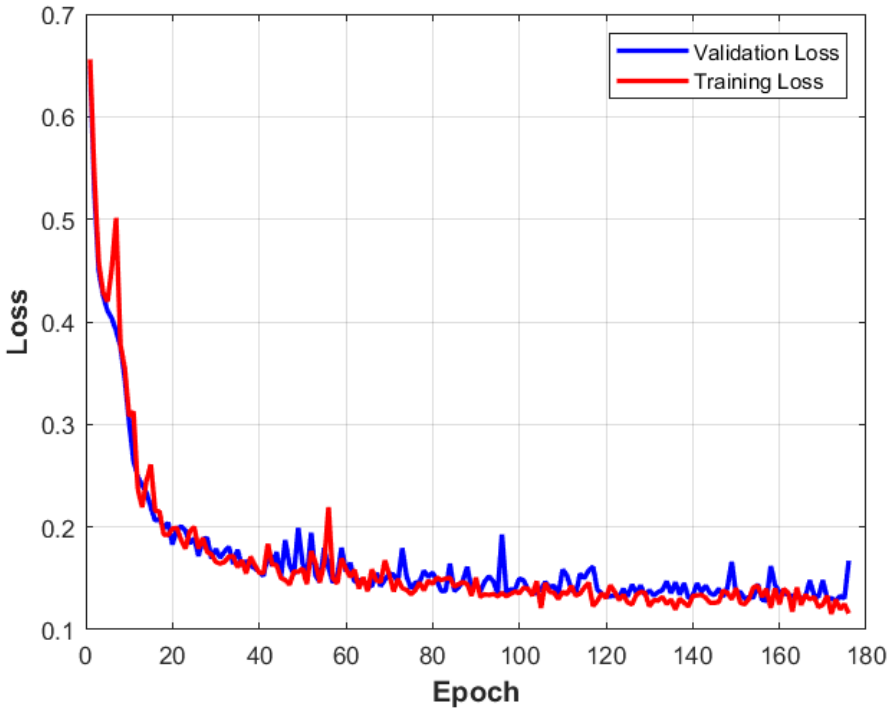


Figure 65, Train & Val loss – VNET ISEG

Validation Results on MrBrains 2018 – UNET3D

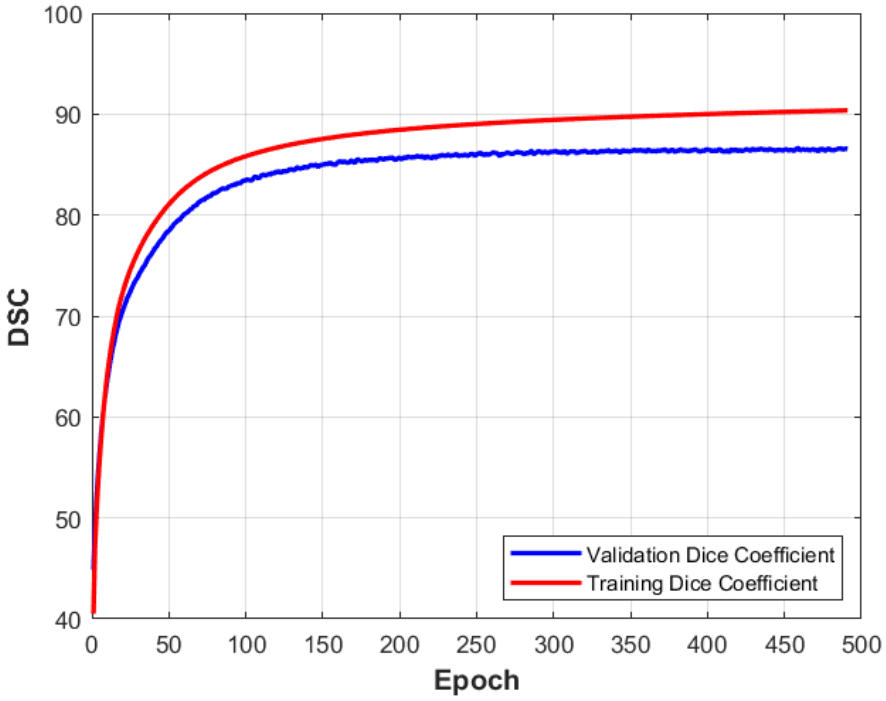


Figure 66, Learning curve, dice coefficient – UNET3D

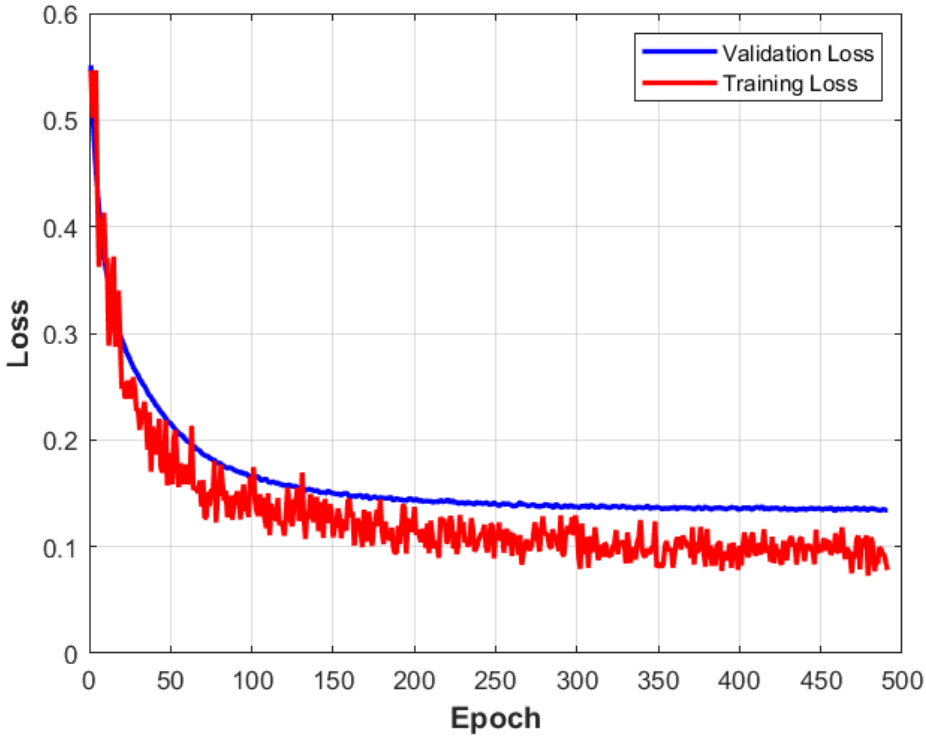


Figure 67, Train & Val loss – UNET3D

Validation Results on MrBrains 2018 – VNET

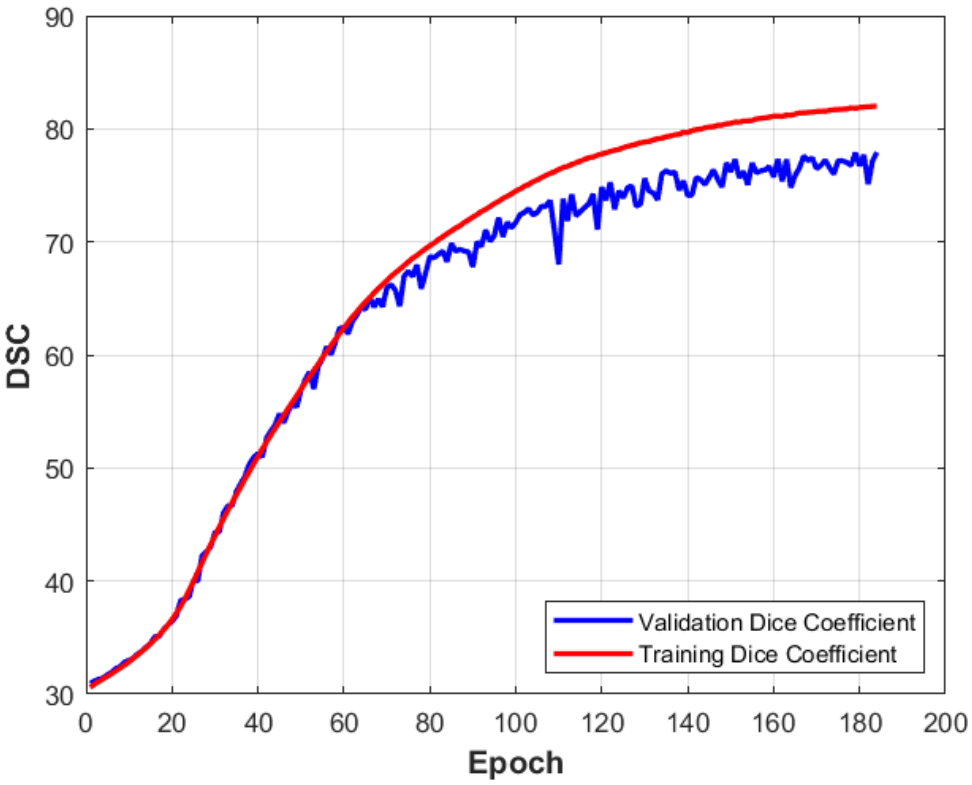


Figure 68, Learning curve, dice coefficient – VNET MRBRAINS

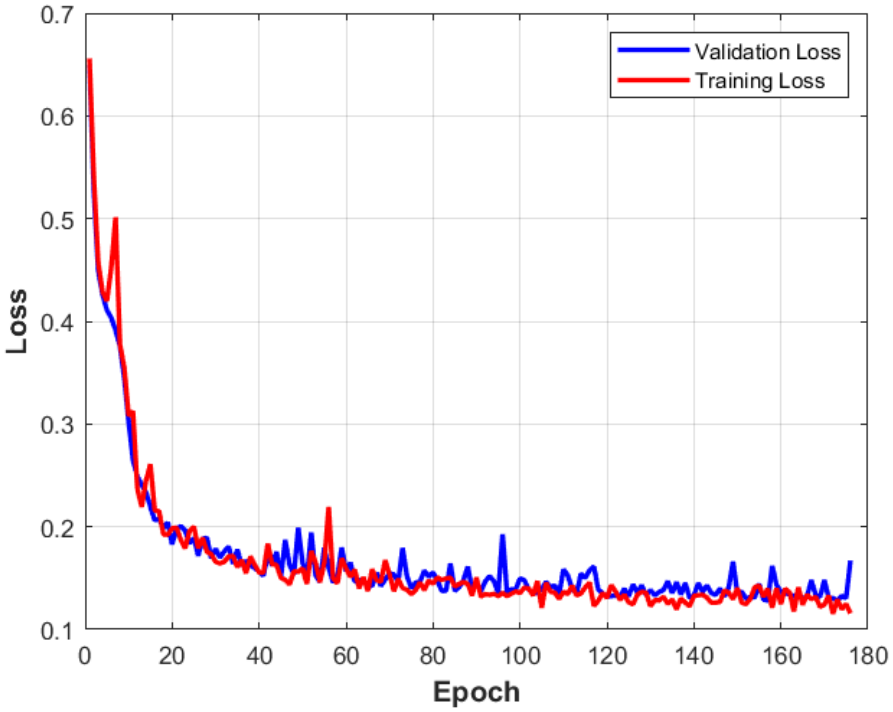
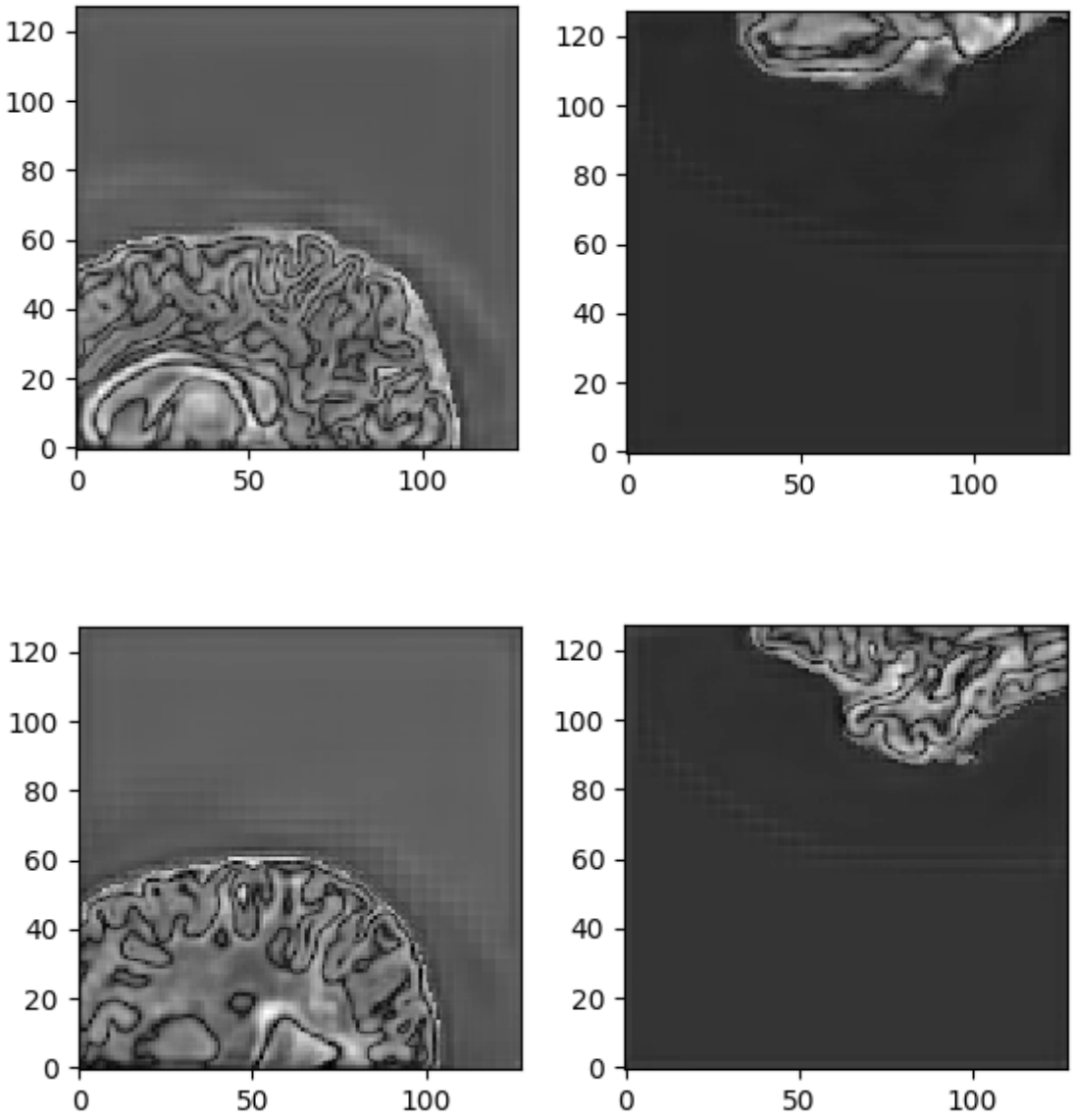


Figure 69, Train & Val loss – VNET MRBRAINS

Visualization of output feature volumes

As a side note and for intuitive purposes, we validate that DNNs learn highly semantic information based on the brain segmentation task in the final layer. We visualize the output volumes by keeping the maximum element without normalization or softmax. We visualize a few neighbour slices as a valid proof. As expected, higher-level structures are learned. In these tasks they correspond to complex brain structures. We observe that the deep network tries to somehow reconstruct the input modalities structures in order predict a good volumetric brain segmentation. We consider this as a critical finding as well as a point of reference for future research.



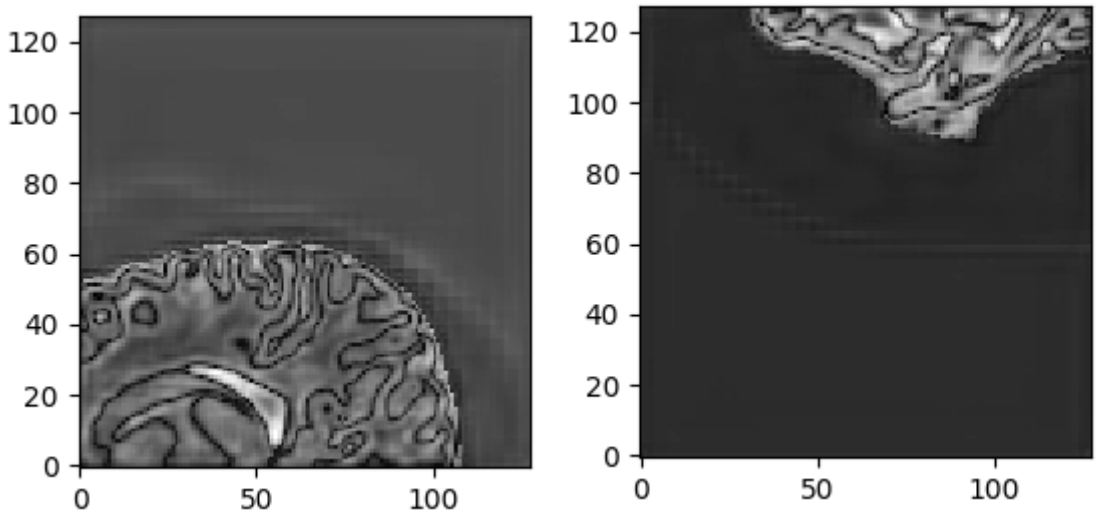


Figure 70, 3D Unet output volumes visualization

Visualizing sub-volume predictions

We visualize the predicted 2D segmentation for a particular slice. As expected from the quantitative evaluations, the network predicts air voxels with excellent accuracy, while the predictions are not so good near the boundaries. However, predictions need to be visually evaluated with their accompanying ground truth data by physicians.

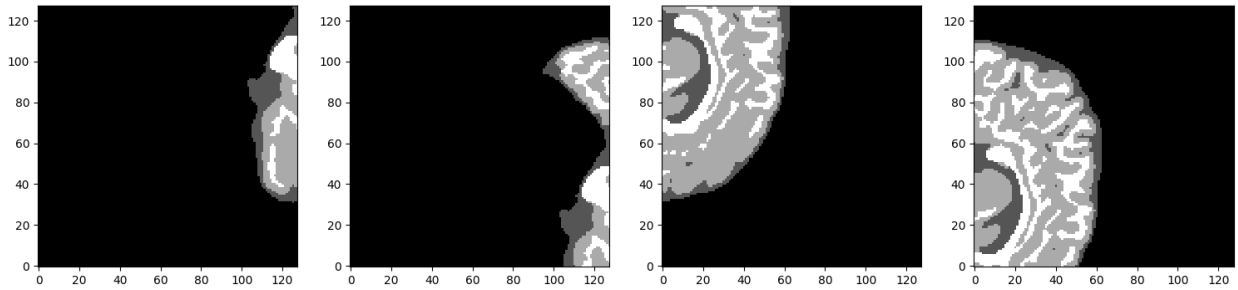


Figure 71, Predicted Segmentation outputs – Validation Set

Comparison with ground truth segmentation

The produced maps are visually close to ground truth data. Although an expert can clearly distinguish the differences, Fig. 72 & 73 serve as a nice illustration of the expressive power of 3D deep networks.

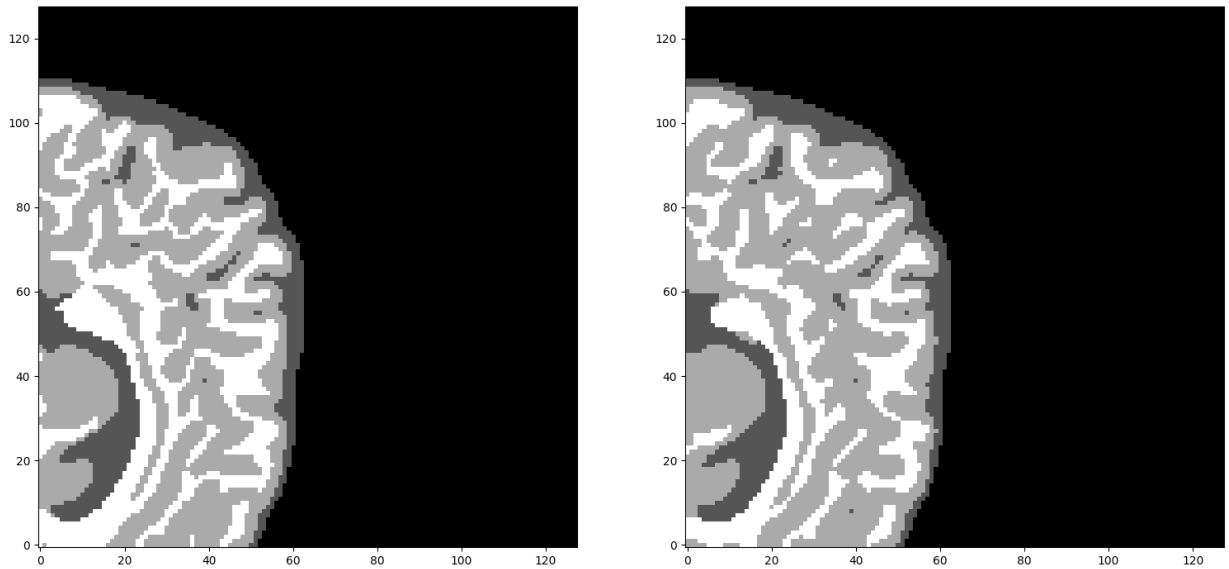


Figure 72, Comparison with G.T. sub- volume segmentation

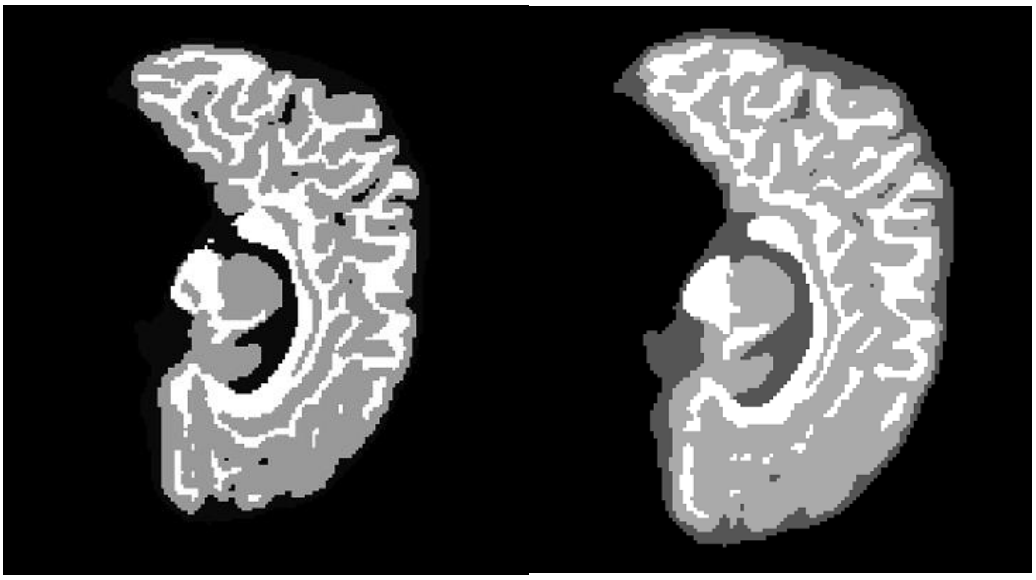


Figure 73, Full Slice segmentation G.T. VS Combined segmentation from 4 overlapping predictions

CHAPTER 7 – Conclusion & Future Work

7.1) Conclusion & Future Work

Medical image segmentation and especially brain segmentation needs to be very accurate and precise than the other non-medical image segmentation applications. We conclude by stating that DL in medical data analysis is here to stay. Even though there are many challenges associated to the introduction of deep learning in clinical settings, DL methods produce valuable results because of the tremendous expressive power of deep networks. The impact of deep learning in medical images will be affected by the breakthroughs in computer vision. Medical image challenges including but not limited to MICCAI challenges will bring more researchers from the community of computer vision in the medical image domain. Further improvements in deep networks will reduce the training and inference time of 3D deep networks.

In addition, we mentioned that the training data has to be representative of the data the network will meet in the future. If the training samples are from a data distribution that is different from the one met in the real world, then the network's generalization performance will be lower than expected. Considering the large difference between the high-quality images one typically works with when doing research and the messiness of the real, clinical world, this can be a major obstacle when putting deep learning systems into production - clinical practice. In this direction, the rise of the sub-field of domain adaptation aims to solve these challenges by enabling cross-dataset evaluations (i.e. different MRI protocols, MRI's from different vendors).

The use of deep learning within medical imaging is still in its infancy. Radiology is expected to be revolutionized by artificial intelligence breakthroughs. Biomedical engineers will play a significant role in this change along with radiologists. We are able to produce good generalization properties in dense maps prediction. However, in more complex tasks such as report generation, current state-of-the-art models perform poorly, compared to humans.

A widely used technique is transfer learning [47], also called fine-tuning or pre-training: first you train a network to perform a task where there is an abundance of data, and then you copy weights from this network to a network designed for the task at hand. For two-dimensional images one will almost always use a network that has been pre-trained on the ImageNet data set. The basic features in the earlier layers of the neural network found from this data set typically retain their usefulness in any other image-related task (or are at least from a better starting point than random initialization of the weights). Starting from weights tuned on a larger training data set can also make the network more robust. Focusing the weight updates during training on later layers requires less data than having to do significant updates throughout the entire network. One can also do inter-organ transfer learning in 3D, an idea we have used for kidney segmentation, where pre-training network to do brain segmentation decreased the number of annotated kidneys needed to achieve good segmentation performance.

The field of 3D DNN's is constantly evolving in a fast-evolving pace. More recent publications include the idea of producing multiple branches from a single image modality such as multiple scale volumes that are fused with various ways in the network. Our goal is to provide an open-

source PyTorch library that implements a series of 3D state of the art DNN's called 3D medical zoo. The first version is released along with the material of this master thesis. Finally, we aim to release pretrained models, similar to RGB images feature extractors that exist in 2D networks.

Medical Zoo PyTorch alpha release: https://github.com/black0017/MedicalZooPytorch

Appendix A - Medical Imaging 1-Page Glossary

Radiography: Radiography is the process of obtaining an image for diagnostic examination using x-rays.

Computed tomography: A kind of imaging procedure that produces cross-sectional images or "slices" of structures in the human body including organs like the brain, heart and lungs, as well as soft tissue in the abdomen and pelvis, which can form a more complete picture than x-ray images.

Ionizing vs. nonionizing radiation: Ionizing radiation has enough energy to remove tightly bound electrons from atoms, thus creating ions. Nonionizing radiation has enough energy to move atoms in a molecule around or cause them to vibrate, but not enough to remove electrons (i.e. sound waves, visible light and microwaves)

Modality: Modality refers to a type of technology (for example, magnetic resonance or computed tomography).

X-rays: beams that pass through the body to produce images of anatomical structures. The beams are absorbed in different amounts depending on the density of the material they pass through. The resulting image analysed by the radiologist is called a radiograph.

Image registration: the process that searches for the correct spatial alignment of medical images.

Image restoration: attempts to recover the original image from the degraded by mathematically modelling the degradation process and applying the inverse process on the degraded image**.**

Image contrast: the ability of the imaging system to sense small variations in radiation intensity at its detector end and to display that intensity variation.

Image enhancement: suitably modify the image so that the resulting image will be of superior quality to the eye of the observing physician.

Image Smoothing: concerns image processing techniques used for suppressing noise, which for various reasons reside in the image.

Super-resolution: a class of techniques that enhance the resolution of an imaging system

Medical image reconstruction: deals with the generation of slice images. These images are created from lower dimensional input data. In MR the signal is sampled in the k-space of the image and reconstructed using an inverse Fourier transform.

Appendix B - MRI 1-Page Glossary

B_0: The static external magnetic field of an MR scanner. The field strength in clinical MR imaging ranges from 0.064–3.0 tesla (up to 8T in experimental applications).

Coil: Component of an MR scanner that serves to transmit RF pulses and/or receive MR, signals.

Contrast-to-noise ratio (CNR): Measure of the ability to differentiate two adjacent anatomic structures in an MR image based on their signal intensities in relation to image noise.

Echo time (TE): The interval between excitation of a spin system and collection of the MR signal. TE predominantly determines the amount of T2 contrast of the resultant image.

Field of view (FOV): The area of anatomy covered in an image. The FOV is usually square, though a Rectangular FOV may be chosen to reduce scan time. A smaller FOV improves spatial resolution but decreases SNR

Frequency encoding: While the echo is being sampled, a gradient field is switched on in one dimension, imparting different precessional frequencies to the nuclear spins along that dimension. In this way, a spectrum of resonance frequencies is obtained instead of a single frequency. The frequency information serves to locate the individual signal components in space along the gradient.

Frequency-encoding gradient: The gradient field that is switched on while the MR signal is being collected, hence it is also called readout gradient.

Gradient: Defines the strength of the change of a quantity in a specific spatial direction. A magnetic field gradient in MR imaging refers to the linear change in magnetic field strength created along the x-, y-, or z-axis of the stationary magnetic field. Such gradients are needed for slice selection. In a more general sense, the term “gradients” is also used to denote the gradient coils.

Inter-slice gap: The distance between the nearest edges of two adjacent slices.

Inversion time (TI): The interval between the 180° inversion pulse and the 90° excitation pulse in an inversion recovery sequence. The TI can be selected to null the signal from a specific tissue such as fat, which is done by applying the 90° RF pulse when the magnetization of that tissue is zero.

K-space: The mathematical space for storage of the measured raw data before the MR image is reconstructed by applying 2D or 3D Fourier transform. The centre lines of k-space predominantly determine image contrast while the peripheral lines mainly affect spatial resolution.

Larmor frequency: Frequency at which spins precess about a magnetic field. The precession or resonance frequency is proportional to the strength of the magnetic field applied.

Phase: The angle by which a rotating magnetic vector of a precessing spin in the xy-plane differs from that of a second vector.

Phase encoding (part of spatial encoding): Accomplished by switching a gradient to impart different phase shifts to the spins in an excited slice, according to their position along the gradient. Spatial position can then be identified by a unique amount of phase shift.

Phase-encoding gradient: The gradient that is switched on for Phase encoding during readout of the MR signal.

Receiver bandwidth: The spectrum of spin frequencies registered in MR imaging during readout.

Repetition time (TR): The interval between two successive excitations of the same slice. By changing the TR, the user can determine the amount of T1 contrast of the resultant image.

Resonance frequency: Frequency at which resonance occurs, which corresponds to the Larmor frequency of protons.

Signal-to-noise ratio (SNR): Measure of image quality expressed as the relationship between signal intensity and image noise.

Voxel: Volume element that is represented by a → Pixel in the two-dimensional MR image; voxel size determines → Signal-to-noise ratio and spatial resolution.

Appendix C - Where to find medical imaging data

Datasets for brain MRI segmentation that were used in this thesis:

1. http://iseg2017.web.unc.edu/
2. https://mrbrains18.isi.uu.nl/

Other MRI datasets:

1. https://www.smir.ch/
2. http://fcon_1000.projects.nitrc.org/indi/abide/
3. http://www.humanconnectomeproject.org/
4. https://openneuro.org/
5. http://adni.loni.usc.edu/
6. https://grand-challenge.org/
7. https://www.oasis-brains.org/
8. https://brain-development.org/ixi-dataset/
9. https://www.med.upenn.edu/sbia/brats2018/data.html

References

[1] Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.

[2] Milletari, F., Navab, N., & Ahmadi, S. A. (2016, October). V-net: Fully convolutional neural networks for volumetric medical image segmentation. In 2016 Fourth International Conference on 3D Vision (3DV) (pp. 565-571). IEEE.

[3] Suzuki, K. (2017). Overview of deep learning in medical imaging. Radiological physics and technology, 10(3), 257-273.

[4] Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., ... & Sánchez, C. I. (2017). A survey on deep learning in medical image analysis. Medical image analysis, 42, 60-88.

[5] Dolz, J., Gopinath, K., Yuan, J., Lombaert, H., Desrosiers, C., & Ayed, I. B. (2018). HyperDense-Net: A hyper-densely connected CNN for multi-modal image segmentation. IEEE transactions on medical imaging.

[6] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[7] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708).

[8] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).

[9] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1), 1929-1958.

[10] Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.

[11] Weishaupt, D., Köchli, V. D., & Marincek, B. (2008). How does MRI work?: an introduction to the physics and function of magnetic resonance imaging. Springer Science & Business Media.

[12] Faro, S. H., & Mohamed, F. B. (Eds.). (2006). Functional MRI: basic principles and clinical applications. Springer Science & Business Media.

[13] Brown, M. A., & Semelka, R. C. (2011). MRI: basic principles and applications. John Wiley & Sons.

[14] Lakhani, P., Gray, D. L., Pett, C. R., Nagy, P., & Shih, G. (2018). Hello world deep learning in medical imaging. Journal of digital imaging, 31(3), 283-289.

[15] Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3431-3440).

[16] Drozdzal, M., Vorontsov, E., Chartrand, G., Kadoury, S., & Pal, C. (2016). The importance of skip connections in biomedical image segmentation. In Deep Learning and Data Labeling for Medical Applications (pp. 179-187). Springer, Cham.

[17] Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence, 40(4), 834-848.

[18] Schlemper, J., Caballero, J., Hajnal, J. V., Price, A., & Rueckert, D. (2017, June). A deep cascade of convolutional neural networks for MR image reconstruction. In International Conference on Information Processing in Medical Imaging (pp. 647-658). Springer, Cham.

[19] Schlemper, J., Caballero, J., Hajnal, J. V., Price, A. N., & Rueckert, D. (2018). A deep cascade of convolutional neural networks for dynamic MR image reconstruction. IEEE transactions on Medical Imaging, 37(2), 491-503.
[20] Bermudez, C., Plassard, A. J., Davis, L. T., Newton, A. T., Resnick, S. M., & Landman, B. A. (2018, March). Learning implicit brain MRI manifolds with deep learning. In Medical Imaging 2018: Image Processing (Vol. 10574, p. 105741L). International Society for Optics and Photonics.
[21] Chaudhari, A. S., Fang, Z., Kogan, F., Wood, J., Stevens, K. J., Gibbons, E. K., ... & Hargreaves, B. A. (2018). Super-resolution musculoskeletal MRI using deep learning. Magnetic resonance in medicine, 80(5), 2139-2154.
[22] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).
[23] Haskins, G., Kruecker, J., Kruger, U., Xu, S., Pinto, P. A., Wood, B. J., & Yan, P. (2018). Learning deep similarity metric for 3D MR-TRUS registration. arXiv preprint arXiv:1806.04548.
[24] Cao, X., Yang, J., Zhang, J., Wang, Q., Yap, P. T., & Shen, D. (2018). Deformable image registration using a cue-aware deep regression network. IEEE Transactions on Biomedical Engineering, 65(9), 1900-1911.
[25] LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., & Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. In Advances in neural information processing systems (pp. 396-404).
[26] Vogl, T. P., Mangis, J. K., Rigler, A. K., Zink, W. T., & Alkon, D. L. (1988). Accelerating the convergence of the back-propagation method. Biological cybernetics, 59(4-5), 257-263.
[27] Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 5(2), 157-166.
[28] Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In Proceedings of COMPSTAT'2010 (pp. 177-186). Physica-Verlag HD.
[29] Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. Neural networks, 2(5), 359-366.
[30] Wang, L., Nie, D., Li, G., Puybareau, É., Dolz, J., Zhang, Q., ... & Thung, K. H. (2019). Benchmark on automatic 6-month-old infant brain segmentation algorithms: the iSeg-2017 challenge. IEEE transactions on medical imaging.
[31] Schlemper, J., Caballero, J., Hajnal, J. V., Price, A. N., & Rueckert, D. (2017). A deep cascade of convolutional neural networks for dynamic MR image reconstruction. IEEE transactions on Medical Imaging, 37(2), 491-503.
[32] Bermudez, C., Plassard, A. J., Davis, L. T., Newton, A. T., Resnick, S. M., & Landman, B. A. (2018, March). Learning implicit brain MRI manifolds with deep learning. In Medical Imaging 2018: Image Processing (Vol. 10574, p. 105741L). International Society for Optics and Photonics.
[33] Liu, C., Wu, X., Yu, X., Tang, Y., Zhang, J., & Zhou, J. (2018). Fusing multi-scale information in convolution network for MR image super-resolution reconstruction. Biomedical engineering online, 17(1), 114.
[34] Lin, G. C., Wang, W. J., Kang, C. C., & Wang, C. M. (2012). Multispectral MR images segmentation based on fuzzy knowledge and modified seeded region growing. Magnetic resonance imaging, 30(2), 230-246.
[35] Fischl, B., Salat, D. H., Busa, E., Albert, M., Dieterich, M., Haselgrove, C., ... & Montillo, A. (2002). Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain. Neuron, 33(3), 341-355.

[36] Yousefi, S., Azmi, R., & Zahedi, M. (2012). Brain tissue segmentation in MR images based on a hybrid of MRF and social algorithms. Medical image analysis, 16(4), 840-848.
[37] Amin, S. E., & Megeed, M. A. (2012, May). Brain tumor diagnosis systems based on artificial neural networks and segmentation using MRI. In 2012 8th International Conference on Informatics and Systems (INFOS) (pp. MM-119). IEEE.
[38] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2818-2826).
[39] Wang, L., Nie, D., Li, G., Puybareau, É., Dolz, J., Zhang, Q., ... & Thung, K. H. (2019). Benchmark on automatic 6-month-old infant brain segmentation algorithms: the iSeg-2017 challenge. IEEE transactions on medical imaging.
[40] Menze, B. H., Jakab, A., Bauer, S., Kalpathy-Cramer, J., Farahani, K., Kirby, J., ... & Lanczi, L. (2014). The multimodal brain tumor image segmentation benchmark (BRATS). IEEE transactions on medical imaging, 34(10), 1993-2024.
[41] Bakas, S., Akbari, H., Sotiras, A., Bilello, M., Rozycki, M., Kirby, J. S., ... & Davatzikos, C. (2017). Advancing the cancer genome atlas glioma MRI collections with expert segmentation labels and radiomic features. Scientific data, 4, 170117.
[42] Mendrik, A. M., Vincken, K. L., Kuijf, H. J., Breeuwer, M., Bouvy, W. H., De Bresser, J., ... & Jog, A. (2015). MRBrainS challenge: online evaluation framework for brain image segmentation in 3T MRI scans. Computational intelligence and neuroscience, 2015, 1.
[43] Juntu, J., Sijbers, J., Van Dyck, D., & Gielen, J. (2005). Bias field correction for MRI images. In Computer Recognition Systems (pp. 543-551). Springer, Berlin, Heidelberg.
[44] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., ... & Lerer, A. (2017). Automatic differentiation in pytorch.
[45] Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T., & Ronneberger, O. (2016, October). 3D U-Net: learning dense volumetric segmentation from sparse annotation. In International conference on medical image computing and computer-assisted intervention (pp. 424-432). Springer, Cham.
[46] Tustison, N. J., Avants, B. B., Cook, P. A., Zheng, Y., Egan, A., Yushkevich, P. A., & Gee, J. C. (2010). N4ITK: improved N3 bias correction. IEEE transactions on medical imaging, 29(6), 1310.
[47] Ataloglou, D., Dimou, A., Zarpalas, D., & Daras, P. (2019). Fast and Precise Hippocampus Segmentation Through Deep Convolutional Neural Network Ensembles and Transfer Learning. Neuroinformatics, 1-20.

Figure List

A comparative analysis of multi-modal brain-MRI segmentation with 3D deep neural networks

Master Thesis

Deep Learning in Medical Image Analysis

MSc in Biomedical Engineering 2017-2019

Supervisor: Evangelos Dermatas | Student: Adaloglou M. Nikolaos | 1004130