Facultad de Ciencias

Application of Super Resolution Convolutional Neural Networks for correcting Magnetic Resonance Imaging (MRI) artifacts

(Aplicación de redes neuronales convolucionales de súper resolución para corregir artefactos en imágenes de resonancia magnética)

Trabajo de Fin de Grado
para acceder al

Grado en Física

Autora: María Peña Fernández

Director: David Rodríguez González

Fecha: Junio 2022

Agradecimientos

En primer lugar, agradecer a David su disposición, y a Lara su ayuda siempre que lo he necesitado. Gracias también a mis padres y a mi hermano, por apoyarme durante estos años y comprender los noes que recibían por respuesta a muchos de los planes que sugerían. A Abram, por estar siempre ahí, en las buenas y en las no tan buenas. A Ana y a Constanza, por ayudarme a entender lo que pasaba por mi cabeza cuando ni yo misma lo hacía. En general, gracias a todo aquel que, de una manera u otra, ha colaborado para que hoy yo esté aquí.

Abstract

Artifacts are false structures or contrast disturbances seen in magnetic resonance (MR) images. They can be a source of wrong diagnosis, so it is important to detect or correct them if possible. We can classify them into three groups according to the origin of the problem: the molecular physics of the scanned tissues, the behavior of the patients, and the MR scanner or the sequences/techniques used to create the images.

As with many other imaging-related applications, the development of Deep Learning offers new opportunities in the area of medical imaging. Here we examine the potential of super-resolution convolutional neural networks for quality assurance (QA) enhancement of magnetic resonance images. In particular, for the correction, or attenuation of artifacts, and also noise and blur image defects in brain MR images. To our knowledge, this approach has not been performed before.

In addition to these computational aspects, we give a physical description of the MR phenomenon and the image formation process. This will give us the basis for understanding the occurrence of artifacts and the different features related to magnetic resonance imaging used for medical diagnosis.

Key words: Magnetic resonance, super resolution convolutional neural networks, artifacts, Deep Learning, magnetic resonance imaging

Resumen

Los artefactos son falsas estructuras o alteraciones del contraste que se observan en las imágenes de resonancia magnética (RM). Pueden ser una fuente de diagnósticos erróneos, por lo que es importante detectarlos o corregirlos si es posible. Podemos clasificarlos en tres grupos según el origen del problema: la física molecular de los tejidos escaneados, el comportamiento de los pacientes y el escáner de RM o las secuencias/técnicas utilizadas para crear las imágenes.

Al igual que con muchas otras aplicaciones relacionadas con la imagen, el desarrollo del Deep Learning ofrece nuevas oportunidades en el área de la imagen médica. En este caso examinamos el potencial de redes neuronales convolucionales de super resolución para la mejora de la calidad de la imagen de RM. En particular, para la corrección, o atenuación de artefactos y de los defectos de imagen de ruido y emborronamiento de imágenes de RM cerebrales. Hasta lo que nosotros sabemos, este enfoque no se ha realizado con anterioridad.

Además de estos aspectos computacionales, se realiza una descripción física del fenómeno de la resonancia magnética y del proceso de formación de imágenes. Esto nos dará las bases para entender la aparición de los artefactos y las diferentes características relacionadas con las imágenes de resonancia magnética utilizadas para el diagnóstico médico.

Palabras clave: Resonancia magnética, redes neuronales de super resolución, artefactos, Deep Learning, imagen por resonancia magnética

Index

Chapter 1

Introduction

Magnetic resonance imaging (MRI) is a non-invasive technique, unlike computed tomography, for example, in which images are taken by using X-rays, that provides high-resolution images based on the phenomenon of magnetic resonance (MR). It is mainly used in medicine to detect tissue anomalies and diseases like cancer or to study the structure of organs, such as the heart. However, it is also utilized in other fields like industry, as it provides information on the structure of the materials under study (see [1]). In this dissertation, we will focus only on the medical applications of MRI (particularly in brain MR images), even though the underlying physics is similar in all cases.

Nevertheless, this technique can be affected by different factors that worsen the quality of the magnetic resonance images used for medical diagnosis. These include patients' movements, image processing errors, and equipment defects or limitations. They give rise to the so-called artifacts and also to noise and blur in the images. The first ones are false structures or contrast alterations seen in MR images. As they can be a source of wrong diagnoses, it is essential to detect or correct them if possible.

1.1. State of the art

Until recently, when artifacts and image defects appeared, the solution usually was to repeat the resonance, adjusting for the different factors that could have affected the MR image. Studies were also focused on improving images quality by developing medical equipment with powerful detectors capable of obtaining images with better resolution. All of this was costly and time-consuming.

As with many other image-related applications, the development of Deep Learning offers new opportunities in the medical imaging area. It has allowed to perform post-processing once the image has been taken, reducing the inconveniences that failures could cause to patients and physicians. In particular, in the case of image resolution enhancement, the importance of Deep Learning dates back to 2014, when Dong et al. proposed the so-called super resolution convolutional neural network [2] and to 2017, when this network was first applied to medical images (specifically to chest radiographs) [3]. Due to its promising results and simplicity, this method is still being studied, and new variations are being added to improve its performance (see [4], [5], [6]).

1.2. Motivation and objectives

Seeing the success of this simple network in improving image resolution, we wondered if it would also be possible to reduce or eliminate the artifacts (or defects such as noise or blurring) that MR images may have. This approach, in particular, has not been done before, as far as we know, so this work could be the beginning of a new line of research.

The first aspect we will study is the basic principles of magnetic resonance imaging. This will allow us to have a vision of the physical phenomenon that gives rise to the images we are going to work with. It will also allow us to expand the knowledge acquired during the Physics degree on a technique commonly used in medicine. We will also introduce the most significant aspects of Deep Learning, which are new to us and will enable us to learn from a theoretical point of view the basis of the program we will later develop.

Moreover, we will use Python, which is one of the most powerful programming languages nowadays, to develop the code. Among the libraries it has, we will work with PyTorch, of interest for computer vision, and with TorchIO, which will allow us to generate artificially the artifacts that we will aim to correct. The results it provides will be analyzed to study the performance of the super-resolution network applied to MR image artifacts and defects. The explanation of the developed program appears in Appendix B.

1.3. Structure of the work

The work is structured in five chapters, the first of which is the current introduction. The contents are distributed in the rest of them as follows:

- **Chapter 2** explores both the phenomenon of magnetic resonance and the posterior formation of MR images. It is also included an extensive description of image quality, in which the different types of artifacts that can occur are described.
- **Chapter 3** studies Deep Learning from a theoretical point of view, focusing on those aspects of interest for our problem. It also describes the neural network we will use in the next chapter, as well as the Python library applied to generate the artifacts.
- **Chapter 4** explains the dataset used for the simulations and shows examples of the generated artifacts. Then, the results of a model that we will take as baseline are analyzed, and its performance is compared to the one of modified models.
- **Chapter 5** reviews the results obtained in the previous chapters, proposing new lines of research to continue with the study.

Chapter 2

Basic principles of Magnetic Resonance Imaging

Different particles can produce the phenomenon of magnetic resonance, like electrons or protons (see [7]). We will study the second case known as nuclear magnetic resonance (along the bachelor thesis, we will refer to it as MR). This technique is relatively recent in the field of radiology. In 1938, Isidor Isaac Rabi made the first discovery related to magnetic resonance when he demonstrated that molecules within a magnetic field could emit waves at concrete frequencies. Felix Bloch and Edward Mills Purcell extended that discovery to liquids and solids in 1946. Nevertheless, it was not until the 70s that the first MR images were created. From there onwards, the interest in MRI has been growing, as it brought advances in diagnosis and medical research. Thanks to the development of the technology used to create these images, different MRI techniques have been explored and exploited in the last decades, making them more specific for the body part under study [8].

In this chapter, we will explain in detail the phenomenon of MR, as well as some aspects related to MR signal digitalization. Furthermore, different factors that affect image quality and some techniques to improve them will be analyzed. Finally, we will describe the MR system used to create them. For this purpose, we will rely on the books [9], [10], [11], [12] and [13], which will be properly referenced in the corresponding sections.

2.1. Magnetic resonance phenomenon

Atoms consist of a nucleus that contains protons, with positive charge, and neutrons, with no charge, surrounded by electrons negatively charged located in orbitals. All of these sub-atomic particles have an spin angular momentum, commonly called *spin*[1], or *intrinsic spin angular momentum*, which is an intrinsic quantum property. As the atomic nucleus is composed of these particles, it inherits this property.

Magnetic resonance is based upon the behavior of atomic nuclei that possess spin when a magnetic field is applied. In addition, as nuclei are positively charged and have spin, they also have a magnetic moment, whose orientation is parallel to the axis of rotation. The most commonly used atom for medical MR techniques is protium, ^{1}H, whose nucleus consists of a proton and has a spin of 1/2. It is because it responds largely to an applied magnetic field and also because

[1] It is analogous to the classical spin, but we can not consider the particles as if they were rotating.

hydrogen is present in body tissues composed of water and fat, which are the most common ones in a body [9]. Therefore, we will refer to the ^{1}H nucleus throughout this chapter.

2.1.1. Production of net magnetization and application of a RF pulse

If we have a sample with hydrogen atoms (a body tissue, for example), their protons have spin vectors of the same magnitude but different orientations. The net sum of all vectors results null due to this randomness, i.e., the net magnetization is zero. However, once this sample is subjected to a magnetic field $\mathbf{B}_0$ [2] [3], orientations change. Here, protons' spins start to rotate around an axis parallel to $\mathbf{B}_0$, so that the z component of spin is time-constant, but both x and y components vary with time (see Figure 2.1). This movement is called *precession*, and its speed (the rate at which spins within $\mathbf{B}_0$ wobble) is given by the Larmor equation,

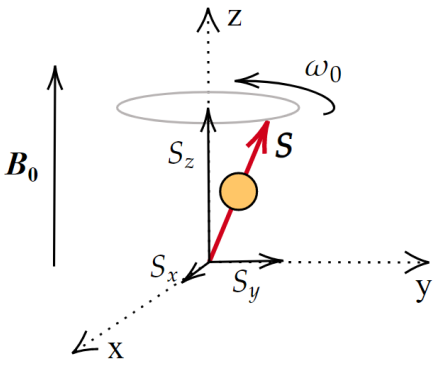


Figure 2.1: Precession movement of a proton (in orange) within a magnetic field $\mathbf{B}_0$, that has a frequency ω_0. **S** is the spin vector or magnetic moment, and S_x, S_y and S_z its projections onto the Cartesian axes.

$$\omega_0 = \gamma\, B_0, \tag{2.1}$$

where ω_0 is the Larmor frequency (in Hertz, which is the convention established in MRI) and $\gamma = \gamma/2\pi$ is the gyromagnetic ratio in units of MHz T^{-1}. In the case of ^{1}H, its value is $\gamma_{^1\mathrm{H}} = 42.58$ MHz T^{-1}. B_0 is the magnitude of the applied magnetic field [10]. For example, if a 1.5T magnetic field is applied, the Larmor frequency results in $\omega_0 = 63.87$ MHz, which is in the range of radiofrequency (between 1 kHz and 300 GHz).

After this process, if we sum the contribution of all protons in the sample, the x and y components are still random, so the net magnetization in the transverse plane xy is still null. However, S_z is constant with time for each proton, making it and $\mathbf{B}_0$ to be coupled. Therefore, we have parallel and antiparallel-aligned protons, as S_z's direction can be along (spin-up) or against $\mathbf{B}_0$ (spin down). This difference in the positions produces a splitting of the energy into two levels that differ $\Delta E = \hbar\omega_0$, known as Zeeman effect. Moreover, the number of parallel ($N_\uparrow$) and antiparallel ($N_\downarrow$) oriented protons is not equal, as the quotient between them follows a Boltzmann distribution, given by

$$N_\downarrow / N_\uparrow = e^{-\Delta E/k_B T}, \tag{2.2}$$

where $k_B = 1.38 \cdot 10^{-23}$ JK^{-1} is the Boltzmann constant and T the absolute temperature. It is noteworthy to mention that spin-up orientation is the one that has lower energy.

As there is a different number of parallel and antiparallel proton magnetic momenta, the sum of all vectors results in a nonzero quantity. It implies a nonzero net magnetization in the z-direction, $\mathbf{M}_0$. The magnitude of this quantity is proportional to the one of the magnetic field $\mathbf{B}_0$, and they both point in the same direction:

$$\mathbf{M}_0 = \chi \mathbf{B}_0, \tag{2.3}$$

[2] Vector quantities will be written in bold letters.
[3] By convention, we will assume that the magnetic field is applied in the z-direction of the Cartesian system.

where χ is the magnetic susceptibility, i.e., the quantitative measurement of the degree of magnetization of a substance immersed in a magnetic field. It is a fundamental quantity in MR as it provides the necessary signals in the experiments.

In particular, the manipulation of $\mathbf{M}_0$ is essential in MR. First of all, it is needed to apply a pulse of energy. If the central frequency of this pulse is equal to the Larmor one (see Equation (2.1)), then the so-called resonance condition is fulfilled. As the Larmor frequency is within the radiofrequency range, the pulse is a radiofrequency (RF) one. When it is applied, a proton wobbling at ω_0 frequency with spin-up or spin-down orientation suffers an absorption or emission of the radiofrequency energy, respectively, i.e., there is a transition between parallel and antiparallel spin states. As there are more spin-up-oriented protons than spin-down-oriented ones, net absorption will be observed in a sample with hydrogen protons. It makes the net magnetization change its direction from the z-direction, following a movement called *nutation* (see [11]). It combines both the precession in the $\mathbf{B}_0$ direction at Larmor frequency and the reduction of its projection in the z-axis, resulting in a spiral movement.

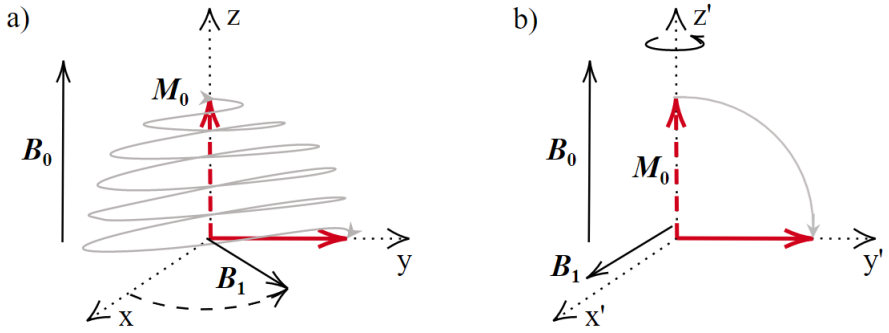


Figure 2.2: Schematic representation of the magnetization vector $\mathbf{M}_0$'s movement (nutation) when a 90°RF pulse is applied. The dashed red arrow is the initial position of $\mathbf{M}_0$ and the continuous red one is the final position. a) represents the situation in the laboratory frame of reference, and b) does it in the rotating frame.

The RF pulse effect is represented by a new additional magnetic field $\mathbf{B}_1$, perpendicular to $\mathbf{B}_0$. Thus, the magnetization vector tends to be aligned with both fields, so it begins to separate from the z-axis nutating perpendicular both to $\mathbf{B}_0$ and $\mathbf{B}_1$ [9](see Figure 2.2). This movement lasts the period of time the pulse is applied, and the angle between the initial and final position of the magnetization vector α is known as the flip angle. This quantity characterizes the pulse, and the most commonly used in MR techniques are 90°and 180°RF pulses. It is because the magnetization over the transverse plane xy produces the MR signal, which will be explained in the next section. The 90°pulse makes the magnetization vector flip to the xy plane and the 180°one in the opposite direction to the initial one.

To easily observe the effect this pulse causes in the net magnetization, it is common to use a rotating coordinate system in which the x and y axes rotate around the z-axis at Larmor frequency.

2.1.2. Relaxation

Once the radiofrequency pulse has finished, those protons that have absorbed or emitted energy return to their equilibrium position. In the sample, this process results in a net emission of energy and the return of the magnetization vector to its initial position. Due to this process, called *relaxation* or *nuclear relaxation*, the MR signal rapidly fades. Two different processes

produce this fading, T1 relaxation and T2 relaxation. In order to explain them, we will assume that we have applied a 90°RF pulse [12].

T1 relaxation, also known as longitudinal relaxation, has to do with the return of the net magnetization vector $\mathbf{M}_0$ to its initial position before the pulse was applied (along the z-axis). The relaxation time T1 is, by definition, the time needed for the projection of $\mathbf{M}_0$ in the z-axis to return to 63% of its initial value and depends on B_0. This net magnetization in the longitudinal direction (along the z-axis) increases its value over time following an exponential growth process. T1 depends on the tissue or sample the hydrogen protons are immersed in. For example, T1 in fat is much shorter than in water. This quantity represents the time needed for the out-of-equilibrium spins to recover.

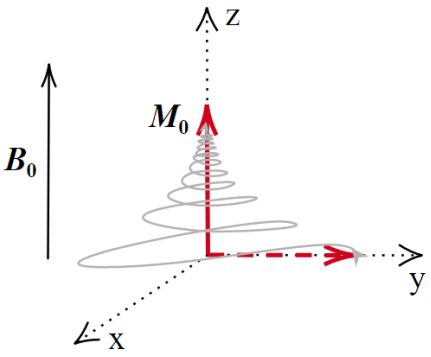


Figure 2.3: Relaxation movement of the net magnetization vector $\mathbf{M}_0$ (in red) after a 90°RF pulse. The dashed red arrow is the initial position of $\mathbf{M}_0$ and the continuous red one is the final position.

As the magnetization along the z-axis increases, its transverse value decreases, causing the MR signal to fade. In this case, the excited protons do not exchange their energy between them, as we will see that it is what happens in the T2 relaxation, but they give it up to the surroundings. Then, we can consider that this process is due to a 'spin-lattice' interaction.

On the other hand, we have transverse relaxation, a process based on the loss of phase coherence in the rotating spins. At first, when the pulse has finished, spins precess synchronously at the same speed in the xy plane. By synchronous, we mean that spins are in phase, i.e., they all point in the same direction. Then, it is considered that they are in phase coherence. However, this coherence is gradually lost for two reasons with time. The first one is the result of the interaction between small magnetic fields spins are associated with, and it causes the dephasing (lose of coherence) that, at the same time, produces the net transverse magnetization to reduce its value. This process is the so-called T2 relaxation, where time T2 is, by definition, the time needed for the transverse component of the net magnetization vector to reduce its value to 37% of its original one. It determines the time the MR signal needs to fade after the excitation. The second reason for this dephasing is the inhomogeneities in the magnetic field. The process of this decay of $\mathbf{M}_0$ is known as T2* relaxation.

The equations of motion of the net magnetization vector both in the nutation and the relaxation processes are detailed in Appendix A. The movement of $\mathbf{M}_0$ during relaxation is represented in Figure 2.3.

2.2. Formation and reconstruction of images

When an MRI technique is applied to a patient, the first step that has to be done is the selection of the plane of space we want to get an image of. Depending on the way the body is divided, there are three different cutting or anatomical planes: axial (divides the body into upper and lower sections), sagittal (divides the body into right and left parts), and coronal (divides the body into anterior and posterior sections). Let us suppose we want an axial MR image. A vari-

able magnetic field in the craniocaudal direction[4] (from top to bottom of the body) is generated to excite just a slice (anatomical volume under examination) parallel to this plane. To do so, once the main magnetic field $\mathbf{B}_0$ is applied, a gradient G_z has to be switched on to change its value along the z-direction. As a result, the magnetic field has different strength and therefore, different ranges of frequencies along the body. Then, an RF pulse is applied, whose central frequency is equal to the Larmor frequency of the selected slice, in order to excite it. Gradients along the x and y axes can be switched on instead of G_z if we want to select a slice in another anatomical plane. They can also be switched on in combination if an oblique slice is desired. The shallower the variable field is, the thicker the slice will be [13].

Once the slice position is selected, more magnetic gradients along the x and y axes have to be applied to identify the spatial position of an MR signal. This process is called *spatial encoding* and is divided into two steps: phase encoding and frequency encoding.

Firstly, we explore the phase encoding, which is the most difficult step of the whole process. As we have already selected and excited a slice (divided into cubic voxels, see Figure 2.4.a.) with an RF pulse, its hydrogen protons are now rotating in phase with the frequency corresponding to the magnetic field. Now, G_z is switched off, and a new magnetic gradient called phase encoding gradient is switched on along one of the sides, for example, along the y-direction, G_y. It makes the nuclei immersed in a stronger field relax at a higher frequency than those inside a weaker field, so the first ones get an advanced phase. Once G_y is switched off, nuclei of different rows within the slice are now out of phase. However, they all still precess at the main magnetic field's frequency. Spatial encoding of a row by phase has been produced.

The next step is to differentiate the signal emitted by each voxel in a row. A new gradient G_x, called frequency encoding gradient, perpendicular to both G_y and G_z is used to do so. The effect produced in the hydrogen atoms (by columns of the slice) is the same as before. In this case, it alters the phase encoding differently for each voxel according to its position. As each column of voxels relaxes at a frequency that depends on its position, spatial encoding of columns by frequencies has been achieved.

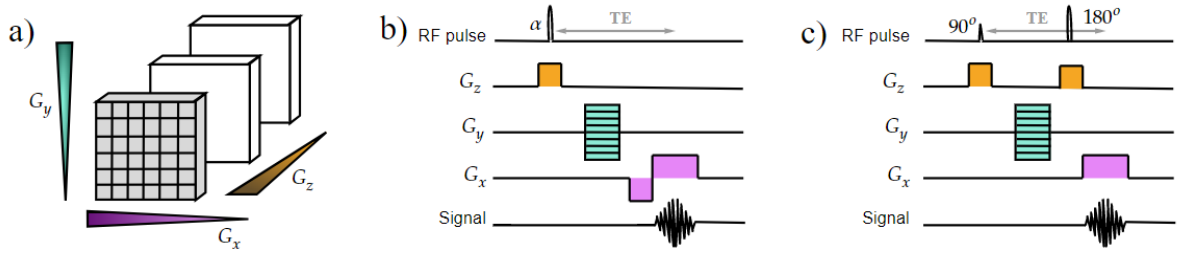


Figure 2.4: a) Schematic representation of the application of the selection (G_z), the phase-encoding (G_y) and the frequency -encoding (G_x) gradients drawn as triangles on a slice composed by voxels. b) Sequence of formation of a gradient echo with an excitation α pulse. G_y represents all the different values the phase-encoding gradient takes, and TE is the echo time, previously defined. c) Sequence of formation of a spin echo with an excitation 90°pulse and a refocusing 180°one.

It is during the application of the frequency encoding gradient that the electrical signal (echo) is collected. There are two types of echo according to the way they are produced. We have gradient echos (GRE) if G_x is a bipolar gradient, which means that it is divided into two gradients of the same magnitude and opposite directions $(-G_x, G_x)$. The first one produces the phase loss, and

[4]From now on, we will designate it as the z-direction.

the other has the inverse effect and makes the nuclei to be in phase again. It is during the application of the second gradient that the growing-amplitude electrical signal (echo) is collected. If $-G_x$ is applied during t_x, the positive one is applied during $2t_x$. Then, we also collect the signal during a new relaxation process, i.e., the decreasing-amplitude part of the electrical signal (see Figure 2.4.b.). On the other hand, we have spin echos (SE) if G_x is unipolar. In this case, the effect of refocusing the spins is produced by a 180°RF pulse (see Figure 2.4.c.). The maximum of the echo is obtained when all spins are again in phase. Echo time (TE) is the elapsed time between the excitation RF pulse is applied, and the maximum of the signal is collected.

It is required as many different G_y as rows the slice has, and the variation between them, ΔG_y, should be constant. Firstly, a gradient G_{y1} is switched on, followed by the frequency encoding gradient G_x, and then an echo is collected. The process is repeated with G_{y2}, G_{y3},...G_{yn} and 0, -G_{y1}, -G_{y2},..., -G_{yn} with the same G_x, where $2n+1$ is the number of rows the slice is divided in. Therefore, a composed signal with different frequencies associated with each voxel relaxation rate will be obtained for each G_y. It is noteworthy that the signal intensity will be higher if G_y is shallow because the loss of coherence of spins will be smaller, and then, the net magnetization will be larger. This consideration is relevant when enhancing the contrast of MR images. On the other hand, strong G_y gradients allow a better spatial resolution, as the difference between the frequency of each row is larger.

Once that signals have been collected, the next step that has to be done to create a final image is to digitalize them, which makes working with them simpler. To do so, these analog signals have to be amplified and demodulated first. By demodulating, we mean to delete the frequency corresponding to the main magnetic field (frequency of the carrier wave) from the range of frequencies signals have. The magnitude of the bandwidth (BW) we can work with now is useful because it indicates the frequencies accepted by the signal receiver to be digitalized. Remarkably, demodulation creates two different components: a real and an imaginary component. Therefore, two filtered signals arise, whose consequences will be covered in the following subsection (see Subsection 2.2.1).

After that, the digitalization process starts. The intensity of the signals in fixed intervals of time t_s, whose value is the inverse of BW, has to be measured. The number of samples taken corresponds to the number of columns the excited slice has. Then, a crucial theorem in the field of information theory has to be used to ensure that the analog signal is rightly reconstructed. It is the so-called Nyquist-Shannon theorem, whose proof is out of the scope of this dissertation. It is related to the sampling frequency $f_s = 1/t_s$, which can be defined as the number of samples of the analog signal taken in a second. Therefore, to recover the original signal and have the proper digital representation, the Nyquist-Shannon theorem's condition must be fulfilled:

Theorem 2.2.1. *(Nyquist-Shannon Theorem) Let f_s be the sampling frequency and f_{max} the maximum frequency collected in the filtered echo. Then, f_s has to be at least twice f_{max}, i.e.,*

$$f_s \geq 2f_{max}.$$

The minimum accepted sampling frequency is the so-called Nyquist frequency f_{Nq}, i.e., $f_{Nq} = 2f_{max}$.

Note that this f_{max} corresponds to the value of half the bandwidth, so $f_{max} = \mathrm{BW}/2$. Thus, $f_{Nq} = \mathrm{BW}$. If f_{Nq} is below that value, the aliasing phenomenon appears, making collected signals indistinguishable and not allowing the analog signal to be reconstructed.

2.2.1. k-space and Fourier transforms

Once we have described how these signals are created and digitalized, we can point out where this information is stored and how it can be transformed into an image. Firstly, we can picture the measured values of an echo (for example, from the real component signal) from a particular gradient field G_y constituting a line. We will have as many lines as phase encoding fields were applied. All these lines can be displayed as rows of a matrix (a two-dimensional grid composed of values displayed in rows and columns), arranged by the strength of G_{yi}. Columns are separated by an interval of time equal to the sampling time t_s, and rows by the elapsed time between two consecutive RF excitation pulses, known as the repetition time (TR), i.e., information is available in the temporal domain. To construct images, we need to express these values in the spatial frequency domain, where each one is represented by a point (k_x, k_y) in the so-called k-space. To express them in spatial frequency units, we have to use the gradient fields G_x and G_{yi}, such that the general expressions of k_x and k_y are:

$$k_x = \gamma G_x t, \qquad k_y = \gamma G_{yi} t_y, \tag{2.4}$$

where $t \in \{0, \pm t_s, \pm 2t_s ..., \pm m t_s\}$, with m being the number of samples taken in each echo until its maximum value (TE). Therefore, we will have -k_x and k_x values around the central point. Moreover, $G_{yi} \in \{0, \pm G_{y1}, ... \pm G_{yn}\}$ are the different phase gradient fields, arranged from the shallower to the stronger field, that vary a quantity ΔG_y. Finally, t_y is the time each G_{yi} is applied. Both horizontal k_x and vertical k_y points are separated in the k-space by

$$\Delta k_x = \gamma G_x t_s, \qquad \Delta k_y = \gamma \Delta G_y t_y. \tag{2.5}$$

Note that k-space has central symmetry, i.e., the data in the upper right quadrant of the space and the one in the down left quadrant are related. The upper left and the downright quadrants are also related. The time required to obtain a magnetic resonance image varies depending on how these k-spaces are filled and the amount of information they stock.

Now, we have two complete matrices (one for each digitalized signal, real and imaginary) with rows and columns separated by Δk_y and Δk_x, respectively, so we can proceed with the creation of the image, which has a spatial domain. Mathematical algorithms known as Fourier transforms (TF) are the operators used to convert the k-space into the image. An MR image is a matrix composed of rows and columns, creating squares called pixels. TF assigns a shade (in grayscale) to each image pixel according to the signal intensity information k-space has. Therefore, we will have two images corresponding to both k-spaces, a magnitude, and a phase representation. Even though the number of rows and columns are equal in the image and its equivalent k-space, it is relevant to point out that each point (k_x, k_y) has information corresponding to all the pixels of the image. In fact, central points of the k-space contain the information that determines image contrast and brightness, while exterior points determine the spatial resolution (edges and details).

2.2.2. MR sequences

The steps of excitation of a slice, phase encoding, generation, and collection of an MR signal make up an MR sequence. These steps have to be repeated many times if high-quality images are desired. In addition, image contrast and quality vary depending on the characteristics of both the gradients and the pulses applied. Therefore, there are different types of sequences according to their components: spin-echo (SE) sequences and gradient-echo (GRE) sequences, already

explained previously, inversion recovery (IR) sequences, turbo spin-echo (TSE) sequences (modifications of the SE sequences by the addition of more rephasing 180°pulses to accelerate the acquisition process) (see [12])...

SE, TSE, and GRE are the most used ones, and choosing one or another sequence depends on the tissue under analysis or the pathology being studied [14]. They are the sequences used to create the MR images we will work on within this dissertation (see Section 4.1).

The number of echos encoded in each TR is called echo train length (ETL). This quantity is higher in TSE sequences than in SE ones because more RF pulses have to be delivered, so TR also increases. In addition, ETL directly determines the reduction of elapsed time between the emission of the first excitation pulse and the creation of the image. Finally, it is noteworthy that in multi-echo sequences, each echo has a different TE value. Then, the parameter used to characterize and control the image contrast is the so-called effective echo time (TEef), which is the TE of the echo with a null phase encoding (the one that samples the central line of the k-space).

2.3. Instrumentation: the MR system

Having the proper instrumentation is needed to produce an MR image. An MRI system is composed of many components with different functions that make the image's creation process possible. The most important ones will be explored throughout this section.

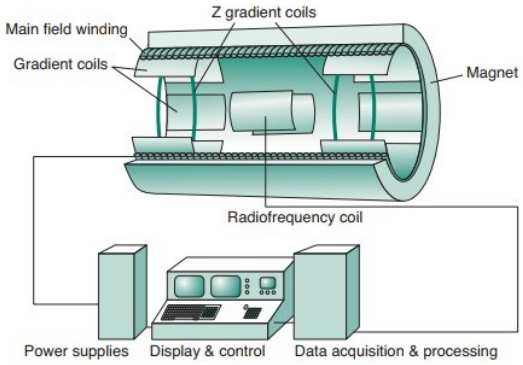


Figure 2.5: Main components of an MR system [15].

Firstly, an MR scanner with a magnet and coils is necessary to generate both the gradients and the RF pulses. Magnets are the basic and most external components of an MR scanner. They generate the main magnetic field, which should be homogeneous and have a stable strength (in many applications, the field values are between 0.1 and 3T). They must surround the patient, and they can have different shapes according to their characteristics and manufacturers. There are three different types of magnets: resistive, permanent, and superconducting magnets. The first ones are electromagnets that create a magnetic field with a maximum strength of 0.3T if they are connected to power supplies. Permanent magnets generate magnetic fields of up to 0.5T that can be maintained without power supplies. The last type is the most powerful and commonly used ones. They can produce magnetic fields with a maximum strength of 18T, which can also be maintained without additional power input [12]. They consist of a coil that should be cooled to about 4K to achieve the superconducting state. In order to control external fields, shieldings are usually used. They are two layers of windings located surrounding the magnet that create the field and provide paths for the returning magnetic field lines.

The gradient system MR scanners have, consists of three gradient coils and their corresponding amplifiers, which are placed in the three perpendicular spatial directions around the patient (see Figure 2.5). They should be steady as well, to avoid distortions in the MR images. Three main

aspects measure the performance of these gradients: the maximum gradient strength (measured in mT m^{-1}), the rise time (i.e., the time needed to acquire the maximum gradient strength), and the slew rate (quotient between the maximum gradient strength and the rise time).

The radiofrequency (RF) system has a powerful RF transmitter and a sensitive receiver, and they both should be stable. The first one delivers the RF pulses, while the second detects MR signals. It has additional coils, both receive and transmit/receive coils, which form windings encircling the patient. The selection of these coils depends on the body part under study and is essential to optimize the quality of the images.

The MR scanner is usually placed in a conductive enclosure that acts as a Faraday cage to reduce interference effects produced by external sources. In addition, a computer system is necessary to control the components of the scanner, coordinate all processes, and create images.

Finally, there are also other components in the MR system with specific functions that are important to ensure proper performance. Some of these devices are a control for the patient table and a cooling system for the magnet.

2.4. Quality of MR images

Factors like patient's body movement, errors while processing images, or equipment defects result in noise associated with MR signals. As they are used to reconstruct images, these images will not be as sharp as desirable. Signal-to-Noise Ratio (SNR) expresses the relationship between the desired signal and the noise (the undesired one). In Subsection 2.4.2 this quantity will be discussed in more detail. In addition, we will explore artifacts, which are structures that can be seen in MR images but do not exist in reality, and image contrast, which is a crucial aspect when differentiating tissues [12].

Before starting with the mentioned aspects, we should consider a factor that also affects the MR images' quality. It is the so-called scan time, i.e., the time needed for an image to be acquired. If this quantity is large, there are more possibilities of the patient moving, which would worsen the image. The scan time depends on the repetition time, the echo time, the number of phase encodings, and the number of acquisitions (number of times a slice signal is measured). They all should be reduced to shorten the scan time. However, as we will observe, it is not straightforward because SNR can also be affected. Then, we need to find a trade-off to optimize both scan time and SNR.

2.4.1. Image contrast

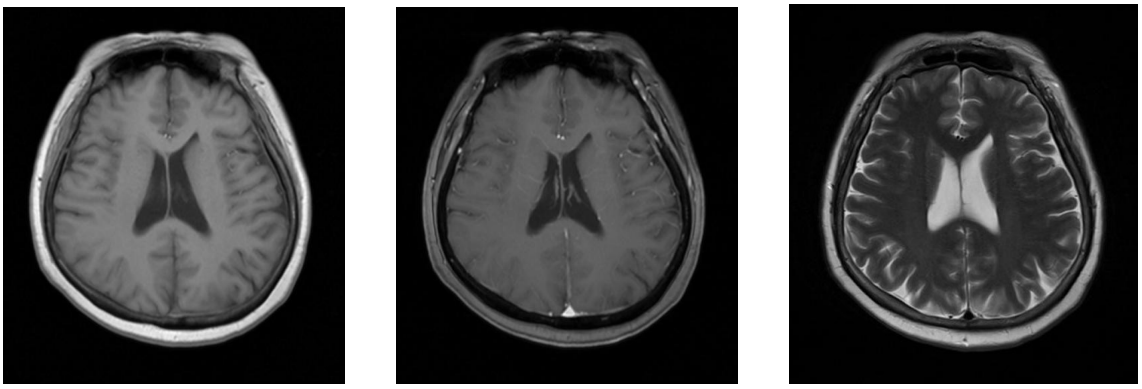
There are three main features of the analyzed tissue that affect the brightness or contrast of the MR image. They are the proton number density (number of protons per unit volume of tissue that contributes to the net magnetization vector), the T1 and the T2 times (see Subsection 2.1.2). Depending on which property is enhanced in an MR sequence, the contrast between tissues in the generated images changes. Images in each case are called proton density, T1 and T2- weighted images, respectively.

The quantity used to weight T1 in MR sequences is the repetition time (TR). If it is short, then only a few spins contribute to the next nutation movement of $\mathbf{M}_0$ produced by the second pulse. This means that the new MR signal will be weaker. Therefore, a short TR implies a strong

T1 weighting and vice versa. In T1-weighted images, tissues with shorter T1 appear brighter than those with longer ones (see Figure 2.6.a.). In addition, if TR is too short, the MR signal may become smaller after each excitation. This phenomenon is called *saturation*. A smaller flip angle excitation pulse is applied to minimize it so that spins have to move a smaller distance to point in the z-axis.

Echo time (TE) is the quantity used to determine the weight of T2. The larger it is, the stronger the T2 weighting will be, and vice versa. In T2-weighted images, tissues with shorter TE appear darker than those with longer TE (see Figure 2.6.c.).

A combination of both T1 and T2 effects is needed to get proton density-weighted images. This type of image is commonly used to evaluate structures of the brain or the musculoskeletal system, among others, as it has low signal intensities.



(a) T1-weighted axial brain image without contrast agent.

(b) T1-weighted axial brain image with contrast agent.

(c) T2-weighted axial brain image without contrast agent.

Figure 2.6: Examples of MR brain axial images of the same person taken with different sequences. They are included in the NYU Langone Health fastMRI dataset that we will use in Chapter 4.

Sometimes, the image contrast produced by these features is not enough to see differences between normal and pathologic tissues. In these cases, substances called contrast agents are applied (orally or intravenously) to enhance the contrast of certain tissues (see Figure 2.6.b.). The most important effect these contrast agents produce is the shortening of T1 or T2 relaxation times. Once they are applied, pathologic tissues in T1-weighted images become lighter, and in the T2-weighted ones, they become darker. It is common to use gadolinium, which is a paramagnetic substance (it has a positive magnetic susceptibility), in contrast agents, whose seven unpaired electrons interact with the hydrogen protons and produce the already mentioned effect.

2.4.2. Signal-to-Noise Ratio

On MRI, SNR is mathematically calculated as the quotient of the signal intensity in an area of interest of the image and the standard deviation of the signal from the background. For example, the signal of the air that surrounds the object, which would be null if there were no noise). It is desirable for this quantity to be high (and greater than one). The parameters SNR depends on are the following ones:

- **Echo time (TE) and repetition time (TR).** In order to increase SNR, TR should be enlarged, and TE shortened. Nevertheless, we should remember that this would imply a

loss of T1 and T2 contrast, respectively. A solution could be to increase TR in T2-weighted sequences and TE in T1-weighted ones.

- **Number of acquisitions.** SNR is proportional to this quantity, so it should be large in order to increase the ratio.
- **Slice thickness and receiver bandwidth (BW).** From a medical point of view, thinner slices are more interesting as they provide a better image resolution. However, it decreases the value of SNR, that at the same time can be compensated by increasing the number of acquisitions or the TR. Moreover, a narrow BW improves SNR, but it also produces more artifacts, and scan time increases.
- **Matrix.** As previously stated, MR images are matrices composed of pixels. Each one contains the information of signal intensity and is associated with a slice voxel. Spatial resolution depends on the size of these voxels and is better as smaller the size is. However, voxel size reduction leads to a loss of SNR.
- **Field of view (FOV).** It corresponds to the area of the object under study we get an image of. If the matrix has a fixed number of rows and columns, then the FOV directly determines the size of the pixels. Thus, pixel size increases with FOV, so spatial resolution is better, but SNR decreases.
- **Magnetic field strength.** The magnitude of the net magnetization is linearly proportional to the strength of the magnetic field, as we expressed in Equation (2.3). Therefore, the intensity of the signal will be higher if the strength increases, and so does SNR.
- **Coils.** Radiofrequency (RF) coils are important components in MR scanners (Section 2.3), whose functions are to transmit the RF pulses and receive the MR signal (or only the second one). A way to increase SNR, with no effect on both the scan time and spatial resolution, is by placing the RF coil as close as possible to the organ under examination and surrounding it.

2.4.3. Artifacts

As previously stated, artifacts are false structures or contrast alterations seen in MR images. They can be a source of wrong diagnosis, so it is essential to detect or prevent them if possible to avoid deleterious consequences for the patient. There are three main causes of their appearance: the tissues' molecular physics, the physiological patient behavior, and the MR scanner or techniques used in the image creation [13]. Artifacts can be divided into three groups according to this classification.

Among the artifacts caused by the molecular physics of the tissues under study, there are three different groups. Firstly, chemical shift artifacts are caused by a natural difference in the magnetic field applied to fat and water molecules because of their nature. It produces an additional variation of the precession frequency of hydrogen protons of both molecules once G_y is applied, which leads to a shift of the adipose tissue that can be seen in MR images as a bright and a dark border around the water tissues (see Figure 2.7). The difference in tissue composition also produces the so-called black contour artifacts in sequences that collect gradient echoes. Water molecules precess quicker than fat ones, so sometimes their protons cancel themselves, which is seen in MR images as a darker contour around the water tissue (see Figure 2.8). Fat suppression techniques can be used to suppress the signal of fat tissues or detect these tissues. Those based

on the water and fat frequency difference include changes in echo times to be able to characterize both adipose and water tissues. Other fat suppression techniques affect RF excitation pulses and the weight of T1 times. Finally, magnetic susceptibility artifacts are observed as distortions produced by the difference in the magnetic susceptibility of the substances of the excited body area.

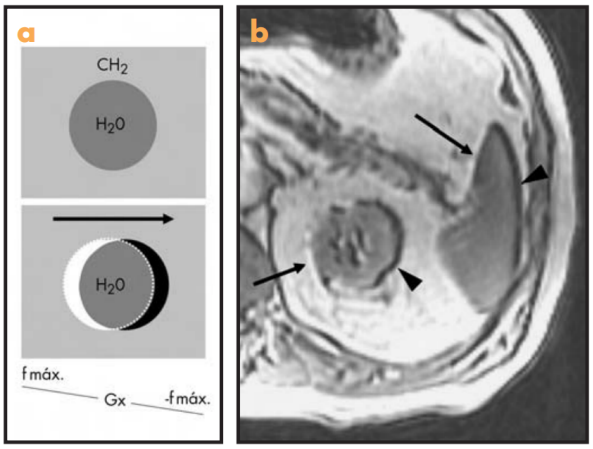


Figure 2.7: Chemical shift. a) A scheme of the phenomenon of chemical shift. b) A real MR abdominal image with this type of artifact [13].

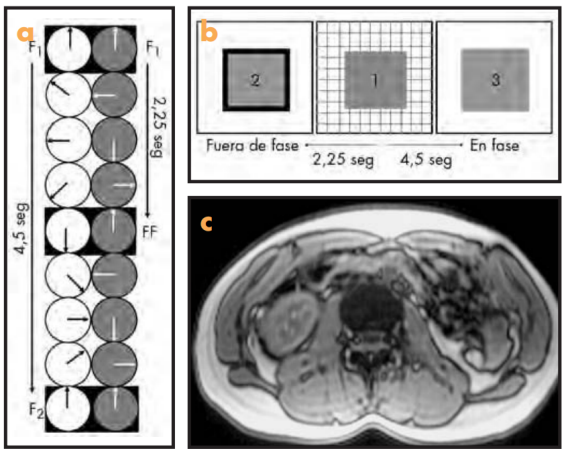


Figure 2.8: Black contour. a) and b) Schemes of the phenomenon of black contour. c) A real MR abdominal image with this type of artifact [13].

We will not work with this type of artifact because it is difficult to generate them artificially, and we could not find any large dataset that includes them.

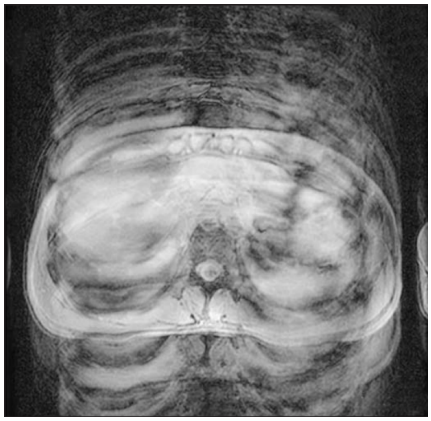


Figure 2.9: Ghosts in an MR abdominal image due to the patient's breathing [13].

Figure 2.10: Flow artifacts that can be seen as signal reductions in a) spinal cord and b) brain MR images [13].

Motion and flow artifacts (see Figure 2.10) are produced due to the patient him or herself. The first ones are caused by natural processes like breathing, the beating heart, and peristalsis. They produce both blurring and unreal structures (ghosts) in MR images (see Figure 2.9). There are algorithms, additional RF pulses, and agents that suppress muscle movements that can be utilized to reduce their effects. On the other hand, flow artifacts are caused by the flow of blood and brain liquid, among others, and only in the phase-encoding direction (y direction). As the spins of these substances are moving with them, when magnetic field gradients are applied they can have assigned a wrong phase value and therefore be depicted in an incorrect place in the

image. These artifacts can be prevented by using special pulses, by saturating the blood (which produces no signal), and by changing the orientation of both the frequency and phase-encoding gradients.

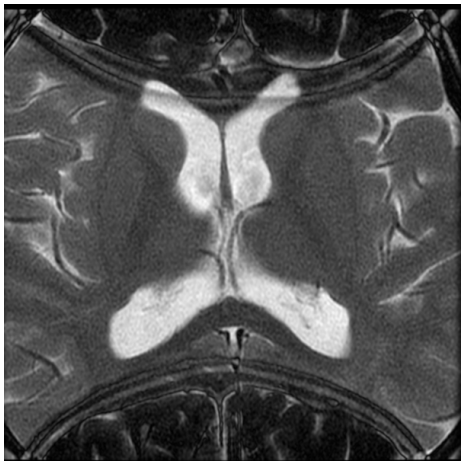


Figure 2.11: Aliasing artifact in a MR brain image. Case courtesy of Dr Usman Bashir, `https://radiopaedia.org/`, rID: 16491.

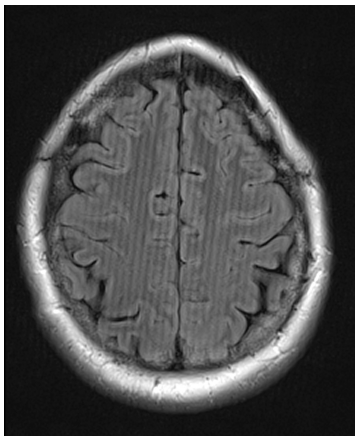


Figure 2.12: Spike artifact in a MR brain image. Case courtesy of Assoc Prof Frank Gaillard, `https://radiopaedia.org/`, rID: 19695.

Finally, there are different artifacts among the ones produced due to the MR scanner or the techniques used. Phase wrapping (also known as aliasing, Figure 2.11) is one of them, which occurs when the body part under analysis is larger than the FOV and causes a projection of the body part outside the FOV onto the opposite side of the image. It can be prevented by defining a larger FOV, by using special algorithms, or by saturating the regions outside the FOV. Moreover, the Herringbone artifact, also known as spike artifact, is generated by electromagnetic spikes, gradient coils, or fluctuating power supplies that produce aberrant points in k-space and looks like oblique lines throughout the MR image (Figure 2.12). Another common artifact is the so-called bias field artifact, which is produced by imperfections in the field coils used in the MR scanner, or by magnetic susceptibility changes at the boundaries between anatomical tissue and air. It can be seen as dark shadows in the image.

Chapter 3

Deep Learning to improve images' quality

Deep Learning is a set of automatic learning algorithms aiming to obtain the best data representations using computational architectures. Those representations are learned in successive layers, composed of neurons that imitate the biological neurons' functions, and this layout is what 'Deep' refers to. The depth of the model, also called a neural network, is determined by the number of layers it has. As we will see, the parameters that will help determine the optimum representations are updated simultaneously, which makes Deep Learning an outstanding technique among those included in Machine Learning. Even though the first steps in Deep Learning were made in 1989 by Yann LeCun et al., it was not until the last decade that it has become more relevant [16]. The main reasons are the recent developments in hardware, the possibility to access larger datasets on the internet, and also the development of optimized algorithms to train deep network architectures (which was possible thanks to the first two reasons).

Supervised and unsupervised learning are the two main different types of learning. In the first type, we need to have the data correctly labeled so that the neural network can learn by using solved examples. However, in the unsupervised learning approach, we do not have data labels, so the model solves the problem by extracting the information it needs from the input data. There are other learnings approaches not so used in medicine. One of them is called semi-supervised learning, which combines the two explained types of learning, and another is reinforcement learning, which consists of learning the optimal behavior in an environment to maximize the so-called reward [17]. From now on, we will focus on supervised learning, which is the one we will use in our models.

In this chapter, the main aspects of the learning process for our particular case, as well as the application of convolutional neural networks to images, will be explored. We will also explain a tool that will allow us to generate artifacts in high-resolution medical images.

3.1. The learning process of a regression problem

We aim to use Deep Learning to improve images' quality. Therefore, we are facing a regression problem, which differs from the classification one in the fact that its goal is to predict continuous values (the corrected image) and not the class (an integer number) they belong to [18]. We will explain in this section the learning process oriented to this type of Deep Learning problem.

3.1.1. General aspects of the learning process

Among the different aspects we will explore in this subsection, we will start with the two main parts that can be distinguished in any learning process. They are the following:

- **Training:** It includes the construction and learning process of the model. We have some initial low-quality images (inputs), which are introduced in the model. It returns predictions, which correspond in our case to the corrected images, and they are compared with the true targets of the inputs (the sharp images). This comparison is what allows the model to learn and contributes to its update to improve its predictions. The so-called *epochs* are the number of times the model works through the entire training dataset. Then, we can consider it an iterative process. The model training scheme is represented in Figure 3.1.

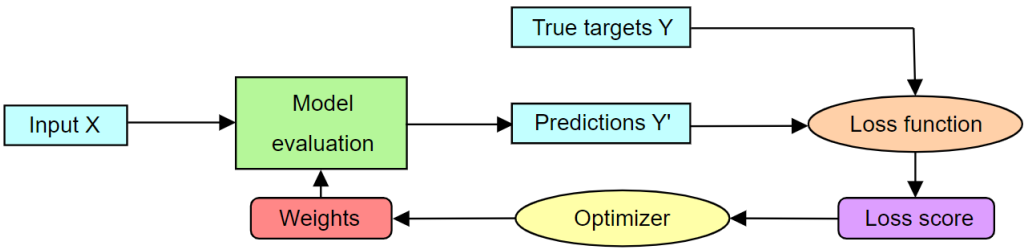


Figure 3.1: A scheme of the steps followed in the supervised learning process.

- **Test:** Once the model has been constructed and has learned from the training dataset, it has to be tested. To do so, we introduce a new dataset into the model, making sure that the images it contains have not been used during the training phase. The test step aims to evaluate the performance of our model, so once it returns the predictions we have to compare them with the true targets, which we know in advance.

In order to monitor and evaluate the learning process during the training, we can use the value given by the loss function, as can be observed in the above figure. It should decrease as the number of iterations increases until a quasi-constant minimum value is achieved. This function quantifies the error between the prediction the model returns and the true target. Before showing its expression, it will be helpful to introduce some terms first.

Let us take W_i with $i \in \{1, ...N\}$ the net's *weights*, i.e., the parameters that are updated during the training process, and N being the number of layers it has. Be b_i with $i \in \{1, ...N\}$ the so-called *bias* terms. Then, the first computation performed in the initial layer is

$$z_1 = W_1\mathbf{X} + b_1, \tag{3.1}$$

where $\mathbf{X}$ is a tensor that includes the inputs. However, z_1 is not the output of that layer. Before moving to the next one, z_1 should be transformed by a non-linear function called *activation function*, g, such that $a_1 = g(z_1)$. Many types of mappings could be used as activation functions. We will use the Rectified Linear Unit (ReLU) one, such that $g(x) = \max(0, x)$, where x is a particular pixel. This function works properly, particularly with images. a_1 is the tensor that should be used in the same computations of the next layer. This process is repeated along the whole model until we finally obtain a_N, which contains the information of the corrected images, i.e., the predictions. Once we have them, we can easily compute the error.

Many functions can be used as loss functions, and their selection depends on the problem we want to solve. The loss function that we will use for our regression algorithm is the Mean Squared Error (MSE). Given the input images $\{\mathbf{X}_i\}$ and the predictions $\{\mathbf{Y}_i\}$ returned by the model in a certain iteration, it can be defined as follows:

$$L_{MSE}(\{\mathbf{X}_i\}, \{\mathbf{Y}_i\}) = \frac{1}{n}\sum_{i=1}^{n} ||\mathbf{Y}_i - \mathbf{X}_i||^2, \tag{3.2}$$

where n is the number of samples in the set.

Apart from the loss function, the training process can also be monitored by the accuracy, which should increase as it does the number of iterations. Many metrics could be considered to determine it. In problems of image quality restoration, the most widely-used metric is the Peak-Signal-Noise-Ratio (PSNR), measured in dB, which relates the maximum image signal intensity and the distorting noise it has. It is described as follows:

$$PSNR(\mathbf{X}_i, \mathbf{Y}_i) = 20 \cdot \log_{10}\left(\frac{MAX_i}{\sqrt{L_{MSE}}}\right), \tag{3.3}$$

where MAX_i is defined as the maximum value a pixel of image $\mathbf{Y}_i$ could take. Note that it tends to infinity as the predicted image approaches the original one. It is common to use this metric when the MSE is working as the loss function because both quantities are related.

However, even though this metric is very simple to calculate, its value does not represent at all the perceptual visual quality of the images [19]. For the sake of comparing the performance of the model by different approaches, we will also use the Structural Similarity (SSIM) one (see [20]). This index is influenced by three attributes of an image: the luminance (l), the contrast(c), and the structure (s), such that

$$SSIM(\mathbf{X}_i, \mathbf{Y}_i) = f(l(\mathbf{X}_i, \mathbf{Y}_i), c(\mathbf{X}_i, \mathbf{Y}_i), s(\mathbf{X}_i, \mathbf{Y}_i)) = \frac{(2\mu_X\mu_Y + C_1)(2\sigma_{XY} + C_2)}{(\mu_X^2 + \mu_Y^2 + C_1)(\sigma_X^2 + \sigma_Y^2 + C_2)}, \tag{3.4}$$

where μ_X and μ_Y are the averages of the low-resolution image $\mathbf{X}_i$ and the corrected one $\mathbf{Y}_i$, respectively. Moreover, σ_X^2 and σ_Y^2 are the variances of $\mathbf{X}_i$ and $\mathbf{Y}_i$. Finally, σ_{XY} is the covariance of both images, and C_1 and C_2 two constants. The maximum value of this metric is 1, and it is reached only if $\mathbf{X}_i = \mathbf{Y}_i$.

A variation of this metric is the Multiscale Structural Similarity (MS-SSIM) (see [21]). It also measures the luminance, contrast, and structure of the image. However, in this case, images are re-scaled a certain number of times, and the SSIM is computed in each scale. It is a way to incorporate image details at different resolutions into the measurement. For that reason, the MS-SSIM provides a more consistent result than SSIM compared to what we can visually perceive in the image.

Nevertheless, although both the loss function and the accuracies provide good results during the training process, there are still some situations that could appear, which would give rise to poor predictions when using the trained model.

On the one hand, if the model is very simple, and does not distinguish right between images' particularities, then we are in a situation of *underfitting*. The commonly-used solutions to solve

this problem are using a more complex model architecture and collecting more data or modifying the one we have (*data augmentation*) so that we can train with more images.

On the other hand, we can have *overfitting*, which could be caused by using complex models or a low number of images for the training. It can also be solved by the addition of more data or by the so-called *regularization*, which can be used to reduce the model's complexity.

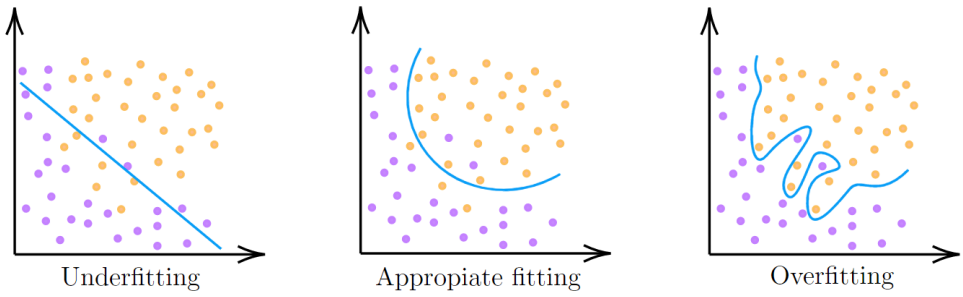


Figure 3.2: Fitting situations that can appear when a model is used to classify some data (points) into two different groups. The group those points belong to can be identified by their color: yellow or purple. The result of the classification provided by the model is shown as a blue curve.

Figure 3.2 shows an academic example of a classification problem (see [22]), in which we can distinguish the three cases that we could have: underfitting, appropriate fitting and the overfitting.

As we have explained, two different datasets are needed when we work with models. In particular, the training dataset is divided into training and validation, such that we can perform the model hyperparameters' tuning. Therefore, we need to have the following dataset division to complete the learning process:

i) **Training set:** It is the set with a higher number of images, and they are those utilized to train the model. Each of those images with a kind of artifact should also have linked the sharp image, such that it is possible to get the loss value and the accuracies once the prediction is returned by the model. The net uses this set to modify the parameters it has for the purpose of achieving better results for the regression problem.

ii) **Validation set:** It is also used in training but it is not utilized to modify the model's parameters. Its main objective is to perform the model hyperparameters' tuning. Moreover, this set of images can have a control function, as both the loss and the accuracies can be performed with them so that we can check whether the process is running successfully, but this evaluation is biased. Situations of poor fittings can be detected by using this set. The net goes through this dataset after each epoch.

iii) **Test set:** The number of images in this set is lower than in the *training set*. It should not be used in training and aims to definitively evaluate the model's performance without bias (unlike the validation set).

The percentage of images that should be considered in each set is not fixed. The decision of this parameter depends on different aspects, such as the dimensions of the dataset we have and the structure of the model. In our case, we will choose 60% of the total dataset for the training and 20% both for the validation and the test sets.

3.1.2. The training process as an optimization problem

Once the loss function is calculated, the model weights have to be updated. Determining the most effective ones is the main objective when training a net because they provide the connexions between the different model's layers. It is possible thanks to the optimizer.

A widely-used optimization algorithm is Gradient Descent (see [23]), which minimizes the functions by iteratively moving towards its minimum. According to the number of samples considered before updating the parameters, we have different convergence situations. In our case, we will consider the Mini-Batch gradient descent, which takes a certain number of images, typically a power of two because of computational reasons, like 32, 64, 128..., depending on the size of the dataset used and the memory of the CPU/GPU used. The size of the mini-batch is also a hyperparameter that has to be fixed at the beginning of the learning process.

Some modifications could be made to the Gradient Descent to improve its efficiency. Particularly, we will use the Adam Optimizer (see [24]), which aims to converge faster to the minimum by using algorithmic characteristics of other methods (RMSProp [25] and gradient descent with momentum [26]). Once we have W_i, and b_i, the weights and bias of layer i, they can be updated as follows,

$$W_i = W_i - \alpha A(dW_i), \qquad b_i = b_i - \alpha B(db_i), \tag{3.5}$$

where $A(dW_i)$ and $B(db_i)$ are two functions that depend on the weights and bias derivatives, respectively. α is a variable known as *learning rate*. It is also a hyperparameter, and, as we can see, its value determines the variation of weights and bias in each iteration. It is essential to take an adequate learning rate because if it is too high, we are more likely to be wrong in our predictions. This is because the model might converge too quickly to a local minimum (not the optimum solution). In addition, if it is too small, the running time would be large, and it is possible that the model gets stuck and never reaches a global minimum.

Among the different neural networks that can be used, the convolutional ones (CNNs) have revolutionized the field of artificial vision. This type of network is the one that we will use for the simulations, and it will be explored in the following section.

3.2. The use of convolutional neural networks in artificial vision

Neural networks learn characteristic patterns of the inputs, making them useful for particular applications. Unlike dense networks, convolutional ones learn local patterns, which are translation invariant. This means that they recognize image characteristics independently of where they are in the sample. It makes them highly efficient in image processing problems. In addition, to train those models, fewer samples are needed because they can generalize.

Another representative aspect of CNNs is the fact that they learn about the images hierarchically, i.e., they firstly learn simple characteristics of the training images, while the deepest convolutional layers combine them to create more complex distinctions.

2D *filters* are used by the net to find the different patterns of an image. They are matrixes of n rows and m columns (both n and m should be smaller than the image side lengths) that move through the image and multiplies themselves with the pixels of a specific region of the image (known as a patch) in each step, resulting in a lower-dimension matrix. This process is

known as *convolution*, and the resulting matrix is called a feature map, which contains localized information about certain characteristics of the initial image. There are as many feature maps resulting from a convolution as filters (all of them with the same size) the layer has. Once we apply the activation function to these maps, the resulting ones will be the input of the following convolutional layer, which has distinct filters. The process is sequentially repeated until we reach the last layer, whose output is the prediction of the model.

The values of the filters' matrix are the weights to be optimized. Both the number of filters included in each layer and their size are hyperparameters that have to be fixed when building our net. During the training process, they will be updated to minimize the loss function.

Concerning the convolution operation, there are some different parameters that we need to take into account when constructing the architecture of our model:

- *Stride:* It is the size of the step taken by the filters through the images. It is common to utilize 1-pixel-steps, but taking a larger number would simplify the model. This is because the number of parameters the net has to learn is reduced as stride increases, as we will see in the below expressions. However, if it is too high, the information of the features transferred from the image to the map will be less precise.
- *Padding:* If our model has many layers, the final image will be extremely small. Moreover, the information of the pixels in the image's borders will be underrepresented compared to the rest because they are multiplied by the filters only once. We can add extra frames of empty pixels around the image to avoid these situations. This process is known as padding.

Once we have described those terms, let us show the structure of a $n \times n \times c$ image ($c = 1$ if we consider a gray-scaled image, and $c = 3$ if it is colored) after a convolution operation, with padding p, stride s, and k $f \times f$ filters. Typically, squared filters with an odd f value (like 3 or 5) are used. The output of the corresponding convolutional layer has the following size:

$$\left\lfloor \frac{n + 2p - f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n + 2p - f}{s} + 1 \right\rfloor \times k.$$

As we have observed, convolution allows us to have connections sparsity, i.e., it supposes a reduction in the number of parameters the model needs to learn compared to the one needed by a classic dense network. In particular, let us take the convolutional layer of the previous example. The number of parameters that the model should update just for this layer is

$$\text{n. parameters} = (f \cdot f \cdot c + 1) \cdot k, \tag{3.6}$$

i.e., we consider the values of the 2D or 3D filters (if we work with gray-scale, or colored images, respectively) plus the bias term of each of them.

The number of parameters is commonly computed to provide information about the complexity of the model. If it is high, the training will demand more time, which is an aspect that should also be taken into consideration.

3.2.1. CNNs used in this work

Once we have explored how CNNs work, we can focus on the ones we will use to remove artifacts from poor images.

We will utilize a network based on the one proposed by Dong et al. in 2014, called super-resolution convolutional neural network (SRCNN) (see [2]). It is a simply-structured model that provides outstanding results. In addition, the number of layers and also the size of the filters added are moderate (< 10), so the learning process is fast. To get an idea, the corrected images provided in article [2] has a PSNR range of 25.60-35.66, SSIM values between 0.7184 and 0.9542, and an MS-SSIM range of 0.9261-0.9959. We can see the general structure of the network in Figure 3.3.

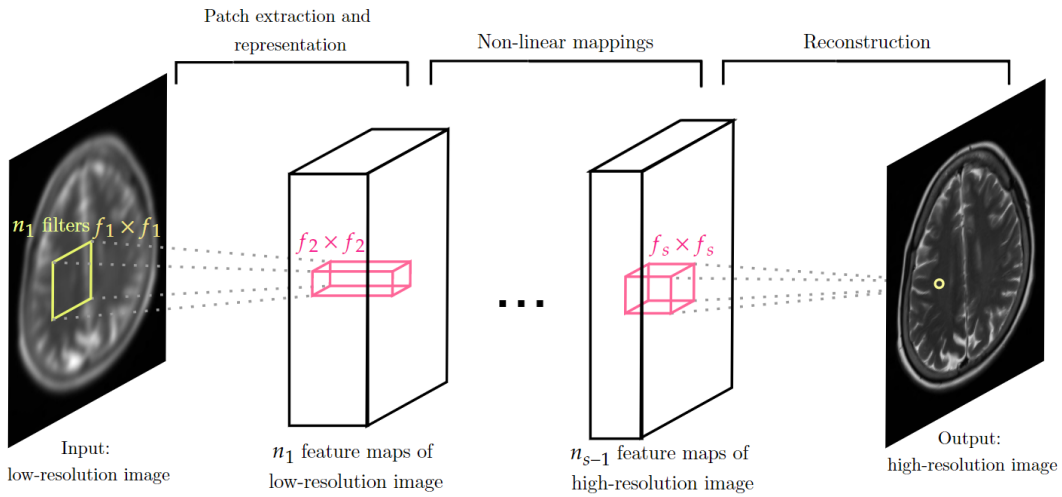


Figure 3.3: Internal operations of SRCNN. We have a low-resolution image as input, and the first convolution layer of the network extracts a set of feature maps by patches. Then, the intermediate layers produce continuous non-linear mappings until we reach high-resolution patch representations. Finally, the last layer produces the final high-resolution image as output.

Three main operations should be carried out by the layers of the model to obtain as a model prediction a corrected image as similar as possible to the sharp one:

- **Patch extraction and representation:** The first layer of the convolutional network extracts a set of feature maps following the process explored previously . As it was already mentioned, the number of feature maps is set by the number of filters.
- **Non-linear mappings:** It is the process carried out by the intermediate layers of the network. It consists of some non-linear mappings, equal to the number of intermediate layers, that are applied sequentially and obtain high-resolution patch representations.
- **Reconstruction:** The last convolutional layer's filters take the resulting high-resolution patches of the previous operation and process them to produce the final high-resolution image.

The number of layers and filters are parameters we should fix when we create the net architecture. The size of those filters will also be fixed. We will observe the changes produced when we vary them and compare them with what we would expect (see Section 4.4).

As its name suggests, this model is intended to correct images that have low resolution, which happens when they have too few pixels, too much compression, or both. For this purpose, Dong et al. upscale the initial images to make them have a lower resolution, and this result is what they take as the model's input. In our case, we will not perform upscaling but a previous step

in which we will generate the artifacts.

Moreover, we have to consider the fact that training a model like this could not be generalized to all types of artifacts. Therefore, we would have to train those networks with datasets that include only one particular type of artifact, and we would be able to remove only this artifact from the input image. This situation will be analyzed in the following chapter (see Subsection 4.5.2).

3.3. The artificial addition of artifacts by using TorchIO

Processing medical images usually presents challenges, including a lack of large datasets to train and high computational costs. It is the first issue the one that we are dealing with here.

As we have mentioned, we can have fitting problems if there is not enough data. However, it is not always easy to access medical images due to a variety of reasons. These include concerns over patient privacy if we work with images from real patients, and the time and costs needed to acquire those images only for clinical trials. Traditionally, computer vision libraries[1] use to overcome those difficulties by supporting data augmentation that includes simple operations like rotations or zooms, and others like blurring for colored images. When working with medical images, the purpose of data augmentation is to simulate artifacts and anatomical variations so that we can train with datasets that include those problems that appear in real life.

TorchIO is a Python library designed to be integrated into Deep Learning workflows (see [27]). It allows efficient preprocessing and augmentation, among others, helping researchers to focus on their experiments without worrying about the data preparation.

Our main objective in this work is to remove artifacts from medical images using CNNs. However, it is not common to find large datasets that include those low-quality images that also incorporate the corresponding sharp images. As a result, we have to use TorchIO in order to generate different types of artifacts so that we can use them, and also the original images, to proceed with the supervised learning process.

Among the different interfaces TorchIO includes, we will use the one called *transforms*. It provides flexibility in input/output formats, which is an advantage when programming. The transforms that can be performed with this library can be classified into spatial and intensity transforms. The first one includes resampling and reorientation, i.e., modifications of the voxels positions typically used in medical imaging. However, we will focus on the second class: the modification of the values of those voxels, but not their positions.

As we learned in Chapter 2, MR images are reconstructions of signals in the k-space by using inverse Fourier transforms. TorchIO is especially interesting when working with this type of medical image because it provides transforms related to k-space. In particular, it generates artifacts by perturbating the Fourier space, producing intensity artifacts in image space. The library uses the Fast Fourier Transform algorithm to access the k-space. Among the k-space artifacts that it can simulate, we will use:

i) **Random k-space spike artifact:** It is simulated by the addition of noise spikes in the k-space of the corresponding image.

[1] A library is a collection of prewritten code that can be used by the users to optimize tasks.

ii) **Random k-space motion artifact:** The k-space is filled with random rigidly-transformed versions of the sharp images, so in the image space, it can be seen as a repetition of the structures. It is based on the implementation of Shaw et al. (see [28]).

iii) **Random k-space ghosting artifact:** It is simulated by removing every nth plane of the k-space along a certain direction. The center of the $k-$space should not be removed. Otherwise, we would lose essential image information, and the result would no longer be a ghosting artifact.

iv) **Random bias field artifact:** To generate those artifacts, polynomial basis functions, i.e., elements of a basis that describes a function space, are used.

We will also simulate blurriness and random noise, which are not specific transforms for MRI, but they can also be obtained in real life.

To sum up, our interest in TorchIO lies in the possibility it has to simulate artifacts in MR images that we can use as input for our neural networks to train our models so that they learn to remove them. This library still has some limitations with other image modalities like computed tomography, but it is beyond the scope of our work.

Chapter 4

Simulations

So far, we have explained the magnetic resonance phenomenon that gives rise to the images we will work with and whose defects we want to correct. We have also explored the structure of the convolutional networks that we will use to eliminate them and how they must first be generated. Therefore, the next step is to train the super-resolution networks that aim to remove the defects and test their effectiveness.

To do so, we will first describe the dataset we use in the dissertation. Then, we will see some images with artifacts that have been transformed with TorchIO to understand what we can achieve with this library. Furthermore, we will explain several aspects related to the training process of a model that we will take as a baseline and the test results it provides. We will compare them with the ones provided by other models with some modifications from the first one. Finally, we will perform some extra tests on the base model to analyze its overall performance.

4.1. Open Dataset

The MR images we are going to use for our analysis are provided by NYU Langone Health and are publicly available for those who are interested (see FastMRI Dataset in [29]). Among all the datasets they have, we will use coronal brain images provided in DICOM format.

The DICOM format, an acronym for Digital Imaging and Communications in Medicine, is one of the most widely used in healthcare. It allows the storage of not only medical images but also additional data necessary for their interpretation. These include, among others, information about the patient, aspects related to the equipment used, and the magnetic sequence with which it was obtained. In our work, we only need images, so the first thing we have to do before training our models is to transform that format into jpg.

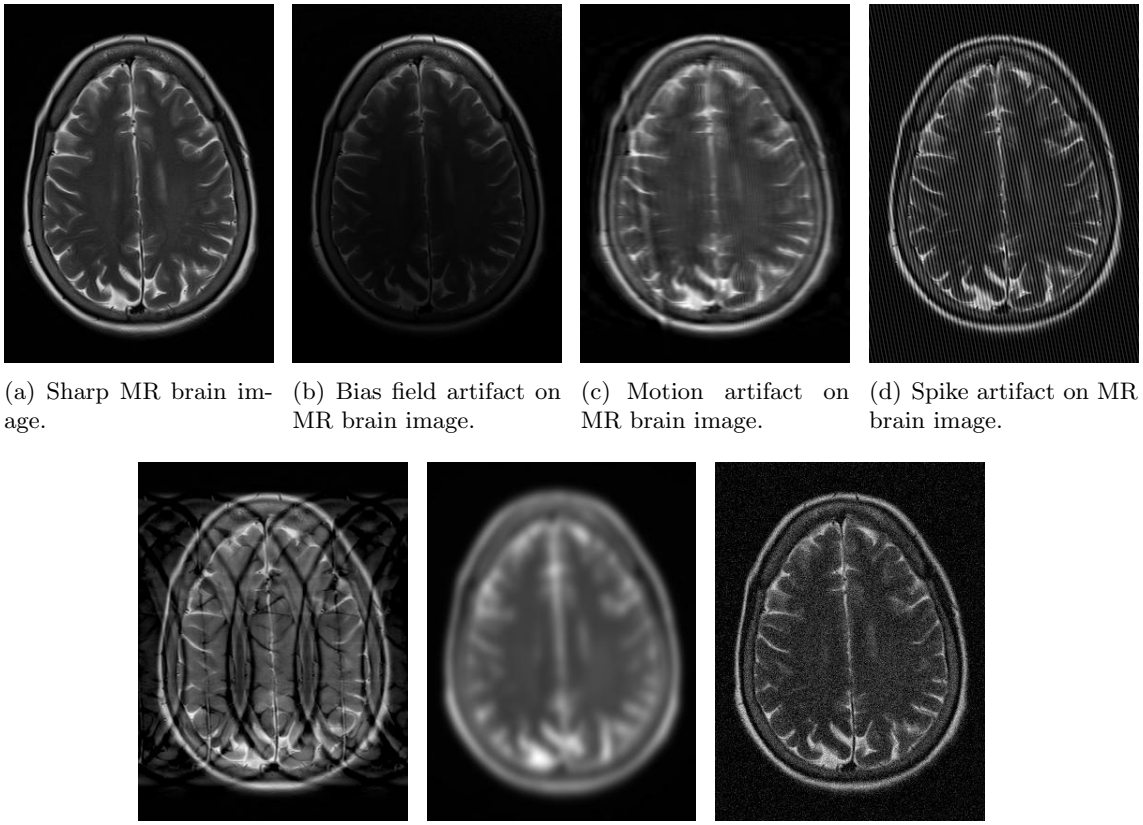
With regard to the dataset images, they have not been filtered, so it is possible to find brains with diseases like cancer. Neither have they been corrected for possible artifacts, so images with this type of problem could also be seen. In addition, it is worth mentioning that in some cases contrast agents have been used, so the appearance of the images may also vary. Moreover, images have not been obtained with a unique type of sequence. The echo and repetition times are not the same in all cases either. None of these aspects will be a disadvantage when working with convolutional networks. It is because, when working with supervised learning, the image to be improved is always compared with the original one, regardless of what this image shows.

As we have mentioned in Section 2.3, some of the components of an MR system may vary depending on the manufacturer, so there could be differences in the resulting MR images. In this dataset, the brain images come from different scanners so that the networks to be trained will not be specific to a particular manufacturer.

Finally, regarding the dataset structure, it contains folders corresponding to the studies performed (each study corresponds to a patient that has undergone the MR). Each of these studies contains images generated with different magnetic sequences so that more information can be collected from the brain under analysis. In addition, each sequence has images at different head heights, i.e., different slices of the brain. The total number of images in the dataset is about 200,000. Nevertheless, due to computational performance problems, we will only use 20,000 of these images.

4.2. Preprocessing

As explained in the previous chapter, once we have the original images and in order to introduce them in the network, we have to transform them with the TorchIO library. As a result, we will have both the sharp original images and those with artifacts and image defects created artificially. It is worth mentioning that the effects generated are random, which means that both the orientations of artifacts and their intensity vary with the image on which they are created.



(a) Sharp MR brain image.

(b) Bias field artifact on MR brain image.

(c) Motion artifact on MR brain image.

(d) Spike artifact on MR brain image.

(e) Ghosting artifact on MR brain image.

(f) Blurring on MR brain image.

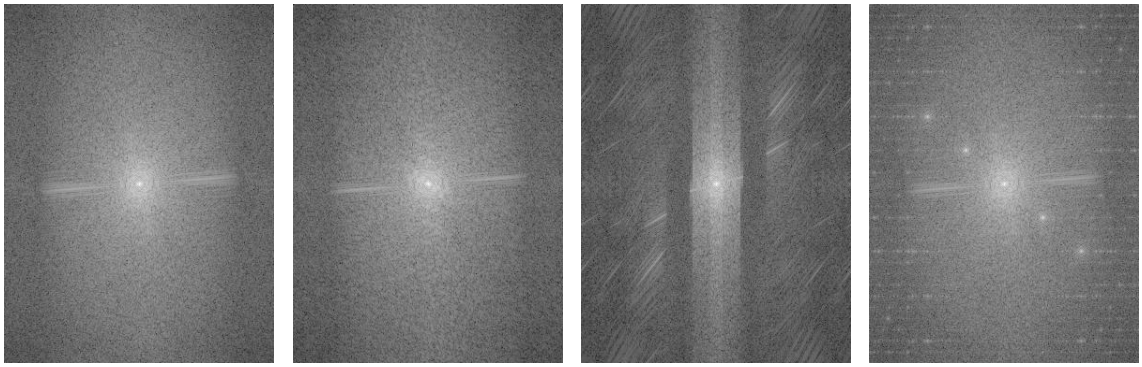
(g) Random noise on MR brain image.

Figure 4.1: MR artifacts and image defects created on the same brain MRI using TorchIO.

Figure 4.1 displays an example of an image that has been transformed using six different effects. Image (b) shows the effect of the bias field. It is clear that the objective of training the model, in this case, is to improve the contrast and illumination of the brain so that it is possible to homogenize the image. In the case of image (d), we can observe the lines in the MR image, which we want our neural networks to be able to eliminate. It is worth mentioning that the thickness of these lines is also a parameter that varies according to the image. Regarding the noise artifact (image (g)), we would like to eliminate this effect. However, it would not be enough just to blur the image so that the wrong pixels are not visible, but we need it to be seen clearly.

On the other hand, certain similarities can be seen between the motion and ghosting artifact and blurring defect. The first is considered as a combination of the other two since the image can be slightly blurred while showing structures that do not exist. Therefore, we would expect the performance success of the models that correct each of these three artifacts to be similar.

As it was explained in Section 3.3, the TorchIO library adds the artifacts to the image through k-space transformations. Thus, it is interesting to see the modifications made by the library on the image for each case. They are represented in Figure 4.2. Note that they have central symmetry, as was mentioned in Subsection 2.2.1.

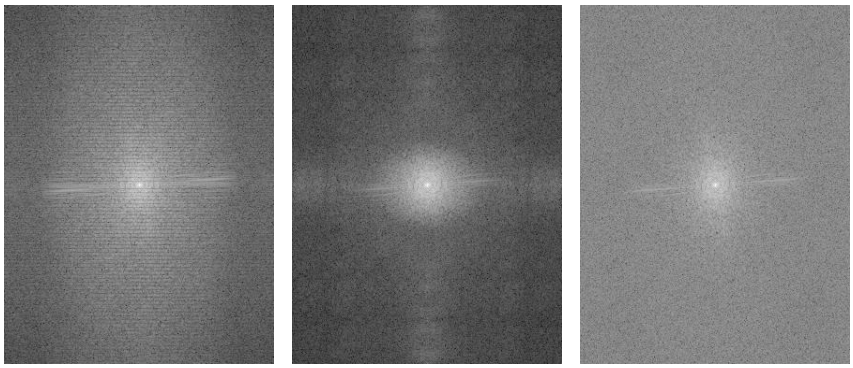


(a) k-space of the sharp image.

(b) k-space of the bias field artifact.

(c) k-space of the motion artifact.

(d) k-space of the spike artifact.



(e) k-space of the ghosting artifact.

(f) k-space of the blurring.

(g) k-space of the random noise.

Figure 4.2: k-spaces corresponding to the images of Figure 4.1.

First, the k-space of the bias field artifact image is very similar to that of the sharp image. We can see a slight difference in the center of the k-space, as it appears that the one with the artifact is distorted with respect to the one of the sharp image. This modification is consistent with the

fact that the central part of the k-space is related to the image contrast, which is affected by the bias field artifact. In addition, in the k-space of the image with the ghosting artifact, there are not many changes compared to the original space, although the thin lines that give rise to the structures called ghosts can be seen. It is noteworthy that when the lines are horizontal, the ghosts are vertical, and vice versa.

Among the artifacts whose k-space looks most different from the one of the original image are the spike and motion artifacts. The first one shows the multiplied centers described in Section 3.3, those that give rise to the lines we can see in the images. Regarding motion artifact one, we observe a clear difference compared to the one observed in image (a). There are not only dark vertical lines but also some lighter diagonal ones so that we have the combination of defects previously mentioned. Note that the center of the space is not changed, so the image contrast is not affected.

Finally, regarding the blurring and noise defects, although TorchIO does not generate them from k-space, it is also interesting to see their appearance. Image (f) shows a darkening in the outer part of the space compared to image (a). Note that this modification is reasonable considering that this area contains information about the spatial resolution of the image, which is the property affected by the blurring. In the case of image (g), a homogenization in the grayscale is observed in all the space except for the central part, resulting in the noise texture of the real image.

4.3. About the baseline model of our work

Once we have all the images with artifacts and their corresponding sharp images, we can train the convolutional networks explained in Subsection 3.2.1. We will start with a three-layer model we will take as a baseline for future analysis.

This first model has 64 filters of 9×9 in the first layer, 32 of 1×1 in the second layer, and a last filter of 5×5 for the final layer that allows the reconstruction of the image. We will consider a stride of 1, and the padding will be the one that allows the output image to have the same size as the input image in each layer. We will follow this criterion not only in this particular network but also in the following models. Throughout this dissertation, we will call it the 9-1-5 model.

Once we know the characteristics of the model to be trained, it is possible to calculate the number of parameters it will have to learn. In all cases, the input images will have a size of 224×224, and they are grayscale. Then, using expression (3.6), we obtain that the total number of parameters is 8129, most of them belonging to the first layer.

4.3.1. Filters and feature maps

As we have already explained, filters are responsible for extracting the necessary information from the images to achieve the objective for which the model is trained. As images, they are matrices with values that can be represented in grayscale.

The figure below shows the 64 filters of the model's first layer that result after training the model to remove noise:

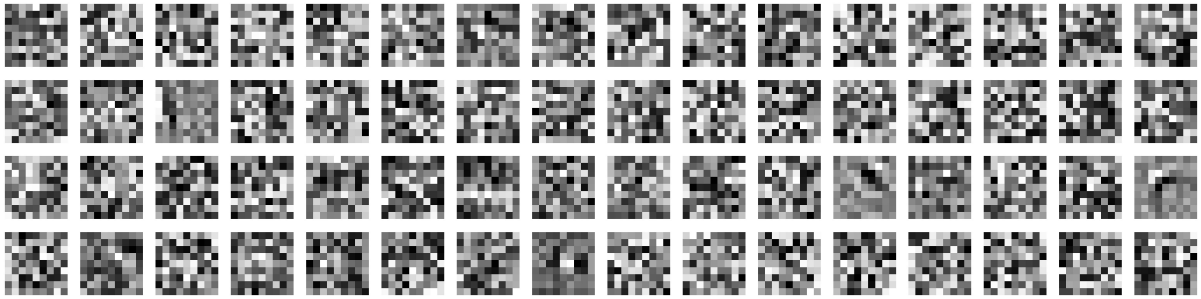


Figure 4.3: The 64 filters 9×9 included in the first layer of the 9-1-5 model once it has been trained to remove noise.

As it can be observed, they correspond to abstract representations, and all of them are different. This allows each of them to have a function within the image recognition. Some of these filters will be responsible for extracting texture information from the image, while others will be in charge of extracting the edges.

Figure 4.4 represents some of the feature maps that result from applying different filters to the image. They correspond to the denoising of an image. It can be seen that each of them provides information from different aspects of the image. For example, in the first layer, the gray maps contain information about the edges of the brain, while the other two show the contrast between the different parts of it. We can also observe that the resulting maps in each layer are similar, so we understand that all layers focus on the same aspects of the images.

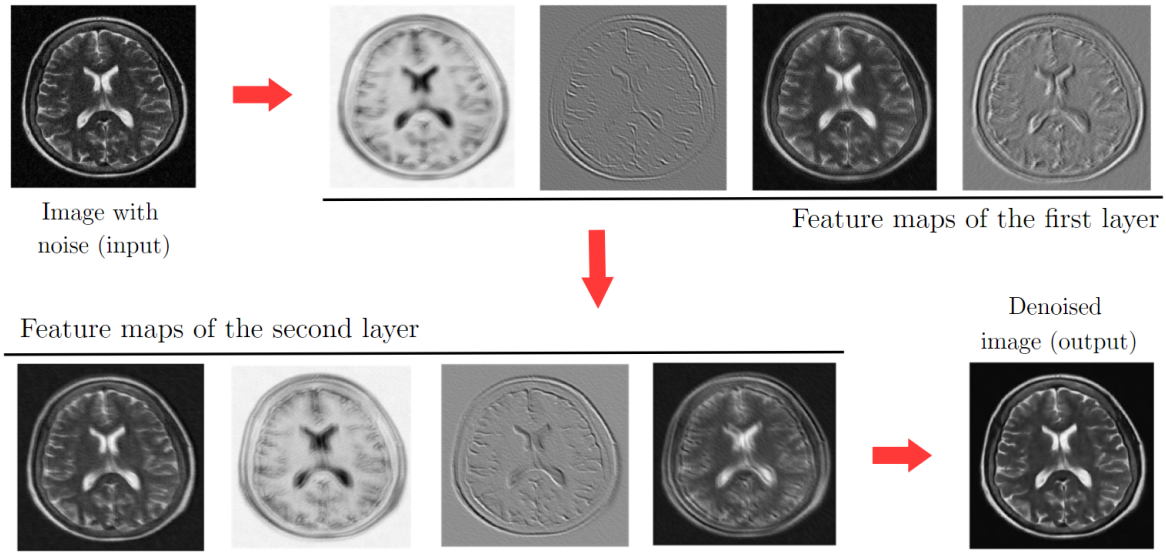


Figure 4.4: Some of the feature maps resulting in each layer.

4.3.2. Training loss and accuracies during the training step

Before showing the results provided by the model, we can analyze the training process. As we studied in the previous chapter (see Section 3.1), during the process, the value of the loss function decreases as the number of epochs increases. The accuracy, on the contrary, should increase with the number of epochs. It is necessary to mention that the images (in the form of a tensor of numbers) used to calculate the metrics are composed of decimal numbers and are

less than 1. For this reason, the loss function values will be low.

Figure 4.5 shows both the loss quantified by MSE and the PSNR accuracy during training to remove each of the artifacts from the images. We have decided to take 40 epoch because, as can be seen, stability in the MSE and PSNR values has already been reached by that moment. Epoch 0 corresponds to the quality indexes of the images before starting the training of the network. It is noteworthy to mention that it took around 125 minutes to train each model (note that this time depends on the machine where the models are run. All those mentioned in this chapter are taken from the same GPU we have been working with).

By observing the graphs, we can see that the models that can offer the best results are those that reduce noise and spike artifact since they are the ones that finally have a lower loss and a higher accuracy. Moreover, the second one provides a significant improvement since, at epoch 0, the value of the loss was 0.008, and it ends up being 0.001 approximately. Regarding the motion artifact, we see only a slight difference between the initial and final values, both in the case of the loss function and the PSNR. This leads us to think that the improvement produced by this model in the images affected by this artifact is not very noticeable.

Furthermore, we see that the cases of blurring and ghosting progress equally during training, although the first one starts with a higher loss. This may indicate that models trained to correct blurring and ghosting will respond similarly to images with those artifacts.

Finally, concerning the bias field artifact, there seems to be a potential improvement in the loss function (starting at about 0.010 and ending at about 0.0035). However, it is the case that results in a higher loss function and a lower accuracy, which could indicate that it is the most difficult artifact to correct by this convolutional network.

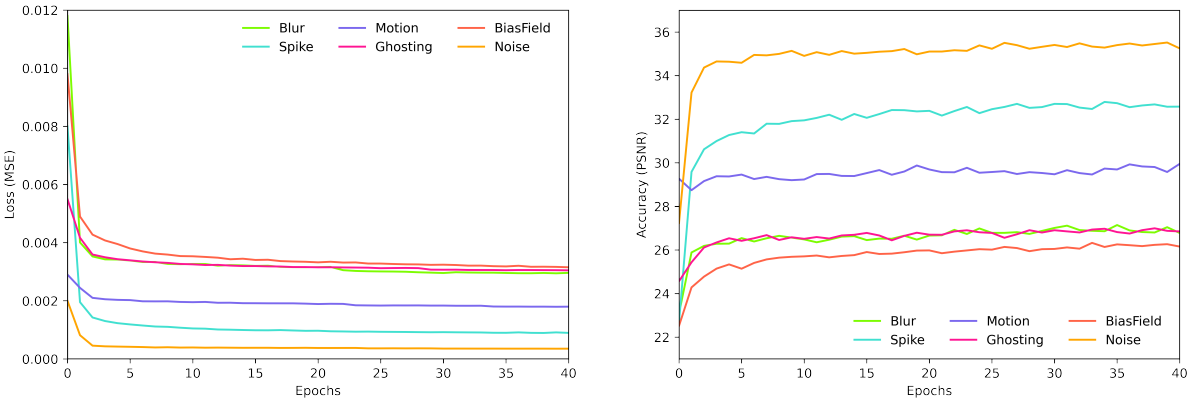


Figure 4.5: Loss (MSE) and accuracy (PSNR) during the training process of the 9-1-5 models that aim to reduce each artifact.

Nevertheless, we have to keep in mind that the index results provided by the training set are not the ones we can use to quantify the quality of the final images. They are only useful to visualize how the training process has been. To obtain information about the model performance, we have to use the test set of each artifact.

4.3.3. Results obtained with the baseline model

To analyze the model performance, we will not only use the metrics described in the previous chapter, but we will also perform a visual evaluation. As mentioned before, each of these metrics considers different aspects of the image. Therefore, possibly the numerical results do not agree when selecting the best-corrected artifact, so a visual comparison helps to determine whether the correction is adequate or not.

Note that, at present, as far as we know, there is no standard method to resolve image artifacts, as there was in the case of image resolution enhancement with interpolation techniques. For that reason, we will limit ourselves to using the results that we consider valid obtained in this subsection and compare them with modified super-resolution models.

Numerical evaluation

The following table shows the PSNR (in dB), SSIM and MS-SSIM values for the test sets of each of the MR image artifacts/defects once the 9-1-5 models have been trained:

	Blurring	**Ghostings**	**Motion art.**	**Bias field**	**Spike art.**	**Noise**
SSIM	0.9107	0.8803	0.9373	0.9241	0.8990	0.9626
PSNR	26.75	26.62	29.93	26.35	32.37	35.49
MS-SSIM	0.9337	0.8549	0.9510	0.9092	0.9715	0.9824

Table 4.1: PSNR, SSIM and MS-SSIM results for each of the artifacts/defects we are working with, with the 9-1-5 model.

Firstly, as we have already mentioned, we appreciate that not all metrics show similar results. For instance, the model for the bias field artifact has higher SSIM than the blurring one but lower PSNR and MS-SSIM. Moreover, it is remarkable the difference between the SSIM and the MS-SSIM of the spike artifact because, although both focus on aspects such as contrast and luminance, there is almost a 10% difference between them. In particular, the second index seems to indicate that the reconstructed image is practically identical to the original and not so with the SSIM (remember that the maximum value they can take is 1).

According to what we can see in the table above, the images that have the best results are those that were modified with noise. In contrast, the ones with the worst quality are those of the ghostings. Furthermore, in view of what was happening in the graphs in Figure 4.5, it seemed that the blurred images and those with ghostings would have similar performance, but only the PSNR result matches. As we can observe, it is quite difficult to make comparisons between the different artifacts due to the distinct criteria of the metrics. These will be useful in future analyses when we compare the results provided within images affected by the same artifacts. Note, however, that all the values of these metrics are within the ranges obtained by Dong et al., discussed in Section 3.2.

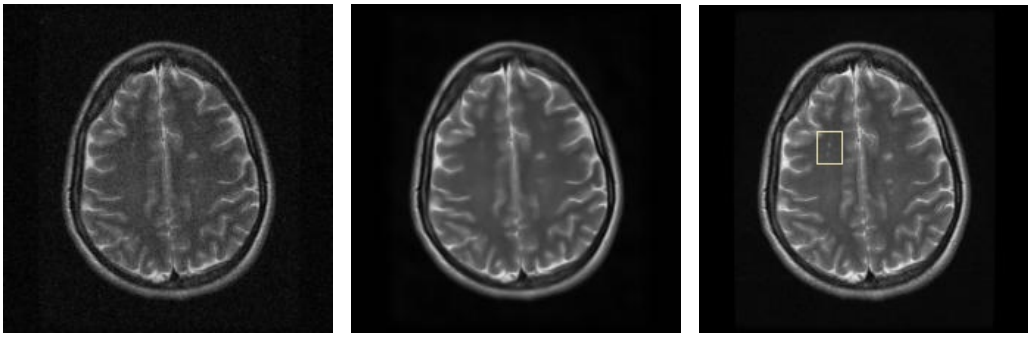
Finally, we can appreciate that the PSNR values of Table 4.1 and those of the 40th epoch of Figure 4.5 are compatible. This seems to indicate that there is no overfitting because if so, the model would have been specialized in the training set data, and when using the test data, the PSNR values would have been much worse.

Visual evaluation

The visual evaluation will allow us to determine qualitatively the artifacts that can be reasonably right corrected with the super-resolution network. Bearing in mind that the physician will use the MR image to diagnose the patient, it seems appropriate to proceed in this way to see if the corrected image is good enough to do so. Furthermore, this will also allow us to see which of the used metrics best suits what we have visually.

To do the visual evaluation, we select one image of the test set of a particular artifact, and we reconstruct it by using the model trained for that artifact. The corrected, the original, and the image with artifact are then displayed to observe the model efficiency.

Let us start by analyzing the model for images with noise, which is the one that provides the best metrics results. In Figure 4.6, it appears that the network works well to remove noise from the image, particularly in the non-brain area. Nevertheless, we see that the corrected image is slightly blurred compared to the original one. Although this defect is not very noticeable, it could be risky for the patient. For instance, in the original MR image, two whiter dots appear in the yellow framed area. They could be a sign of a serious disease such as a tumor. However, our model does not distinguish them well from the noise and tries to eliminate them. As we see in the corrected image, it is barely discernible from the dark background.



(a) With artifact (b) Corrected (c) Original

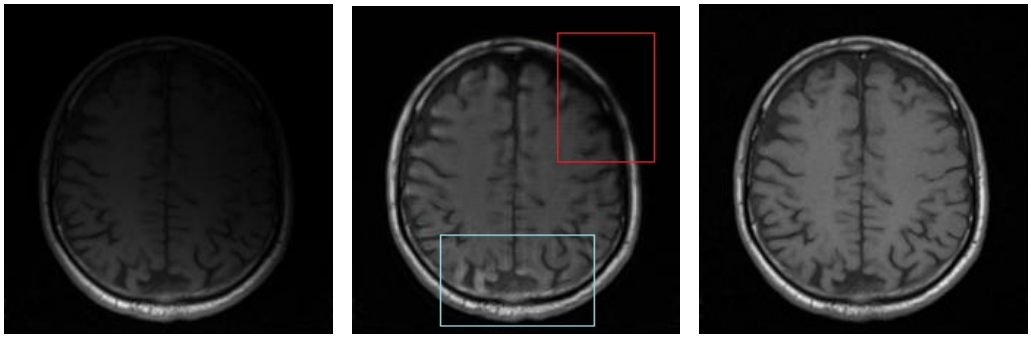
Figure 4.6: Visual comparison of images with noise.

Therefore, despite being able to eliminate noise adequately, it is necessary to take these aspects into account. In the following sections, we will see if we can solve these problems by increasing the complexity of the network.

Regarding the images affected by the bias field artifact, the objective is to visualize those areas that had been darkened. In light of the results, it seems that the model can illuminate the areas that were not seen correctly before. Nevertheless, comparing the image with the original, we see that the brain's gray color is different. This may be one of the reasons why the PSNR metric does not provide better results.

In addition, we see that the correction illuminates areas that are not that bright in the original ones. For example, in the lower part of the brain (the blue framed region), some areas appear lighter than in the original image. This can lead to confusion in the diagnosis, so it would be necessary to see if we can still improve it.

Moreover, there are still some dark areas in which the model is not capable of identifying rightly the different structures that the brain has, as appears in the red highlighted area. There is also a slight blurring compared to the original image that could interfere with the medical purposes.

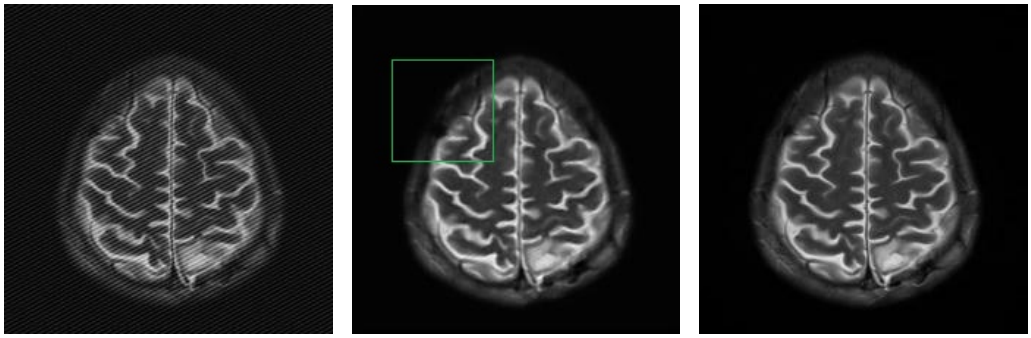


(a) With artifact (b) Corrected (c) Original

Figure 4.7: Visual comparison of images with bias field artifact.

Since the model works adequately for its purpose, and even though the values of the metrics are not high (especially PSNR), we will deal with it in the following sections to see if those defects that have appeared can be improved.

On the other hand, considering the images with spike artifacts, we see that the model can eliminate the lines that appeared. The reconstructed image is very similar to the real one, so it could be considered suitable for medical use. Thus, we can ensure that the SSIM accuracy does not match the visual appearance of images affected by stripes.



(a) With artifact (b) Corrected (c) Original

Figure 4.8: Visual comparison of image with spike artifact.

Among its shortcomings, we find the areas on the edge of the brain that have a darker shade than the real one (the green framed area) and the fact that the corrected image is slightly blurred. We will observe whether they improve with the modified networks.

However, not all artifacts can be well corrected using this type of model. In particular, we have proven its bad performance for motion artifacts and also for ghosting and blurring. They coincide with the ones we have predicted that would behave similarly due to the similar effect

they produce in MR images. Figure 4.9 shows a corrected image of each of them.

First, we see that the blur-corrected image does not appear clearly. It is quite complicated to differentiate the different parts of the brain. The only visible improvement is in the grayscale since the reconstruction seems closer to the original in that aspect. For this reason, the metrics in Table 4.1 do not show noticeable low results. In particular, we see that the PSNR is similar to the one of the bias field when the corrected image of Figure 4.7 was visibly better. As a result, we can also conclude that PSNR is not fully consistent with the visual evaluation.

Nor can ghostings be adequately corrected. As we can see, the model allows to reduce the intensity of these structures, but it is not able to eliminate them. Moreover, it reduces them better outside the area of the brain, which is the area we are less interested in from a medical point of view. In addition, the brain region that was initially unaffected by the artifact is observed to lose quality in the corrected image.

Finally, we see that the model for the motion artifact does not perform any enhancement on the affected image. We expected this since Figure 4.5 showed hardly any improvement of loss or accuracy in the 40 training epochs. This visualization shows that no advancement is achieved.

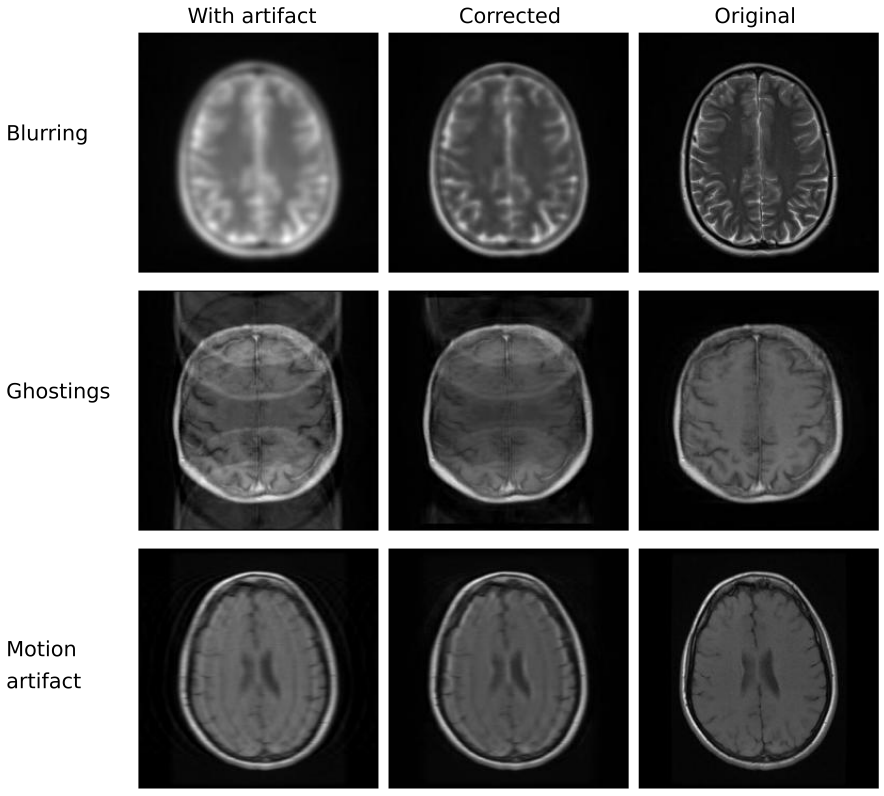


Figure 4.9: Visual comparison of images with motion artifact, blurring and ghostings.

Some alternative techniques used for this type of artifact are based on extracting the so-called residual image (the mask containing the artifact), which is then subtracted from the affected image to generate the reconstruction (see [28], where Tamada et al. obtained a highest SSIM of 0.91 with their dataset). This type of procedure is beyond the scope of our work.

4.4. Analysis of modified models' performances

Once we have analyzed what artifacts the super-resolution model is capable of significantly reducing or eliminating, we will test the model performance on the images if we modify the network depth and the number and size of filters. This procedure is similar to the one performed by Dong et al. in their article [2].

Regarding the metrics, we will use PSNR and MS-SSIM. SSIM is not considered now because we have observed that it works particularly badly for the spike case. Along this section, the values of the tables can be compared with those of Table 4.1, which belong to the model we take as baseline (they will be written in purple in the tables)[1].

4.4.1. Modifying the depth of the network

As it was already mentioned, the network depth depends on the number of layers it has. Some papers suggest that increasing it would benefit the reconstruction result (see [30]). However, it would do so by compromising the training time. We will consider models of 2 to 5 layers, where the layers that are added to the baseline model (or removed) have filters of size 1×1. For the 9-5 model there will be 64 filters, the 9-1-1-5 one will have 64 for the first layer, 32 for the second layer, and 16 filters for the third layer. Finally, the 9-1-1-1-5 model will have 64, 32, 32, and 16 filters, respectively. The last layer of each of them has only one filter. Note that this causes the number of parameters to also vary as the depth of the network changes. Now, the model will have to learn 6849 (for 9-5), 8257 (for 9-1-1-5), and 9313 (9-1-1-1-5) parameters in each case. The following table shows the results of the metrics for the three artifacts and the three modified models under consideration:

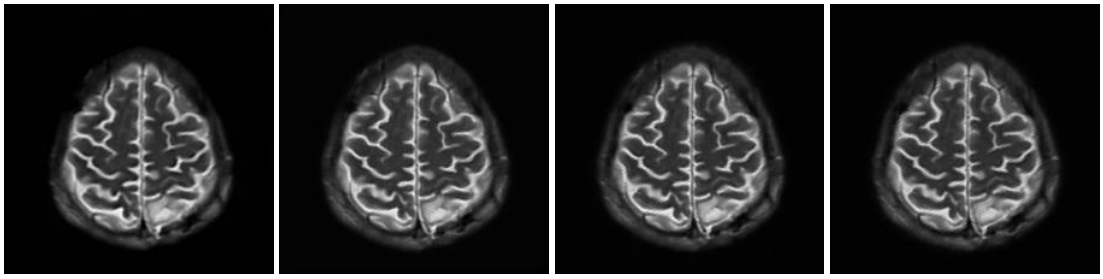
	Noise		**Bias field**		**Spike artifact**	
	PSNR	**MS-SSIM**	**PSNR**	**MS-SSIM**	**PSNR**	**MS-SSIM**
9-5	35.16	0.9815	26.34	0.9106	31.49	0.9636
Baseline	35.49	0.9824	26.35	0.9092	32.37	0.9715
9-1-1-5	35.36	0.9821	26.55	0.9116	**33.27**	0.9736
9-1-1-1-5	**35.51**	**0.9825**	**26.63**	**0.9147**	33.05	**0.9738**

Table 4.2: Results of PSNR and MS-SSIM for different model depth on noise, bias field and spike artifact datasets.

As it can be seen, an increase in the number of parameters generates an improvement in the metrics in all cases except for the PSNR of the spike artifact. Nevertheless, we see that the differences between them are not very large. Considering the number of parameters the model has to learn, the improvement does not seem really significant. In addition, the training time has increased to 130 minutes with 4 layers and 135 minutes for the 5-layer model. In the case of 2 layers, it is reduced to 116 minutes on average.

Let us see an example of the brain reconstructed from an image with a spike artifact (the same included in Figure 4.8) for each model with different number of layers. We have included it because it is the one in which we observe the largest variation in the metrics.

[1] Along the next subsections, the bold values of the tables will represent the best results of each analysis.



(a) 9-5 (b) 9-1-5 (c) 9-1-1-5 (d) 9-1-1-1-5

Figure 4.10: Visual comparison of images with spike artifact by varying the depth of the network.

As it can be seen, the image offers some improvement when distinguishing the edge of the brain. The area we highlighted before and was difficult to resolve now looks much more closely to the original image in the 9-1-1-1-5 case. With respect to the 2-layer model, the image looks more blurred, and the edge of the brain is barely distinguishable, so the decrease of parameters should not be considered in this case.

In summary, we can see improvements both in the metrics used and the visual appearance of the images if we increase the number of layers. Therefore, from the medical point of view, where the physician makes the diagnosis from direct observation of the image, it may be appropriate to consider the 5-layer model. In addition, the increase in the number of parameters is not so large that the complexity is greatly affected.

4.4.2. The effect of the filters' size on the performance

Another aspect that can vary the reconstruction quality is the size of the filters used. To prove if this is the case, let us reduce the filters' size of the first layer (5-1-5 model) and then increase the filters' size of the second layer (9-3-5 and 9-5-5 models). The number of filters for all models will be 64 for the first layer and 32 for the second layer. Now, models will have to learn 4545, 24513, and 57281 parameters, respectively.

Table 4.3 shows the results of the metrics for the different models.

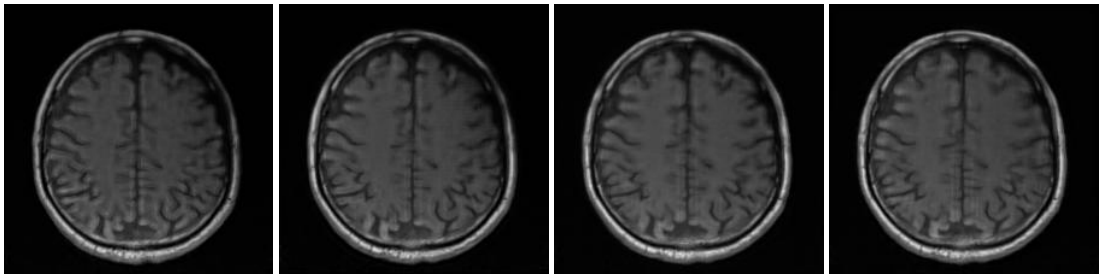
	Noise		**Bias field**		**Spike artifact**	
	PSNR	**MS-SSIM**	**PSNR**	**MS-SSIM**	**PSNR**	**MS-SSIM**
5-1-5	35.20	0.9817	25.98	0.9046	32.54	0.9696
Baseline	35.49	0.9824	26.35	0.9092	32.37	0.9715
9-3-5	35.26	0.9825	26.59	0.9210	33.02	0.9734
9-5-5	**35.66**	**0.9833**	**27.10**	**0.9219**	**33.13**	**0.9736**

Table 4.3: Results of PSNR and MS-SSIM for different filters' size on noise, bias field and spike artifact datasets.

We see that, as before, better results are obtained if the model is more complex. This seems to indicate that the model performance is better if information about neighboring pixels is taken into account in both the input patch extraction process and the mapping process (remember that the filter size indicates the number of pixels considered in each step). In particular, there is a difference of more than a dB in the PSNR and 0.01 in the MS-SSIM in the results of the 9-5-5 model with the bias artifact. For the rest, the performance improvement is not very noticeable.

Regarding training times, they have now become, from simpler to more complex, 120, 138, and 160 minutes. Note that the time difference is now greater than it was in the previous subsection, which clearly indicates the relationship between the number of parameters to be learned and the time it takes to train.

Before concluding this subsection, let us show the brain in Figure 4.11 reconstructed using the models currently under consideration. Remember that the problems were the darker areas that it was not able to lighten and sharpen and those that were lightened compared to the original image.

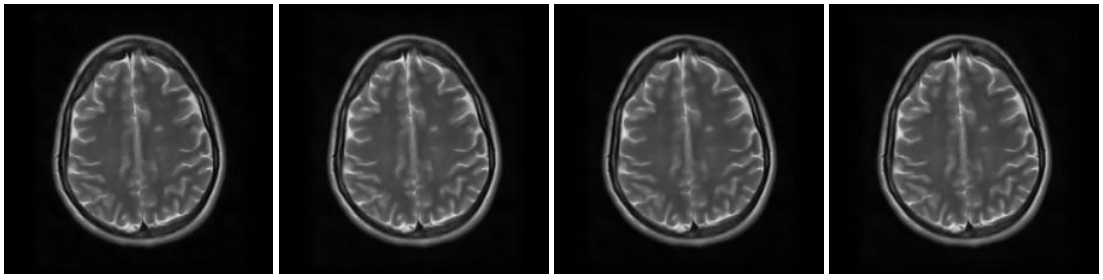


(a) 5-1-5 (b) 9-1-5 (c) 9-3-5 (d) 9-5-5

Figure 4.11: Visual comparison of images with bias field artifact by varying the filters' size.

We can observe that none of the reconstructions manages to homogenize the areas that the base model clarified. Moreover, it we can see that increasing the complexity makes the structure of the darker area on the right side of the brain more visible, especially the one of the 9-3-5 model.

Let us see also the images were affected by noise since it is in the 9-5-5 model where the best result of the whole chapter is obtained.



(a) 5-1-5 (b) 9-1-5 (c) 9-3-5 (d) 9-5-5

Figure 4.12: Visual comparison of images with noise by varying the filters' size.

Visually, the images are very similar to each other. Hardly any changes are seen in the different models, and even when the model acquires the highest complexity, it is not able to distinguish the two white dots appearing in the center of the brain mentioned above. In addition, the slight blurring we detected before can not be improved either compared to the original image. However, the image contrast seems to slightly improve.

In view of the results, it seems that the images have better metrics results as the complexity increases. However, for example, in the 9-5-5 model, the number of parameters is about 7 times

more than the baseline, which leads to a longer training time. Moreover, by looking at the images where most of the differences between indexes appear, we see that the problems we had with the original are not completely solved, but they are improved. Selecting one or another model will be a trade-off between performance and training time.

4.4.3. The effect of the number of filters on the performance

To finish this section, let us explore what happens if we vary the number of filters in the analysis. Again, this is closely related to the complexity of the model. As we appreciated before that increasing the complexity of the model resulted in improvements, but that they were not very visible in the image, we will see two cases of reducing the number of filters and one example of enlarging it. To do this, we will use filters of 9×9, 1×1, and 5×5 for each of the layers, respectively. However, instead of using 64 filters for the first layer and 32 for the second layer as in the baseline model, we will apply 16 and 8 (16-8 model), 32 and 16 (32-26 model), and 128 and 64 filters (128-64 model), respectively. Thus, the number of parameters becomes 1649, 3552, and 20353 in each case.

Table 4.4 represents the values of the metrics obtained of the described models:

	Noise		**Bias field**		**Spike artifact**	
	PSNR	**MS-SSIM**	**PSNR**	**MS-SSIM**	**PSNR**	**MS-SSIM**
16-8	35.03	0.9801	25.82	0.9038	31.60	0.9650
32-16	35.32	0.9813	25.92	0.9032	32.31	0.9696
Baseline	**35.49**	0.9824	26.35	0.9092	32.37	0.9715
128-64	35.47	**0.9831**	**26.54**	**0.9144**	**33.39**	**0.9746**

Table 4.4: Results of PSNR and MS-SSIM for different number of filters on noise, bias field and spike artifact datasets.

This is the latest evidence that model complexity involves an improvement in the metrics. Although in the case of noise, the enhancement is hardly noticeable (for example, PSNR increases approximately 0.4 from the 16-8 to the 128-64 model), in the case of the spike artifact, the improvement is much more evident (the increase is almost 1.8). However, it is worth noting that the metric values do not seem to get much worse when considering a model that only learns about 1600 parameters. This shows that a very simple model can provide good results for the images under consideration. The training times are now approximately 100, 106, and 142 minutes, ordered from the lowest to the highest number of filters.

In this case, we will not make any qualitative comparison because we have already seen the possible differences that can occur with the three artifacts when making different adjustments to the model. Analogously to what happened before, the decision to opt for one or another model will depend on the time it takes to train and the result (and precision) we are looking for.

4.5. Further baseline model analysis

Besides making adjustments to the model, new tests can be performed to justify how to proceed. In particular, we will see what happens if, instead of using 20,000 images in the training step, we take fewer. We will also evidence why each model has been trained separately for the artifacts. Finally, we will test the results when using knee MR images with artifacts in the models trained with brain MR images.

4.5.1. Changing the size of the training set

When we are training computational models, the machine we work with plays an important role. Depending on the computer power, the time it takes to train can vary enormously. During the development of this work, to provide an example, there were problems with the GPU we were using, and when we tried to continue with our personal computer, the same models took around 40 hours to train.

The solution we found during that time was to reduce the size of the analysis dataset, to reduce the running time. The following table presents the results of the metrics when training the model of Section 4.3, but with 2000 images and the comparison with the original model.

	Noise		**Bias field**		**Spike artifact**	
	PSNR	**MS-SSIM**	**PSNR**	**MS-SSIM**	**PSNR**	**MS-SSIM**
2000 images	34.51	0.9810	25.30	0.9088	31.38	0.9663
Baseline	**35.49**	**0.9824**	**26.35**	**0.9092**	**32.37**	**0.9715**

Table 4.5: Results of PSNR and MS-SSIM for different training set size on noise, bias field and spike artifact datasets.

We can observe that the results are worse in all cases when we train with 2000 images. This is reasonable since the model has not seen so many different dispositions of artifacts (remember that these were generated randomly) to know how to correct all the artifacts that later appear in the test data.

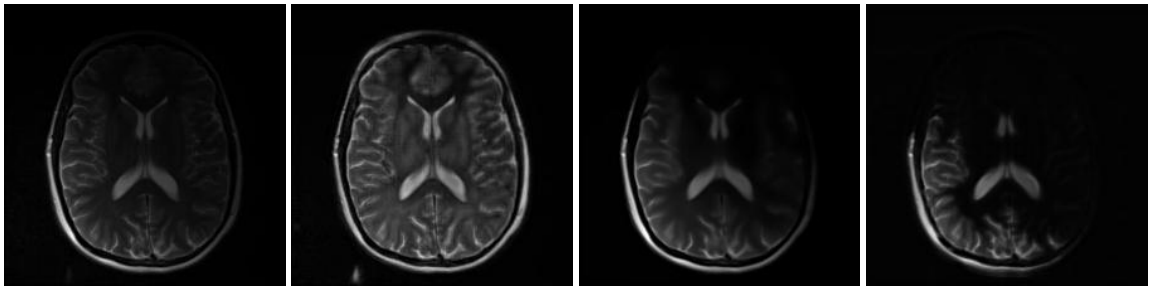
Nevertheless, if we are dealing with an underpowered processor, it is not unreasonable to consider working with a reduced dataset to achieve relevant results. Note that, for example, in the noise case, we started from a PSNR of about 27, and we managed to go up to 34.51. In particular, with 2000 images, we were able to reduce the model running time on our computer to about 3.5 hours.

4.5.2. About the specificity of the models to a particular artifact

Furthermore, we could wonder why we have trained the same models for each artifact, doing 6 times the same process for each of the experiments performed. This subsection shows clear evidence of why it is not enough to train a single model to correct the different image defects. For this purpose, in Figure 4.13 we present an image affected by bias field artifact as the input of the baseline model trained with bias field and spike artifact, and also with noise.

Given the results in Figure 4.13 we appreciate that it is clear that the only model capable of making the brain fully visible is the one trained with images with bias field artifact. In fact, if another model is used, we can see that the reconstruction obtained could have even worse quality than the input, as in this case.

Using images with all artifacts to train a model would not give us a good result either. If we do it, we would be trying to make the model learn from a wider variety of reconstruction possibilities, making it even less specific. This would worsen the performance of the model.

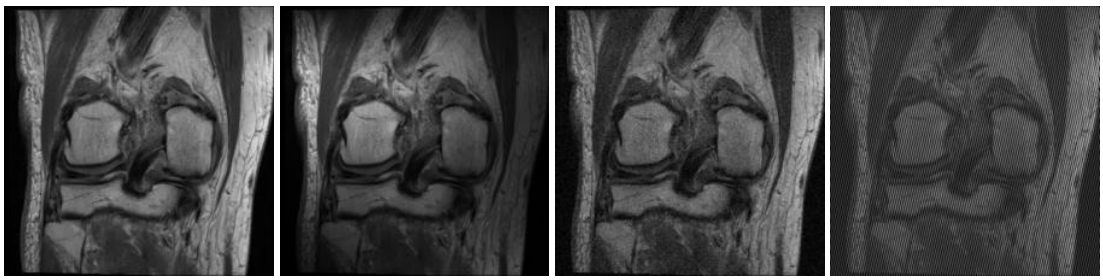


(a) With bias field artifact (b) Bias field artifact model's output (c) Noise model's output (d) Spike artifact model's output

Figure 4.13: Visual comparison of images with spike artifact by varying the depth of the network.

4.5.3. About the generalization to other MR images

Finally, we may wonder if the fact of training models with brain MR images could mean that they can only be used to reconstruct this type of image. Note that the models we have trained have learned to eliminate a given artifact by comparing the real image with the low-quality image. Therefore, we would expect that they can improve any image affected by that particular artifact.



(a) Original image (b) With bias field artifact (c) With noise (d) With spike artifact

(e) Bias field artifact model's output (f) Noise model's output (g) Spike artifact model's output

Figure 4.14: Visual comparison of knee MR images corrected with baseline models.

In Figure 4.14 we present an MR image of a knee affected by spike and bias field artifact and also by noise. The process of originating them has been the same we used for brain images. The real image belongs to a dataset of knee MR images from NYU Langone Health, which was also originally in DICOM format. The corrections made with the baseline model trained with brain images are shown too.

Note that we do not provide the values of the metrics. This is because these images do not come from the same dataset from which we obtained the values in the previous sections. Therefore, the results obtained here would not be comparable to the previous ones.

Given the results of Figure 4.14, we observe that the models also work to correct the knee images. We can see how the spike artifact model can remove the stripes from the image (d), just as with brain images, even though the resulting image has less contrast than the original one. Noise, on the other hand, can also be removed. However, we observe that the reconstructed image has some blurring, but it does not impede the visualization of the different structures of the knee.

The model that corrects for the bias field artifact provides the worst reconstruction quality. As we can appreciate in image (e), there are structures in the central area that acquire a lighter shade than the original image, while others, for instance, on the right side, remain darker. It could be a source of wrong diagnosis, since it can be mistaken with anomalous tissues. However, this fact is reasonable considering that the brain only occupied the most central area of the MR image while the rest was black, so a bias field artifact affecting the latter area would be neither detected nor corrected with the same certainty as the model does with the central area. Therefore, we consider that, for the last artifact, we have to look for alternatives to improve the results. The first one is using the dataset of the same class as the image to be reconstructed to train the model. On the other hand, a set of methods called transfer learning could be used. It is based on using the network trained on certain images with several frozen layers (layers whose weights are not updated), and then only the final layers are trained with few images of the new domain.

Chapter 5

Conclusions and future work

To conclude the study, we will review the key points analyzed in it. First of all, in Chapter 2, we have given an extensive description of the phenomenon of magnetic resonance and image formation through k-space. As a result, it has been possible to understand the emergence of artifacts that worsen the quality of MR images from a physical point of view.

Moreover, in Chapter 3, we have studied the basic fundaments of Deep Learning and, in particular, of the network and Python library we have subsequently used. It has allowed us to learn theoretically about the tool used to test the removal of artifacts from MR images.

Once we analyzed theoretically all the aspects related to our work, we have proceeded with the development of a code in Python language (not previously used in the degree) that included the super-resolution convolutional network designed by Dong et al.. With it, we have been able to achieve the main objective of our work, which was to study whether this network specialized in enhancing image resolution could be useful to reduce or remove the artifacts generated in magnetic resonance imaging. In addition, we have been able to illustrate some of the aspects studied, such as images k-spaces (Section 4.2) and feature maps (Section 4.3), using our developed program.

Firstly, we have seen that ghosting and motion artifacts and also blurring are not correctable with this type of network, while its performance is adequate for images with noise and also for those affected with bias field and spike artifacts. The PSNR values obtained for the latter were 35.49, 26.35, and 32.37 dB, respectively, while the MS-SSIM ones were 0.9824, 0.9715, and 0.9092 (see Section 4.3). We have shown that it is better to appreciate this fact by visual evaluation rather than through metrics, as it has been proved that the different aspects they quantify and the diversity of the evaluated datasets make them inconsistent with each other.

Regarding the images whose defects the model can not improve, it is thought that this may be because they are too strong (or defined). Something similar happened in the Multimodal Brain Tumor Segmentation challenge of 2014 (BraTS) [31], where problems with the datasets led to poor models results. It is suggested for further work to use datasets with softer artifacts of motion and ghosting and also blurring to see if the model performance improves and if it is effective in those cases.

Then, the performance of modified models compared to the one of the baseline model was studied to see if the complexity of the network affects the quality of the reconstructions obtained.

We have observed that in all cases in which the baseline model works, the metrics improve if the number of layers of the model is increased, as well as the size and number of filters. In particular, the spike artifact reconstructions give the best PSNR and MS-SSIM results (33.39 and 0.9749, respectively) using the 128-64 model, i.e., increasing the network filters. Regarding the bias field artifact and noise, the best model to correct them is the one with the largest filters (the 9-5-5 one). It gives PSNR and MS-SSIM results of 35.66 and 0.9833 and 27.10 and 0.9219, respectively. We have also seen that the image's visual quality also improves, although the initially visible defects are not completely resolved (Section 4.4).

Finally, in Section 4.5, we have observed that the number of images trained also affects the networks' performance, being better the larger the dataset is. Moreover, it was evidenced the importance of using specialized models for a particular type of artifact and the fact that other images, such as knee MR images, can also be corrected for noise and spike artifact with networks despite being trained in brain datasets. We propose as future work to use transfer learning, described in Subsection 4.5.3, as a technique to improve these results, as well as the one of the bias field artifact, whose quality was worse with knee images. It might be interesting to compare them with the performance of a network trained directly on knee images.

To sum up, we have observed that super-resolution convolution neural networks are simple models that can effectively improve the quality of images affected by spike and bias field artifacts and noise. They provide relevant results, which could have an impact not only in the area of medical imaging but also in other fields where image quality is affected by artifacts or image defects. For example, in astronomy, where images are affected by noise, it is necessary to perform a post-production step, where this defect is tried to be removed. Currently, studies in this aspect are focused on Deep Learning, using more complex networks than the one studied here (see [32]). It is suggested as a new line of research to use this network for astronomical images, making the necessary adjustments in layers and filters to see if significant denoising results can be obtained.

Bibliography

[1] B. Balcom, P. Cano Barrita, C. Choi, S. Beyea, D. Goodyear, and T. Bremner. "Single-point Magnetic Resonance Imaging (MRI) of cement based materials". In: *Materials and Structures* 36 (2003), pp. 166–182.

[2] C. Dong, C. C. Loy, K. He, and X. Tang. "Image Super-Resolution Using Deep Convolutional Networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38 (2014).

[3] K. Umehara, J. Ota, N. Ishimaru, S. Ohno, K. Okamoto, T. Suzuki, N. Shirai, and T. Ishida. "Super-resolution convolutional neural network for the improvement of the image quality of magnified images in chest radiographs". In: *Proceedings of SPIE - The International Society for Optical Engineering* 10133 (2017), 101331P1–101331P7.

[4] H. Liu, J. Xu, Y. Wu, Q. Guo, B. Ibragimov, and L. Xing. "Learning Deconvolutional Deep Neural Network for High Resolution Medical Image Reconstruction". In: *Information Sciences* 468 (2018), pp. 142–154.

[5] Y. Gao, H. Li, J. Dong, and G. Feng. "A deep convolutional network for medical image super-resolution". In: *2017 Chinese Automation Congress (CAC)* (2017), pp. 5310–5315.

[6] W. Shuaifang, W. Wu, G. Jeon, A. Ahmad, and X. Yang. "Improving resolution of medical images with deep dense convolutional neural network". In: *Concurrency and Computation: Practice and Experience* 32 (2018), e5084.

[7] M. Mitra. "Electron Magnetic Resonance in Nanoparticles and its Characteristics". In: *SOAOJ publishing* 1 (2019), pp. 1–6.

[8] R. Edelman. "The History of MR Imaging as Seen through the Pages of Radiology". In: *Radiology* 273 (2014), S181–200.

[9] M. A. Brown, Richard C. S., and B. M. Dale. *MRI: Basic Principles and Applications.* Wiley-Blackwell, (2010).

[10] D. Mcrobbie, E.A. Moore, M. Graves, and M.R. Prince. *MRI from picture to proton.* Cambridge University Press, (2006).

[11] R. K. Hobbie. *Intermediate Physics for Medicine and Biology.* Vol. 457. (1999), S516–538.

[12] D. Weishaupt, V. Köchli, and B. Marincek. *How Does MRI Work?: An Introduction to the Physics and Function of Magnetic Resonance Imaging.* Springer, (2006).

[13] J. Lafuente Martínez and L. Oleaga Zufiría. *Monografía SERAM: Aprendiendo los fundamentos de la resonancia magnética.* Editorial Médica Panamericana, S. A., (2007).

[14] C. Philpott and P. Brotchie. "Comparison of MRI sequences for evaluation of multiple sclerosis of the cervical spinal cord at 3 T". In: *European journal of radiology* 80 (2010), pp. 780–785.

[15] M.Y.M. Chen, T. L. Pope, and D. J. Ott. *Basic Radiology.* McGraw Hill / Medical, (2010).

[16] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel. "Backpropagation applied to handwritten zip code recognition". In: *Neural Computation* 1 (1989), pp. 541–551.

[17] L. Iglesias, P. Bellón, A. Barrio, P. Fernández-Miranda, D. Rodriguez, J. Vega, A. Mandly, and J. Blanco. "A primer on deep learning and convolutional neural networks for clinicians". In: *Insights into Imaging* 12 (2021). 117.

[18] S. Lathuilière, P. Mesejo, X. Alameda-Pineda, and R. Horaud. "A Comprehensive Analysis of Deep Regression". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42 (2018). 9.

[19] P. S. Fisher and A. M. Eskicioglu. "Image quality measures and their performance". In: *IEEE Transactions on Communications* 43 (1995), 2959–2965.

[20] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. "Image Quality Assessment: From Error Visibility to Structural Similarity". In: *IEEE Transactions on Image Processing* 13 (2004), pp. 600 –612.

[21] Z. Wang, E. Simoncelli, and A. Bovik. "Multiscale structural similarity for image quality assessment". In: *Conference Record of the Asilomar Conference on Signals, Systems and Computers* 2 (2003), pp. 1398 –1402.

[22] M. Krishna, M. Neelima, H. Mane, and V. Matcha. "Image classification using Deep learning". In: *International Journal of Engineering Technology* 7 (2018), pp. 614–617.

[23] E. Dogo, O. Afolabi, N. Nwulu, B. Twala, and C. Aigbavboa. "A Comparative Analysis of Gradient Descent-Based Optimization Algorithms on Convolutional Neural Networks". In: *2018 International Conference on CTEMS* (2018).

[24] D. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization". In: *International Conference on Learning Representations* (2014).

[25] Y. Dauphin, H. Vries, J. Chung, and Y. Bengio. "RMSProp and equilibrated adaptive learning rates for non-convex optimization". In: *arXiv: Learning* 35 (2015).

[26] Y. Liu, Y. Gao, and W. Yin. "An Improved Analysis of Stochastic Gradient Descent with Momentum". In: *NeurIPS 2020* (2020). 1533.

[27] F. Pérez-García, R. Sparks, and S. Ourselin. "TorchIO: a Python library for efficient loading, preprocessing, augmentation and patch-based sampling of medical images in deep learning". In: *Computer Methods and Programs in Biomedicine* 208 (2021). 106236.

[28] R. Shaw, C. Sudre, S. Ourselin, and M. J. Cardoso. "MRI k-Space Motion Artefact Augmentation: Model Robustness and Task-Specific Uncertainty". In: *Proceedings of Machine Learning Research* 102 (2019), pp. 427–436.

[29] NYU Langone Health. FastMRI Dataset. In: (). URL: `https://fastmri.med.nyu.edu/`.

[30] K. He and J. Sun. "Convolutional neural networks at constrained time cost". In: *IEEE Conference on Computer Vision and Pattern Recognition* (2015).

[31] M. Havaei, H. Larochelle, P. Poulin, and P. M. Jodoin. "Within-Brain Classification for Brain Tumor Segmentation". In: *International Journal of Computer Assisted Radiology and Surgery* 11 (2015), 777–788.

[32] A. Vojtekova, M. Lieu, I. Valtchanov, B. Altieri, L. Old, Q. Chen, and F. Hroch. "Learning to Denoise Astronomical Images with U-nets". In: *Monthly Notices of the Royal Astronomical Society* 503 (2020), 3204–3215.

Appendix A

Equations of movement of the net magnetization vector

As stated in Chapter 2, the net magnetization vector is non-zero when protons are immersed in a magnetic field, e.g., $\mathbf{B}_0 = (0, 0, B_0)$ pointing along the z-direction, and points parallel to it. If another field is applied, then its direction changes. The next sections express the movement equations of this vector both when an RF pulse is applied and once the application has stopped [11].

A.1. Magnetization vector when a 90°RF pulse is applied: nutation movement

If a 90°RF pulse is applied, the net magnetization vector $\mathbf{M}_0$ rotates following a nutation movement until it lies on the xy plane. For that reason, it is common to use a rotating coordinate system. Therefore, we will express the movement equations in this coordinate system, as well as in the laboratory reference frame. To do so, we will make use of an important result of quantum mechanics, which states that $\mathbf{M}_0$ changes with time as follows

$$\frac{d\mathbf{M}_0}{dt} = \gamma(\mathbf{M}_0 \times \mathbf{B}), \tag{A.1}$$

where $\mathbf{B}$ is the net magnetic field affecting the protons, which can be time dependent or independent.

A.1.1. Rotating reference frame

First of all, we will show the new coordinates in the rotating reference frame, x', y' and z'. The simplest option is to consider that x and y axes rotate around the z axis with the Larmor frequency $\omega_0 = \gamma B_0$ (Eq. (2.1)) (see Figure A.1). Then, if we have $\mathbf{M}_0 = (M_x, M_y, M_z)$ in the Cartesian coordinate system, we can express it in the rotating coordinate system $\mathbf{M}_{0'} = (M_{x'}, M_{y'}, M_{z'})$ as

$$\begin{aligned} M_x &= M_{x'}\cos(-\omega_0 t) - M_{y'}\sin(-\omega_0 t), \\ M_y &= M_{x'}\sin(-\omega_0 t) + M_{y'}\cos(-\omega_0 t), \\ M_z &= M_{z'}. \end{aligned} \tag{A.2}$$

Note that the quantity inside the sines and cosines is negative, which means that the x'y' plane rotates clockwise.

Let us suppose that we add a new magnetic field, $B_1\cos(\omega_0 t)$ along the x direction, which is the effect of applying a RF pulse. Now, $\mathbf{B} = (B_1\cos(\omega_0 t), 0, B_0)$ and $\mathbf{M}_0 \times \mathbf{B} = (M_y B_0, M_z B_1 \cos(\omega_0 t) - M_x B_0, -M_y B_1 \cos(\omega_0 t))$. Then, using Equation (A.1) and derivating relations (A.2) we get

$$\begin{aligned}
\frac{dM_x}{dt} &= \frac{dM_{x'}}{dt}\cos(-\omega_0 t) - \frac{dM_{y'}}{dt}\sin(-\omega_0 t) + \omega_0 M_{x'}\sin(-\omega_0 t) + \omega_0 M_{y'}\cos(-\omega_0 t) \\
&= \gamma B_0[M_{x'}\sin(-\omega_0 t) + M_{y'}\cos(-\omega_0 t)], \\
\frac{dM_y}{dt} &= \frac{dM_{x'}}{dt}\sin(-\omega_0 t) + \frac{dM_{y'}}{dt}\cos(-\omega_0 t) - \omega_0 M_{x'}\cos(-\omega_0 t) + \omega_0 M_{y'}\sin(-\omega_0 t) \\
&= \gamma B_1 \cos(-\omega_0 t) M_{z'} - \gamma B_0[M_{x'}\cos(-\omega_0 t) - M_{y'}\sin(-\omega_0 t)], \\
\frac{dM_z}{dt} &= \frac{dM_{z'}}{dt} = -\gamma B_1 \cos(\omega_0 t)[M_{x'}\sin(-\omega_0 t) + M_{y'}\cos(-\omega_0 t)].
\end{aligned}$$

As we have $\omega_0 = \gamma B_0$, then the two first expressions can be simplified, such that

$$\begin{aligned}
\frac{dM_{x'}}{dt}\cos(-\omega_0 t) - \frac{dM_{y'}}{dt}\sin(-\omega_0 t) &= 0, \\
\frac{dM_{x'}}{dt}\sin(-\omega_0 t) + \frac{dM_{y'}}{dt}\cos(-\omega_0 t) &= \gamma B_1 \cos(-\omega_0 t) M_{z'}.
\end{aligned}$$

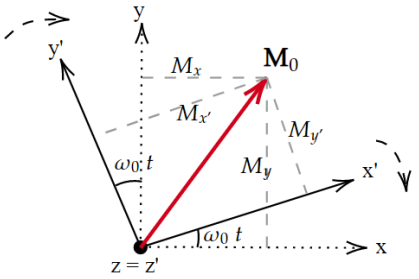


Figure A.1: Magnetization vector M_0 and its components projected in the xy and x'y' planes.

By using these equations, we can now express $dM_{x'}/dt$ and $dM_{y'}/dt$ independently, which results in

$$\begin{aligned}
\frac{dM_{y'}}{dt} &= \gamma B_1 M_{z'}\cos(\omega_0 t)\cos(-\omega_0 t) = \gamma B_1 M_{z'}\cos^2(-\omega_0 t), \\
\frac{dM_{x'}}{dt} &= \gamma B_1 M_{z'}\cos(\omega_0 t)\sin(-\omega_0 t) = \gamma B_1 M_{z'}\cos(-\omega_0 t)\sin(-\omega_0 t).
\end{aligned}$$

As they depend on time, we can average the three derivatives, which results in simpler expressions. As the average of $\cos^2(-\omega_0 t)$ is $1/2$, and the one of $\cos(-\omega_0 t)\sin(-\omega_0 t)$ is 0, then

$$\frac{dM_{x'}}{dt} = 0, \qquad \frac{dM_{y'}}{dt} = \omega_1 M_{z'}, \qquad \frac{dM_{z'}}{dt} = -\omega_1 M_{y'},$$

where $\omega_1 = \gamma B_1/2$.
We can now differentiate the second and third equations, such that

$$\begin{aligned}
\frac{d^2 M_{y'}}{dt^2} &= \omega_1 \frac{dM_{z'}}{dt} = -\omega_1^2 M_{y'}, \\
\frac{d^2 M_{z'}}{dt^2} &= -\omega_1 \frac{dM_{y'}}{dt} = -\omega_1^2 M_{z'}.
\end{aligned}$$

By solving the ODEs, and knowing that $\mathbf{M}_0$ was initially pointing along the z direction, we get

$$\begin{aligned}
M_{x'} &= 0, \\
M_{y'} &= M_0 \sin(\omega_1 t), \\
M_{z'} &= M_0 \cos(\omega_1 t).
\end{aligned} \tag{A.3}$$

A.1.2. Laboratory reference frame

Once we have the expressions of the net magnetization vector's coordinates in the rotating reference frame, we can express them in the laboratory reference frame using equations (A.2). Then,

$$\begin{aligned} M_x &= -M_0 \sin(\omega_1 t)\sin(-\omega_0 t), \\ M_y &= M_0 \sin(\omega_1 t)\cos(-\omega_0 t), \\ M_z &= M_0 \cos(\omega_1 t). \end{aligned} \tag{A.4}$$

These expressions are more complex than the previous ones, which justify using the rotating coordinates.

A.2. Magnetization vector once the relaxation process has started

We can write analytically the equations that represent the projections of the magnetization vector $\mathbf{M}_0 = (M_x, M_y, M_z)$ in the Cartesian axes during the relaxation process. To do so, we need to remember both the definitions of T1 and T2 times and the fact that γ is the gyromagnetic ratio. Then, using the so-called Bloch equations,

$$\begin{aligned} \frac{dM_x}{dt} &= -\frac{M_x}{T_2} + \gamma(\mathbf{M}_0 \times \mathbf{B}_0)_x, \\ \frac{dM_y}{dt} &= -\frac{M_y}{T_2} + \gamma(\mathbf{M}_0 \times \mathbf{B}_0)_y, \\ \frac{dM_z}{dt} &= \frac{1}{T_1}(M_0 - M_z) + \gamma(\mathbf{M}_0 \times \mathbf{B}_0)_z, \end{aligned}$$

where $\mathbf{M}_0 \times \mathbf{B}_0 = (M_y B_0, -M_x B_0, 0)$, as we are considering the external magnetic field $\mathbf{B}_0$ pointing along the z axis' direction. Therefore, by substituting the result of the vectorial product, and solving the ODEs, we get:

$$\begin{aligned} M_x &= M_0 e^{-t/T_2} \cos(-\omega_0 t), \\ M_y &= M_0 e^{-t/T_2} \sin(-\omega_0 t), \\ M_z &= M_0 (1 - e^{-t/T_1}). \end{aligned} \tag{A.5}$$

Note that when $t = 0$, $M_z = 0$, which makes sense, as we are considering we have applied a 90°pulse and then, initially, there is no longitudinal component of the magnetization vector. In addition, we can observe that the amplitude of $\mathbf{M}_0$ decreases in the transverse components as time passes.

Appendix B

Diagram of the developed program

The program we have developed for this work performs the entire preprocessing process, including the conversion from DICOM to jpg format and artifacts generation. It also includes the division of the complete dataset into training, validation, and test set to perform the training.

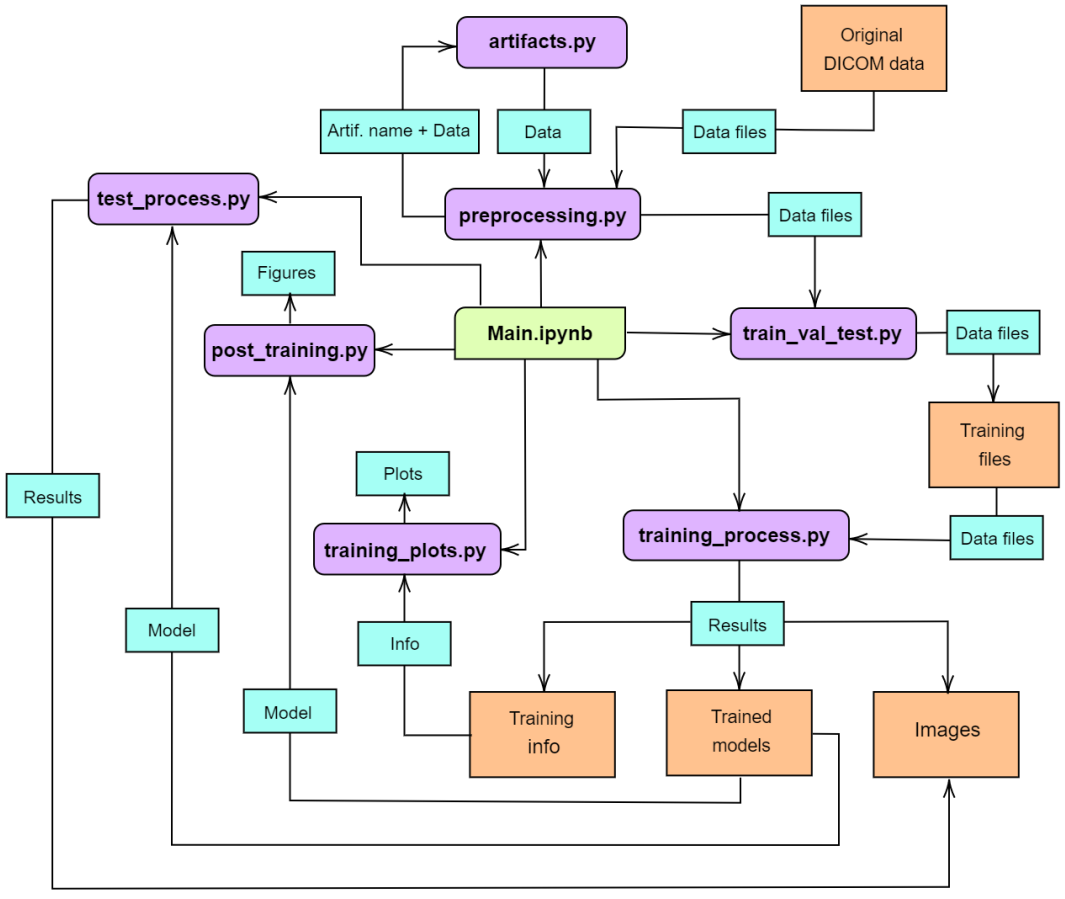


Figure B.1: Diagram of the developed program including the preprocessing steps, as well as the model training and the generation of results figures.

Moreover, it contains the architecture of super-resolution convolutional neural networks to create and train the models from within the program. These models, as well as some images of the process and the metrics at each epoch, are saved. The graphs corresponding to the metrics

results can be generated by the program. We can also observe the performance of these models on the test sets and visualize both the filters and the feature maps of the first layer of each model.

Figure B.1 represents a simplified diagram of the program structure. As can be seen, in addition to the .py modules, there is a Jupyter Notebook called Main.ipynb. This has been created to be able to run the program from it and to perform each step visually. It is properly commented to understand what is done throughout the process. The modules are as follows:

- *preprocessing.py*: It contains functions that allow you to read data from the original folder and also transform it from DICOM to jpg format. In this module, artifacts.py is called by providing the name of the artifact to be generated and the original image. It allows to display and save the original images as well as the modified ones.
- *artifacts.py*: The function it contains first creates the mask of the required artifact using the TorchIO library and then applies it to the input image. It returns the modified image in tensor format.
- *train_val_test.py* It contains functions that allow the division of datasets into training, validation, and test sets and also organizes the folders in a way that makes it easier to know where each image is. It is important to mention that this division is made at the study level and not at the image level to ensure that the same brain is not used for training and testing. In addition, the same division is used for all artifacts so that analyses such as the one in Subsection 4.5.2 can be performed without the test image having been used for training in any of the cases.
- *training_process.py*: It contains the class of the models to be trained. In addition, it performs the whole process of preparing the data to be used for training and also has the training function. It stores both images of the process, as well as the trained model and the resulting metrics at each epoch.
- *training_plots.py*: It has a function to plot the training results saved by the training_process.py module. The result can be displayed and also saved if desired. It was used, for instance, to obtain Figure 4.5.
- *post_training.py*: It contains two functions, one to obtain the filter representations of the first layer of a given model as input and another to represent the feature maps of the same layer for a given image as input. The result can be displayed and also saved if desired. The first one was used to obtain Figure 4.3, and the second to generate Figure 4.4.
- *test_process.py*: It is the module used to evaluate the performance of a model that has already been trained. The trained model is provided as input and returns the metrics' values for the test set of the corresponding artifact. This is what gives us the values of the tables of sections 4.3, 4.4 and 4.5. Moreover, it saves some of these corrected images, which are shown for example in figures 4.6, 4.7 and 4.8.

The code is available at `https://github.com/MariaPFdez/ArtifactsRemoval`