Super-Resolution Enhancement of Magnetic Resonance Images using Deep Learning

Theodor Baur
MSc Artificial Intelligence

School of Computer Science and Informatics
Cardiff University

Supervisor: Frank C. Langbein

August 30, 2025

Abstract

High-quality brain images in magnetic resonance imaging (MRI) are essential for accurate diagnostic tasks such as tumour segmentation, with 3T images having the potential for higher quality than 1.5T. However, many parts of the world only have access to 1.5T machines, and sometimes improperly scanned images contain acquisition artefacts. Deep-learning based super-resolution can enhance 1.5T images to resemble 3T-like quality, aiming to match the diagnostic benefits of higher-quality images without access to a 3T machine. However, most existing work focuses primarily on general MRI image enhancement and does not specifically target 1.5T to 3T conversion or artefact reduction in their approaches. This project aims to target these gaps by degrading a set of 3T images to resemble 1.5T using k-space filtering and simulate GRAPPA and Partial Fourier acquisition artefacts, then implementing and evaluating ESRGAN-based super-resolution models with different loss functions. We found that a composite weighted loss function combining VGG19 perceptual loss, L1 pixel loss, and Sobel edge loss achieved the best performance (SSIM = 0.961, PSNR = 39.096, LPIPS = 0.032), while replacing edge loss with Fourier loss yielded comparable results (SSIM = 0.959, PSNR = 38.599, LPIPS = 0.034). To assess downstream utility, we processed the enhanced images from both models and non-enhanced low quality images using FSL-FAST (for grey matter, white matter, and cerebrospinal fluid segmentation) and DeepSeg (for tumour segmentation). In both cases, the enhanced images exhibited performance more closely aligned with high-quality images than with low-quality counterparts, suggesting that the super-resolution provides images that contain clinically useful information.

Acknowledgments. I would like to thank Frank C Langbein for supervising this project, providing guidance, answering my questions, and offering feedback that significantly improved the quality of this work.

Contents

Chapter 1

Introduction

1.1 Motivation

In clinical practice, MRI scanners are typically 1.5 Tesla (1.5T) or 3 Tesla (3T), with 1.5T operating at half the magnetic field strength of 3T. As of 2023, about 70% of global MRI installations were 1.5T, preferred in healthcare for their lower cost (typically $1-2 million vs $2-3 million for 3T), and adequate image quality for most clinical conditions, including neurological, musculoskeletal, cardiac, and abdominal disorders. 3T machines also face challenges such as heating issues and image artefacts not seen in 1.5T machines (Jamwal, 2025). Sub-Saharan Africa is abandoning low-field MRI ($\leq$1T), but as of 2022 only 3% of scanners were 3T, compared to 57.6% 1.5T and 39.4% low-field (Anazodo *et al.*, 2022).

Whilst 1.5T images are adequate for most cases, 3T scans can capture images at double the signal-to-noise ratio (SNR) allowing higher spatial resolution. For multiple sclerosis (MS), Bachmann *et al.* (2006) found 3T revealed significantly more lesions than 1.5T due to better image quality and lesion conspicuity. In stroke diagnosis, 3T scans show additional ischaemic foci (small spots of brain injury) and ischaemic lesions (tissue areas damaged in a stroke due to lack of blood flow) across different brain regions (Kuhl *et al.*, 2005). This means that 3T helps doctors better assess affected brain tissue guiding a more informed medical response.

Automated tumour segmentation speeds up clinical efficiency by drawing tumour boundaries automatically, allowing radiologists to focus on interpretation, also removing manual variations between clinicians. Accurate boundaries are crucial for radiotherapy planning, surgical navigation, and measuring tumour progression. Whilst it is unclear whether using 1.5T or 3T is better for tumour segmentation, a clearer scan is likely to yield more accurate delineation (Puzio *et al.*, 2025).

This report proposes a deep-learning super-resolution model to enhance 1.5T MRI images to 3T-like quality, having the advantage of only requiring a 1.5T scanner, which is more affordable and common in medical settings, and adequate computational power to run. As such, it would provide insights exclusive to 3T scans, such as MS and ischaemic lesions. The report also evaluates whether the proposed model can provide more accurate tumour segmentation.

1.2 Research Gaps

Although MRI super-resolution is of high interest, most prior work focuses on generic blurring rather than targeting 1.5T-to-3T enhancement specifically, or simulating acquisition artefacts. This project aims to train a super-resolution model for 1.5T data by simulating realistic 1.5T acquisition artefacts, with blurring and noise restrained to 1.5T levels, creating a simulation pipeline inspired by real-world 1.5T constraints. This lets us investigate whether tailoring the task to 1.5T offers clinical benefits over generic blurring super-resolution approaches, potentially extending its use to the many existing 1.5T scanners.

Unlike prior studies, we use a substantially larger dataset of over 1200 scans with 155 slices each, providing a wider variety of data, which intends to improve generalisation across different 1.5T scanner models. To our knowledge, few prior works systematically outline the effect of different GAN loss functions in MRI super-resolution context. We therefore investigate how various loss functions influence characteristics of reconstructed images, offering insights for researchers targeting specific visual qualities for clinical or research use.

1.3 Aim

Develop a clinically useful MRI super-resolution model that enhances 1.5T images to 3T-like quality, using degraded 3T training data that accurately reflects real-world 1.5T and 3T differences.

Chapter 2

Background

2.1 MRI Physics

Magnetic Resonance Imaging (MRI) generates detailed three-dimensional images of the internal structures of the human body using the axial, sagittal and coronal planes (Figure 2.1).

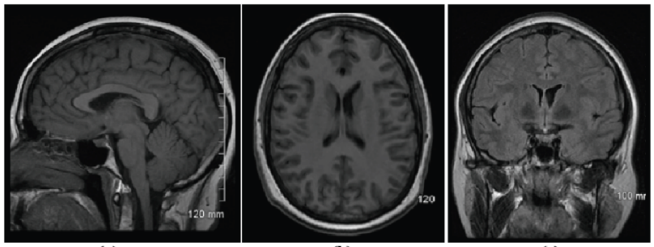

Figure 2.1: Brain MRI obtained from (a) Sagittal Plane, (b) Axial plane and (c) Coronal plane. (Kumar and Dharun, 2016)

2.1.1 Brain Matter Types

Brain matter can typically be categorised into white matter (WM), grey matter (GM), and cerebrospinal fluid (CSF). GM is found on the outside of the brain known as the cerebral cortex, WM is encapsulated beneath it, and CSF surrounds the brain and fills internal spaces, including the ventricles and subarachnoid space (Figure 2.2).

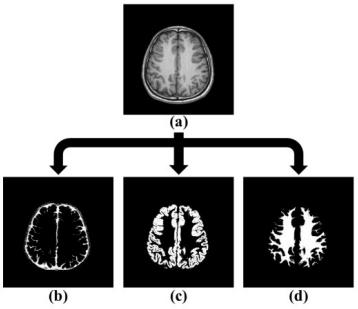

Figure 2.2: Segmentation of CSF, GM and WM. (a) MRI data; (b) CSF, (c) GM, and (d) WM. Taken from Chuang *et al.* (2012).

In MRI, white and grey matter may appear with opposite shades depending on the scan type, as each have different physical properties exploited by MRI.

2.1.2 Net Magnetisation Vectors

Hydrogen atom is abundant in human body tissue, with each nucleus exhibiting magnetic spin. Nuclear spin has a direction and precesses around an axis, pointing in an arbitrary direction unless influenced by an external magnetic field. Under an external magnetic field, hydrogen spin directions align either parallel or antiparallel to the main magnetic field direction, with more aligning parallel. This is shown in Figure 2.3.

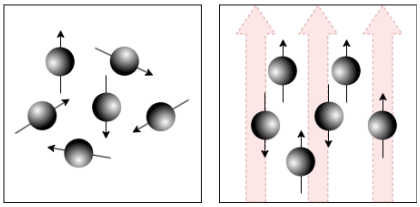

Figure 2.3: Hydrogen spin precession direction without (*left*) and with (*right*) an external magnetic field.

We combine spins into a single magnetic moment called the net magnetisation vector (Figure 2.4), aligning with the magnetic field due as more spins pointing in that direction. Individual spins precess at the same frequency, but are out of phase, creating zero net direction perpendicular to the field as their individual directions cancel each other out (Figure 2.5).

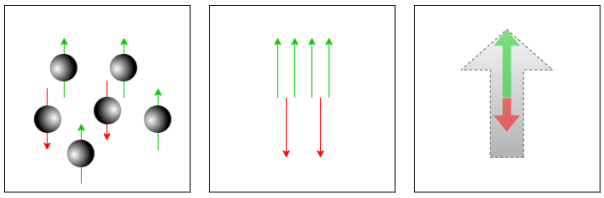

Figure 2.4: Net magnetisation vector of atoms acted upon by external magnetic field.

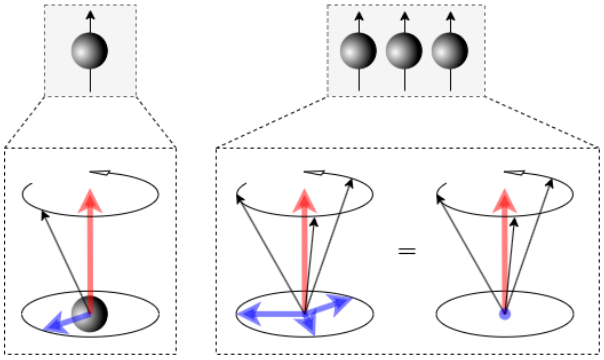

Figure 2.5: Net spin precession direction of single atom (*left*) compared with multiple atoms (*right*).

In MRI, the patient is placed into a magnetic field B_0, creating a net magnetisation vector. A second magnetic field B_1 called the Radiofrequency (RF) Pulse is then applied, perpendicular to B_0 and alternating at the hydrogen spin frequency. This has two effects on individual spins:

1. Each spin temporarily gets knocked towards the transverse plane, in the direction of B_1 and perpendicular to B_0.

2. The spin of each atom precesses in phase with one another.

With spins now in phase with a transverse component, the net magnetisation vector lies in the transverse plane and precesses about B_0. This precessing transverse magnetisation induces a measurable signal in the receiver coils (Figure 2.6).

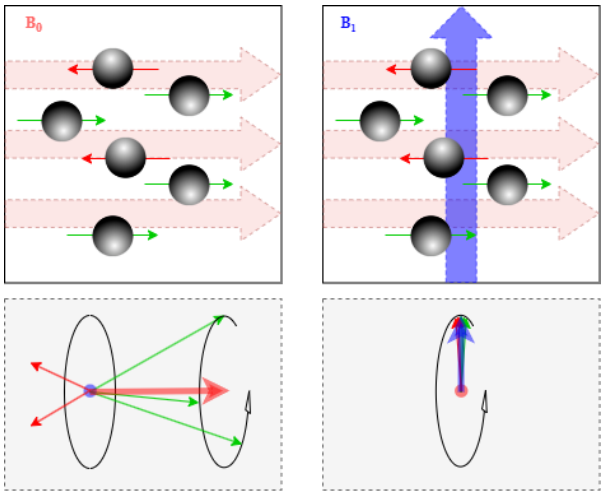

Figure 2.6: Net magnetisation vector before (*left*) and after (*right*) RF pulse.

2.1.3 T_1 and T_2 Relaxation

Whilst the RF pulse is active, all spins are in phase. When this signal stops, they slowly de-phase, eliminating the net transverse component (Figure 2.7). The time taken for de-phasing is called T_2-relaxation time. Each tissue has a different T_2 relaxation, which helps differentiate between GM, WM, and CSF.

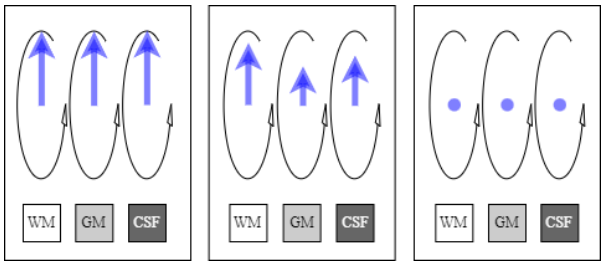

Figure 2.7: Net transverse magnetisation of different matter types at proceeding timesteps after RF pulse.

During the RF pulse, the net magnetisation has no longitudinal (B_0-parallel) component. When switched off, spins realign either parallel or antiparallel to B_0 (Figure 2.8). Time taken to restore longitudinal component is called T_1 relaxation, also varying by tissue.

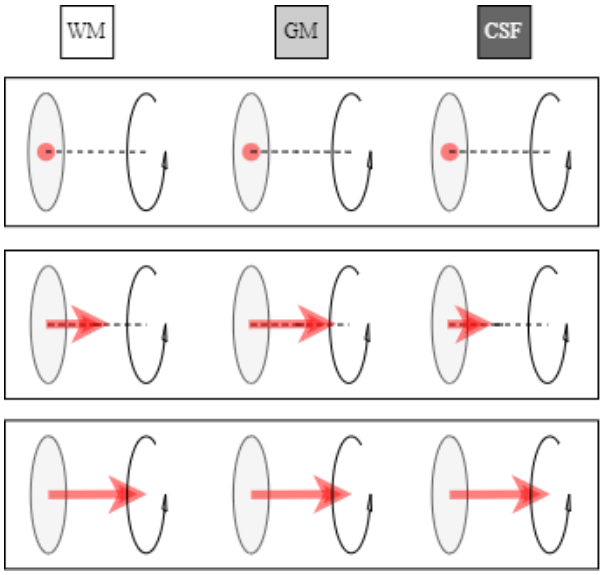

Figure 2.8: Longitudinal magnetisation of different matter types at proceeding timesteps after RF pulse applied.

2.1.4 T_2-FLAIR

T_2 FLAIR (Fluid-Attenuated Inversion Recovery) MRI measures T_2 relaxation while strategically suppressing signal from CSF, which has a longer T_2 relaxation time than GM and WM (Figure 2.9).

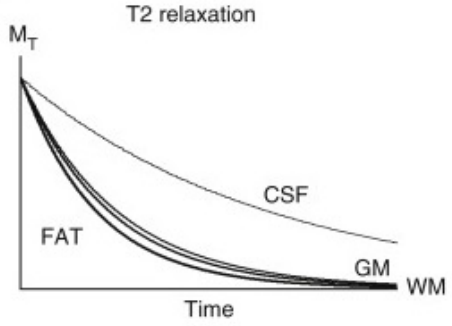

Figure 2.9: T_2 relaxation of different matter (Sodickson and Sodickson, 2016).

Suppression is done by using an inversion recovery pulse to zero cerebrospinal fluid's longitudinal magnetisation at measurement, cancelling its image contribution (Hajnal *et al.*, 1992).

This report uses T_2-FLAIR (T2F) images rather than T_2-weighted (T2W), T_1 pre-contrast (T1N), and T_1 post-contrast images (T1C) as T_2-FLAIR is regarded as the most useful sequence for lesion detection. T2F is sensitive to pathology and makes differentiation between CSF and lesions easier whereas T_1-based imaging is more useful to show anatomical structures (Hu *et al.*, 2020). We

anticipate that our model would be more clinically useful when deployed to detect lesions rather than visualising anatomical detail, so we optimised the model on T2F. Although multi-modal MRI super-resolution exists, we focused on a single modality to use our time efficiently.

T_2-FLAIR is chosen for lesion detection because it provides better contrast between lesions and surrounding tissues. In T2W, white and grey matter often appear similar while cerebrospinal fluid is much brighter, making nearby lesions difficult to see (Hashemi *et al.*, 1995). Lesions generally have smaller contrast in T1N than T2F or T2W (Smith *et al.*, 1985). T1C highlights some lesions metastases or active MS plaques using contrast agents like Gadolinium (Felix *et al.*, 1985), but is less sensitive to others, such as edema (Hua *et al.*, 2020) (Figure 2.10).

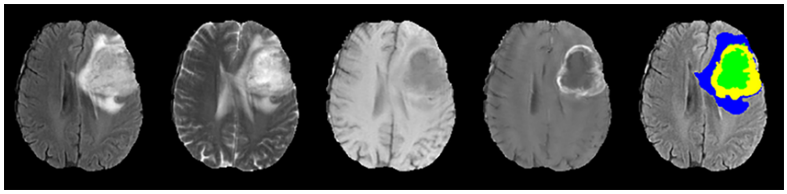

Figure 2.10: Four modalities of brain tumor: T2F, T2W, T1N, T1C (from left to right). The manual segmentation in FLAIR modality is viewed in different colors: edema (blue), necrosis and nonenhancing tumor (green), enhancing core (yellow). Image taken from Hu *et al.* (2020).

2.1.5 MRI Image Acquisition Process

MRI images are inherently 3D stacks of axial (roughly perpendicular to the spine) 2D slices, comprised of voxels (pixels containing some thickness) whose intensities reflect T_1 or T_2-relaxation of tissue. Slices are acquired in the k-space rather the spatial domain, meaning conversion is needed to display these as images.

Slice Selection

To obtain k-space slices, MRI splits the net magnetisation vector into slices, otherwise, it would only capture a single, unusable signal for the entire brain. Applying a gradient magnetic field along the z-axis (perpendicular to the spine) makes each point precess at a slightly different frequency. The machine targets a slice with an RF pulse matching its spins' precession frequency, so only that slice's protons resonate and generate transverse magnetisation.

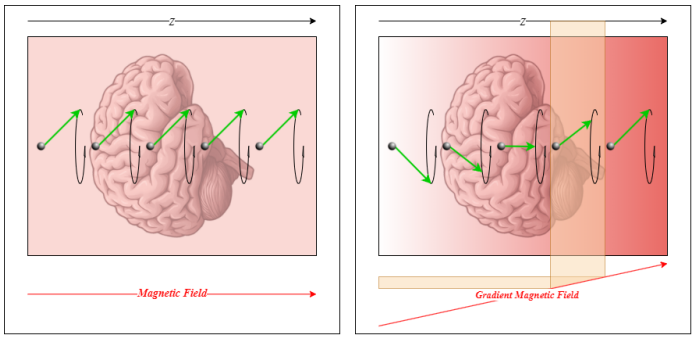

Figure 2.11: Slice selection using a gradient magnetic field.

Frequency Encoding Gradient

After a slice is selected, the resulting signal represents the slice's net magnetisation as all nuclei in this slice precess at the same frequency in phase, forming a sine wave when the amplitude is measured over time. To separate the slice by location, another gradient is applied, which changes frequencies of nuclei base on their x-axis location. When the amplitude over time is taken again, it appears as a complex wave, as different frequencies cancel out (Figure 2.12).

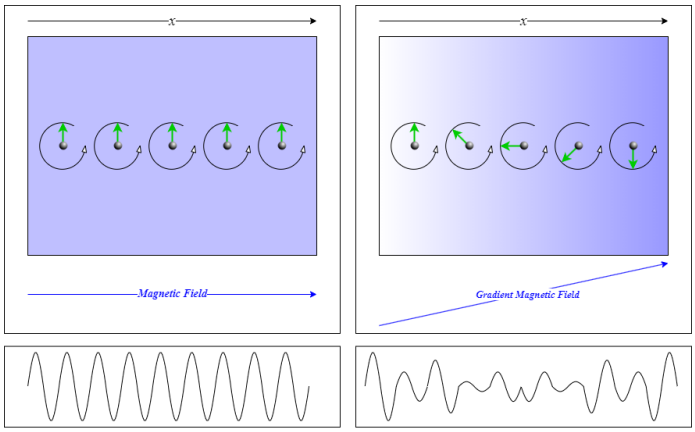

Figure 2.12: Frequency encoding gradient effect on x-axis magnetisation (*top*) and resulting amplitude waves (*bottom*).

Sampling the complex wave at discrete intervals gives numerical values representing the entire magnetisation at each instance, applying a Fast Fourier Transform (FFT) allows us to decompose the signal into its constituent frequencies. Only one unique combination of frequencies and amplitudes (along the x direction) can produce the original set of sampled values from the complex wave. As the x-axis has a gradient field, each constituent frequency maps to an x-axis. As such, so do the amplitudes, meaning that the amount of signal coming from each x-axis can be determined. The process can be seen below in Figure 2.13.

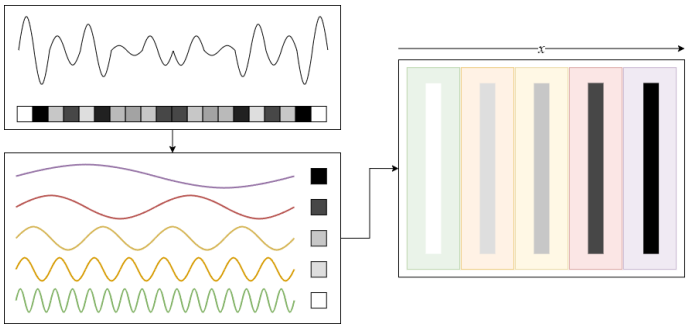

Figure 2.13: Sampled values (*top left*) and their constituent frequencies (*bottom left*) visualised on axial plane (*right*).

Phase Encoding Gradient

Frequency encoding gives x-axis signal strength but complete images require both x and y-axis data. As the frequency encoding gradient only functions when frequency data from entire y-axis columns are the same, changes to frequency cannot be made to encode y-axis data. Instead, a phase-encoding gradient briefly applied before frequency encoding creates phase shifts along the y-axis. Phase shifts are applied evenly, with maximum positive at the top of y and equal negative at the bottom while the midpoint receives no phase change.

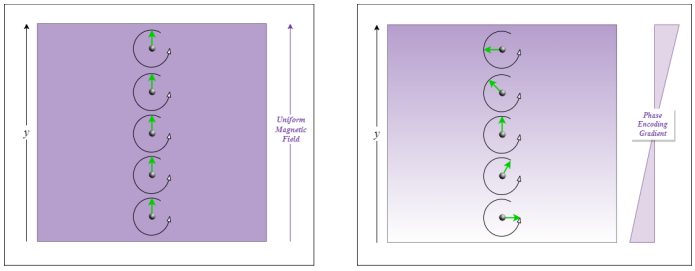

Figure 2.14: Magnetisation phase differences before (*left*) and after (*right*) phase encoding gradient

After frequency encoding, measured waves correspond to a certain phase encoding gradient strength. Repeating these measurements with different phase-encoding strengths, we can stack the Inverse Fast Fourier Transformed (IFFT) waves, giving the k-space representation of an MRI slice. Although not a spatial image, it contains spatial information in the x and y direction, allowing for transformation into a spatial image.

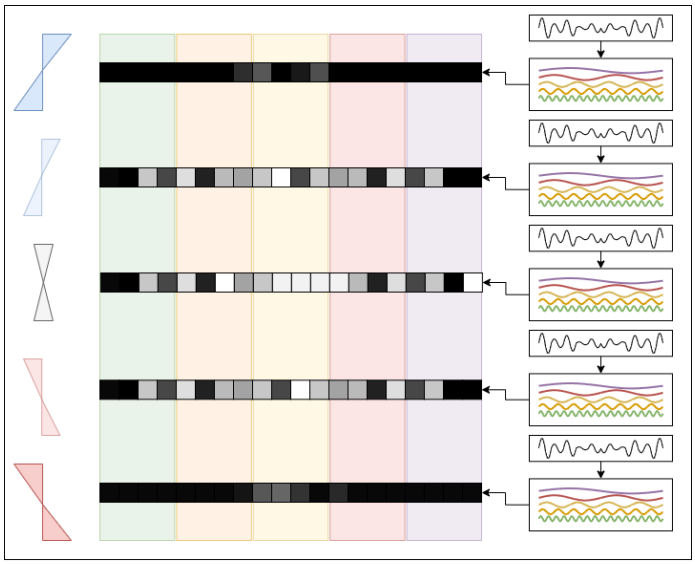

Figure 2.15: Stacking Fourier-transformed signals (*right*) from varying phase-encoding gradients (*left*).

k-Space

As mentioned earlier, *k*-space MRI signal produced using both gradient and phase encoding gradients (Figure 2.16). Each *k*-space pixel represents an entire brain slice's signal measured with marginally different frequency and phase presets with spatial information encoded by varying phase and frequency gradients across repeated measurements. Fully sampled *k*-space is converted into a spatial image using an IFFT operation, reverting frequency components to their corresponding image space positions. IFFT is reversible using FFT, allowing lossless conversion between *k*-space and the image domain.

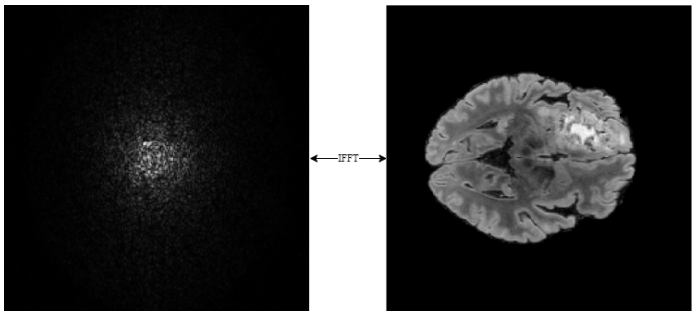

Figure 2.16: MRI signal in *k*-space (*left*) and corresponding spatial image representation (*right*).

Central *k*-space contains lower frequencies, representing image contrast and smooth features, whilst outer *k*-space contains higher frequencies, representing edges and finer detail.

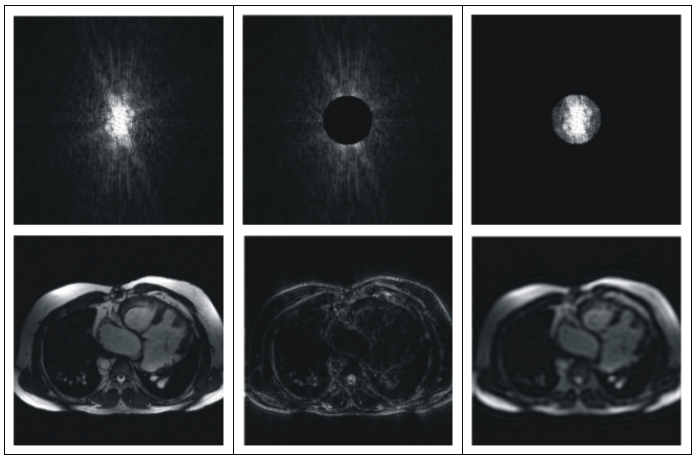

Figure 2.17: Demonstration of the effect of removing central portion (*middle*) and edges (*right*) of k-space on spatial image (Moratal *et al.*, 2008).

2.1.6 Parallel Imaging

Parallel imaging is a technique used by most modern MRI systems (Cummings *et al.*, 2022) to reduce scan time. Traditional MRI contains one receiver coil to acquire a full grid of k-space, while parallel imaging contains multiple coils positioned at different locations to simultaneously acquire a smaller (undersampled) k-space portion. Learning relationships between different coil signals can accurately reconstruct k-space (Cummings *et al.*, 2022).

2.1.7 GRAPPA

The most common parallel imaging techniques today are GRAPPA and SENSE, which are commercially supported by Siemens, Philips, General Electric, and Toshiba (Blaimer *et al.*, 2004). Focusing on simulating one of these techniques for super-resolution would maximise clinical relevance due to their availability. We focus on GRAPPA for convenience, as it operates entirely within k-space without requiring coil sensitivity maps like SENSE, although both are clinically important.

GRAPPA fully samples a central strip of k-space (Auto-Calibration Signal or ACS) to calculate the linear relationship between coils. Outside the ACS, every n-th k-space line is sampled, reducing acquisition time by a factor of n (Griswold *et al.*, 2002). This method can produce artefacts if not calibrated. The full GRAPPA reconstruction process is explained later in Section 3.1.3.

2.1.8 Partial Fourier

Partial Fourier also reduces acquisition time by undersampling k-space. In theory, k-space is symmetrical, so only half of k-space needs to be sampled to reconstruct because of a property called Hermitian symmetry. The sampled half is used to estimate the remaining k-space, cutting the acquisition time by half. In practice, phase errors from sources such as patient movement or eddy

currents cause visual artefacts, so Partial Fourier typically samples around 60-70% to minimise these (Haldar and Liang, 2022). Partial Fourier is explained in further detail in Section 3.1.4. Since Partial Fourier is used widely in modern MRI (Haldar and Liang, 2022) and often combined with GRAPPA (Frost *et al.*, 2010; Tao *et al.*, 2016; Otazo *et al.*, 2010), it is logical to train a model with artefacts from both techniques.

2.2 1.5T MRI and 3T MRI Differences

MRI systems are classified by main magnetic field strength B_0, with most clinical scanners operating at either 1.5T or 3T (Global Insight Services, 2025). Visual differences are caused by the fact that 3T produces double the signal of a 1.5T scanner.

2.2.1 SNR Doubling

A 3T scanner produces around twice the signal of a 1.5T scanner, giving radiologists the choice of whether to use the extra signal to get higher resolution images, or to reduce scan time. Theoretically, a 3T scan can double the signal-to-noise ratio (SNR) compared to 1.5T, as signal strength given by atomic nuclei increases proportionally more than noise (Edelstein *et al.*, 1986). Doubling B_0 flips some nuclei pointing antiparallel to B_0 parallel, strengthening the longitudinal magnetisation vector M, which is roughly proportional to B_0:

$$M \propto B_0$$

When the RF pulse tips spins into the transverse plane, transverse magnetization equals longitudinal magnetisation and is therefore also proportional to B_0. Additionally, when B_0 is increased, the spin frequency of atomic nuclei ω increases proportional to B_0:

$$\omega \propto B_0$$

The receiver coil signal S is proportional to the rate of change of magnetisation $S \propto \frac{dM}{dt}$. The magnetisation vector rotates around B_0 at frequency ω causing the transverse component of M to oscillate, with rate of change of magnetisation proportional to ω and M:

$$\frac{dM}{dt} \propto \omega M$$

Therefore, signal detected in receiver coils S is proportional to ω and M:

$$S \propto \frac{dM}{dt} \propto \omega M$$

As both ω and M are proportional to B_0, we can write:

$$S \propto B_0^2 \tag{2.1}$$

This means increasing from 1.5T to 3T, quadruples idealised signal strength, but increasing B_0 also increases noise, reducing SNR. More precisely, noise v is proportional to spin ω, which is proportional to B_0:

$$v \propto \omega \propto B_0 \tag{2.2}$$

Intrinsic SNR is calculated by dividing the idealised signal shown in Equation (2.1) by the idealised noise shown in Equation (2.2).

$$\boxed{SNR \propto B_0}$$

In practice, 1.5T and 3T k-space differ as noise scales with B_0 but the signal scales with B_0^2. In both strengths, outer regions of k-space contain weak signal often drowned out by noise, but a boundary exists where signal is equal to noise and this dynamic quickly changes where signal dominates noise. This boundary lies closer to the centre at 1.5T than at 3T, effectively shrinking the usable portion of k-space (Figure 2.18).

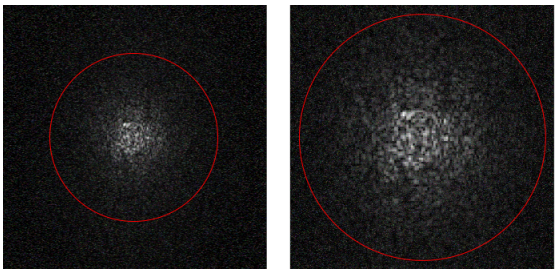

Figure 2.18: Demonstration of differing SNR boundaries on 1.5T (*left*) compared to 3T (*right*) k-space.

However, this section describes an idealised SNR B_0 relationship. Real-world SNR differences between 1.5T and 3T are complex and rely on multiple factors.

2.2.2 General Visual Artefacts

Noise

Due to lower intrinsic SNR, 1.5T images are noisier than 3T images. If no extra fixes are applied, 1.5T images would contain more noise than 3T, obscuring finer textural details which occupy outer k-space regions but are obscured by noise due to low signal.

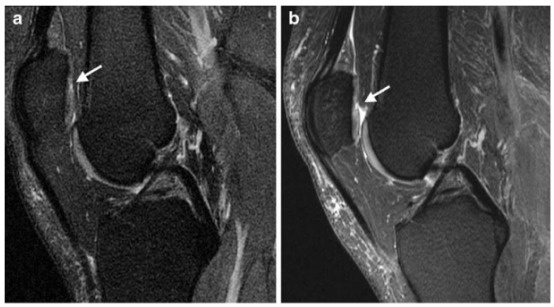

Figure 2.19: Image of knee taken in 1.5T (*left*) compared to 3T (*right*) (Wong *et al.*, 2009).

Blur

1.5T scanners reduce noise by increasing acquisition time. Over the course of multiple scans, signal is added coherently, reinforcing the same k-space voxels, while noise is added randomly cancelling out the effects over time. However, it is often unfeasible to quadruple acquisition time in clinical practice.

Another method is to lower spatial resolution, reducing voxel density. This represents the same brain slice with less voxels, making it more pixelated. This improves SNR by capturing more signal per voxel, but reduces spatial detail as each voxel averages signal over a larger area. The final image lacks detail, especially in edges and textures found in outer regions of k-space, meaning that outer k-space is either sampled less densely or not sampled when the spatial resolution is lower.

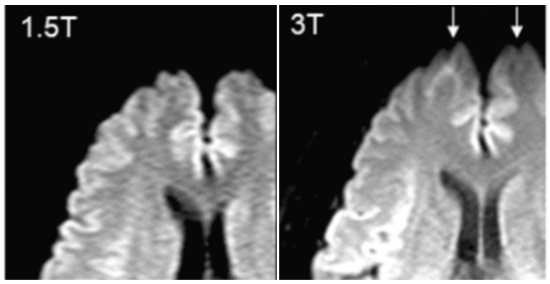

Figure 2.20: Demonstration of blur on image 1.5T (*left*) compared to 3T (*right*) (Tajima *et al.*, 2023).

Contrast

Contrast between different tissues can differ from 1.5T to 3T, as field strength affects T_1 and T_2 relaxation times as a whole and relative to each other. For example, Bottomley *et al.* (1984) reports the kidney T_1 is 32% longer than the liver's at 1.5T, but only 21% longer at 3T (Soher *et al.*, 2007). This means that the same tissues may show different contrasts at 1.5T vs 3T, but this effect isn't always visible or predictable (Soher *et al.*, 2007).

2.2.3 GRAPPA and Partial Fourier

Cartesian undersampling patterns (typically used in GRAPPA) leave distinct ghosting artefacts in the image, especially when low-frequency areas of k-space are partially undersampled. This effectively creates faint duplicates of the brain image direction perpendicular to the undersampling lines (Figure 2.21).

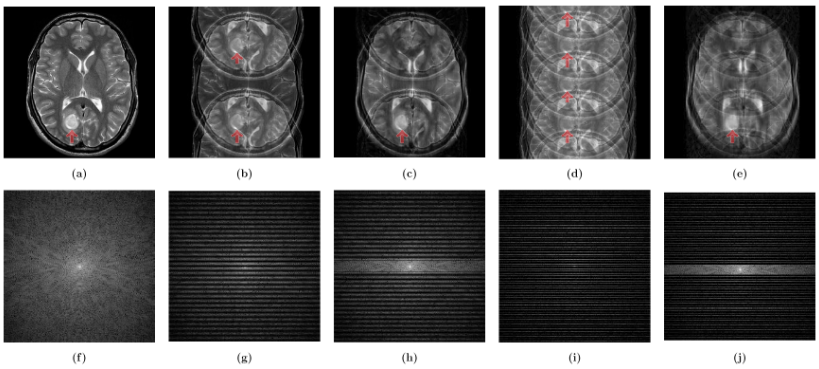

Figure 2.21: Artefacts caused by different Cartesian undersampling patterns (Hyun *et al.*, 2018).

GRAPPA undersamples k-space with a Cartesian pattern (generating artefacts seen in Figure 2.21) but also reconstructs the missing k-space lines, reducing ghosting intensity. The remaining artefact level depends on reconstruction accuracy. This is seen in Figure 2.22, which also shows artefacts from Partial Fourier truncation without reconstruction, including detail loss and ringing artefacts around edges.

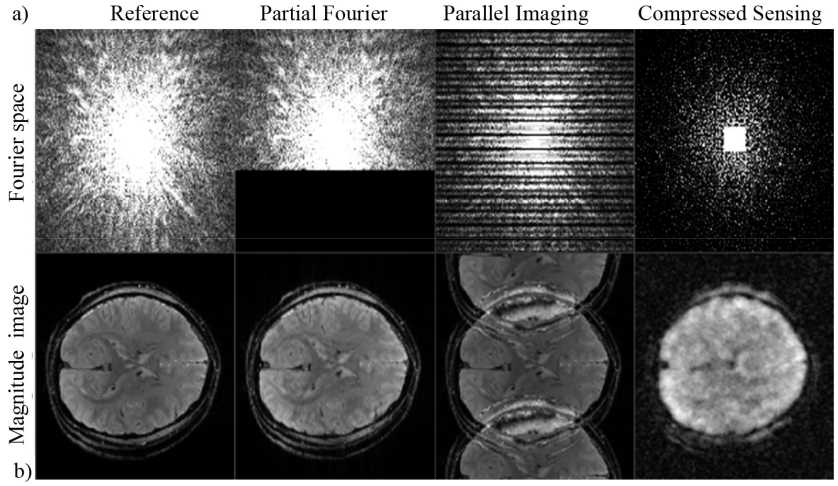

Figure 2.22: Artefacts caused by Parallel Imaging and Partial Fourier (Corbin *et al.*, 2025).

Partial Fourier artefacts naturally decrease with reconstruction. POCS Partial Fourier reconstruction method restores some detail from fully sampled k-space, but doesn't fully remove ringing artefacts around edges. Zero-filling k-space after Partial Fourier has a lower detail-level than POCS, causing slightly less visible ringing artefacts. These effects can be seen in Figure 2.23.

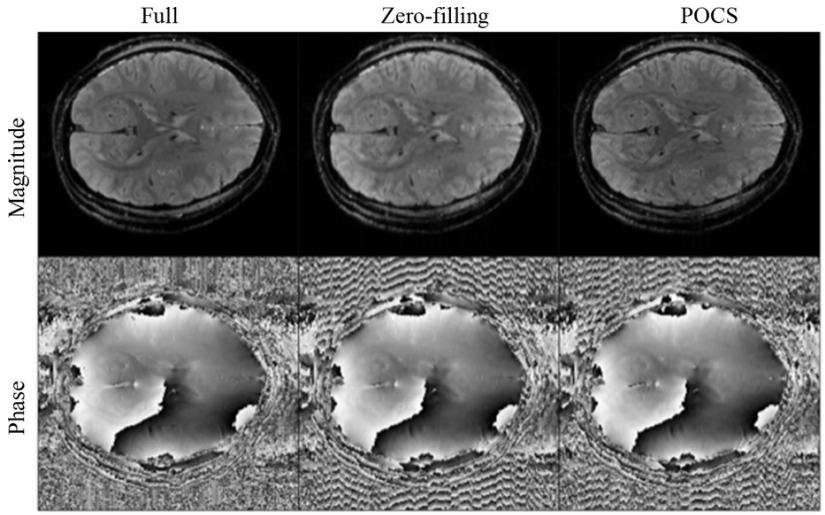

Figure 2.23: Artefacts caused by Partial Fourier after reconstruction (Corbin *et al.*, 2025).

2.3 Existing Approaches to 1.5T/3T Paired Data Challenges

Enhancing 1.5T images to 3T quality via super-resolution is not a new problem, and neither is the issue of the limited availability of paired 1.5T and 3T scans, or inherent issues with using paired scans. As such, several attempts have been made to address the data shortage. The standard approach degrades existing 3T scans to resemble 1.5T scans using a set of transformations. Some approaches also train models on the limited paired 1.5T-3T data.

2.3.1 Simple Kernel-Based Degradation

Shi *et al.* (2015) simulate blurring seen in lower-field MRI using a Gaussian filter, and downsamples to simulate the partial volume effect.

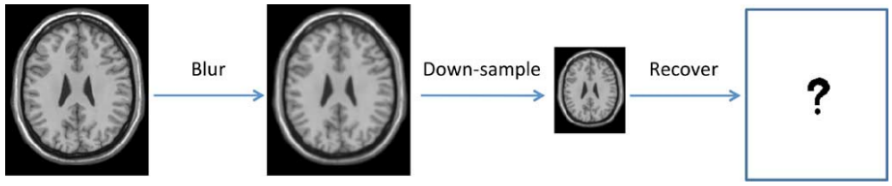

Figure 2.24: Simulation of low-resolution image from high-resolution image (Shi *et al.*, 2015).

Their Gaussian filter is applied by convolving a kernel over the original 3D HR

volume:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

They set $\sigma = 1$, which controls the Gaussian curve spread, meaning around 68% of the weight is within 1 voxel of the centre. This effectively smooths each voxel based on its surrounding neighbourhood while preserving local structure.

The second step simply averages every 8 voxels, halving the spatial resolution. This simulates the partial volume effect where multiple matter types need to be represented in a single voxel, causing the signal to be a blend of the two, which misrepresents the true intensity.

We don't follow this approach as it simplifies the degradation seen in real MRI. While Gaussian blurring is visually similar to the difference between 1.5T and 3T images, the blurring effect only occurs due to underlying k-space differences, which are not simulated. Skipping k-space degradation risks either missing or causing inaccurate visual artefacts.

2.3.2 Simulation and k-Space Degradation for Paired Data Generation

Ayaz *et al.* (2024) uses a more complex approach to 3T T1W scans from the Human Connectome Project, generating "real" and "simulated" sets of training data.

They generate "real" data by splitting 3D MRI volumes into 2D slices and converting them to k-space via FFT. They then applied a square Tukey window to the k-space, tapering values outside the window towards zero, with the steepest drop-off at the edges. Then Tukey window simulates blurring from inherent SNR differences (Section 2.2.1), though they do not specifically target 1.5T to 3T MRI differences. Afterwards, k-space is zero-padded, making LR equal spatial resolution to HR if not already.

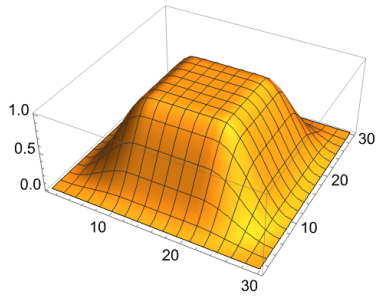

Figure 2.25: Tukey window shape, reproduced from Wolfram Research (2024).

For "simulated" data, they process HR scans with Philips Proprietary Automated Complete Brain Segmentation tool, generating a 3D map labelling each voxel as either: WM, GM, CSF, cerebellum, corpus callosum, hippocampus, brainstem, pons, amygdala, fornix, deep gray structures, and ventricles. This creates "phantoms" with anatomical information but not acquisition artefacts. Using phantoms, T1 and T2 relaxation times are assigned to each tissue, using mean and standard deviations from literature. With these relaxation times,

signal intensities are calculated using Ernst equation for Gradient Echo (GRE) to create an ideal simulated volume. With the ideal simulated volume, sampling can be performed to produce new volumes with the resolution set equal to both the target HR and LR resolutions (0.7mm and 1mm respectively). Afterwards, complex Gaussian noise is added into the k-space of the phantoms before being converted back into the spatial domain.

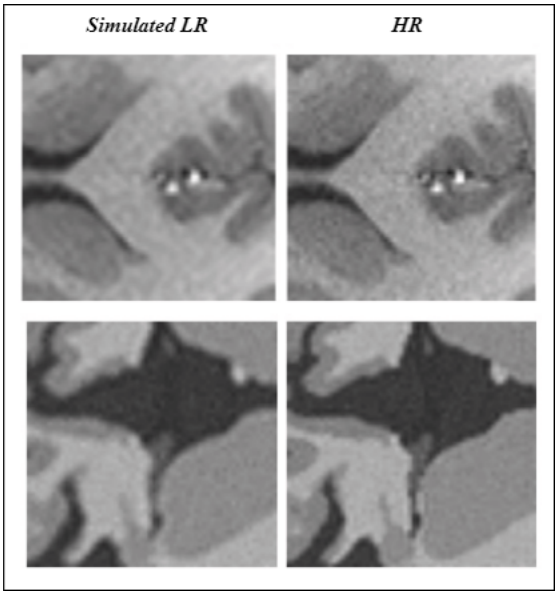

Figure 2.26: Example degradation from Ayaz *et al.* (2024).

Ayaz *et al.* (2024) included simulated data since HR pairs often contain artefacts undesirable for super-resolution. Unwanted artefacts inherent to 3T acquisition may cause super-resolution models to reproduce them, reducing diagnostic utility. Networks trained solely on simulated data performed slightly inferior to those trained purely on real data, but networks trained on a combination of simulated and real data outperformed networks trained purely on real data.

2.3.3 Paired 1.5T and 3T Methods

An approach used by Liao *et al.* (2022) was to search for 1.5T/3T pairs, which do exist in the ADNI dataset, containing scans from the same patient taken in the same visit, though not simultaneously. Since the scans are not simultaneous, they are often misaligned due to head movement. To remedy this, each pair was spatially aligned using SPM12, which estimates the best geometric transformation mapping to minimise differences between the images. However, paired data is scarce and Liao *et al.* (2022) was only able to acquire 157 image pairs for training and testing. Additionally, SPM12 spatial alignment can still contain small motion-related misalignments. Different scanners can also produce discrepancies due to bias fields and even though scans are taken on the same visit, differences in cerebral blood flow or metabolism can cause a difference in physiological state.

2.4 Machine Learning Concepts

To perform super-resolution to convert low resolution 1.5T into 3T MRI scans, we make use of deep-learning, a subset of machine learning that uses neural networks with many layers to learn patterns within data. Specifically, we make use of a generative adversarial network. As such, it is necessary to understand how neural networks function.

2.4.1 Simple Neural Networks

A neural network maps a set of numerical inputs into a desired output using some mathematical calculation. For example, a simple image binary classification model converts pixel intensity values of an input image and attempts to convert them into a single probability that the image belongs to a class.

A single "neuron" is a node which takes a set of inputs $x_1, x_2, ..., x_n$ and multiplies them by corresponding weights $w_1, w_2, ..., w_n$ to compute a weighted sum. It then adds a bias value b and passes the result through an activation function f to introduce non-linearity (Rosenblatt, 1958).

$$y = f\left(\sum_{i}^{n} = 1 w_i x_i + b\right)$$

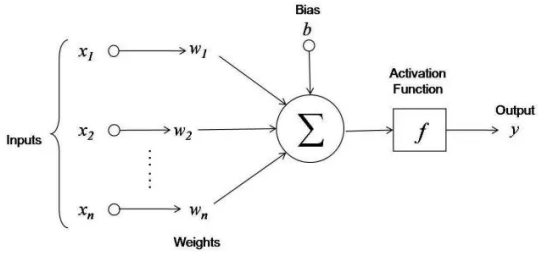

Figure 2.27: Neuron diagram (Arnx, 2019).

In a neural network, a layer is made up of multiple neurons (Rumelhart *et al.*, 1986). Each neuron in the layer takes as input the outputs from the previous layer, and its own output is then passed on to the next layer. A neural network can be thought of as 3 consecutive layers (input, hidden, and output).

In the input layer, raw data features (such as pixel intensities) are fed into the network. Afterwards, there can be multiple hidden layers consisting of multiple neurons that transform these inputs through weighted sums and activation functions, letting the network capture increasingly abstract patterns in the data. Finally, the output layer produces the network's prediction (such as a probability).

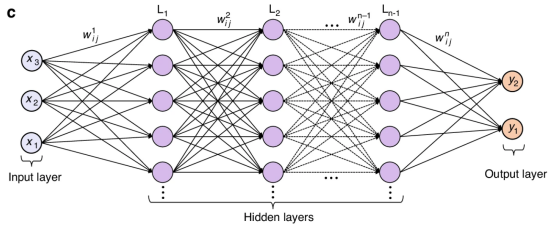

Figure 2.28: Neural network diagram (Fu *et al.*, 2024).

To train a neural network, the weights w_i and biases b are adjusted so that the network's output y is as close as possible to the desired target value for a given input x. This is typically done using a loss function, which quantifies the difference between the predicted output and the true output assigning it a numerical score.

2.4.2 Convolutional Neural Networks

Convolutional neural networks (CNNs) differ from simple neural networks as they modify the hidden layers by adding convolution layers (Lecun *et al.*, 1998). Whilst in a simple neural network, the inputs must be 1D, CNNs accept 2D inputs in the form of a 2D matrix. CNNs are commonly used with images, which are inputted as a 2D matrix containing pixel intensities, retaining spatial information.

A convolution layer applies a small matrix called a kernel to the input. The kernel slides over the input matrix, and at each position, performs an element-wise multiplication with the corresponding local region of the input. The results are then summed to produce a single output value for that position. When repeated across the whole input matrix, this creates a 2D output feature map, which can highlight specific patterns such as edges or shapes. The values inside the kernels are weights that are continuously adjusted in the training process. CNNs can be used in super-resolution by replacing a simple output layer with a single convolution layer, which combines feature maps to predict the pixel values of the upscaled output.

2.4.3 Generative Adversarial Networks

A Generative Adversarial Networks (GAN) contains two separate CNNs, a generator and a discriminator (Goodfellow *et al.*, 2014). The generator takes a low-resolution image as input and produces a high-resolution image while the discriminator is a classification CNN which evaluates whether an image is real or generated. During training, the generator uses the discriminator's classifications in a loss function, trying to fool it into predicting generated images as real. Similarly, the discriminator uses the generator's reconstruction to update its weights to better distinguish real images from generated ones. With this adversarial process, the generator produces increasingly realistic images, and the discriminator becomes more accurate at identifying subtle differences between real and generated images.

2.5 Existing Super-Resolution Models

Single image super-resolution (SISR) is the problem of trying to reconstruct a high-resolution image from a low-resolution image. Generally super-resolution models are either CNNs, GANs, or Transformers. Super-resolution in the MRI domain is an active field of research, and there is also some literature looking at scaling 1.5T scans to 3T scans.

2.5.1 CNN-Based Super-Resolution

One of the earliest CNN-based super-resolution models was SRCNN, which uses a shallow CNN containing only three convolution layers, this approach showed promising results for its time (Dong *et al.*, 2015), inspiring other CNN architectures. FSRCNN extends SRCNN to five convolutional layers, improving PSNR and reducing training time (Dong *et al.*, 2016). FSRCNN has also been experimented with low-field to high-field MRI super resolution by Ayaz *et al.* (2024), achieving comparable but lower SSIM and PSNR scores compared to GAN-based methods. U-Net has also been adapted for super-resolution and includes an encoder-decoder structure and skip connections, allowing it to preserve fine details, resulting in significantly better PSNR and SSIM scores than SRCNN and FSRCNN (Lu and Chen, 2019) but has a significantly higher number of parameters. DeepUResNet further improves upon this by incorporating residual learning into the U-Net architecture, allowing for deeper models that still train effectively. When applied to MRI data, DeepUResNet provides a similar performance to FSRCNN (Ayaz *et al.*, 2024), but is far deeper and takes much longer to train. We focus on FSRCNN due to its high performance in MRI super-resolution and compactness.

2.5.2 GAN-Based Super-Resolution

Recently, SISR has increasingly used GANs, with adversarial training providing an improvement in preserving texture and finer details compared to CNNs. The first GAN model to be proposed in the context of SISR was SRGAN, which uses a CNN-based perceptual loss function to penalise differences in deep features, rather than pixels (Ledig *et al.*, 2017). Later an enhanced SRGAN (ESRGAN) was proposed by Wang *et al.* (2018) which uses Residual-in-Residual Dense Blocks (RRDBs) and removes batch normalisation, which achieves better visual quality with more realistic and natural textures. SISR with ESRGAN has also been evaluated on MRI images, outperforming DeepUResNet and FSRCNN in terms of SSIM and PSNR on real data (Ayaz *et al.*, 2024). GANs focus on generating visually appealing images, but are prone to generating artefacts and are notoriously unstable when training (Kozlov *et al.*, 2024).

2.5.3 Transformer-Based Super-Resolution

Recently super-resolution research has begun exploring attention-based techniques. Transformers are characterised by a multi-head self-attention mechanism, which has proven effective in processing sequential data. However, the original Transformer model has quadratic computational complexity with

respect to the size of the input image, making them computationally expensive while achieving comparable results to modern CNNs (Kozlov *et al.*, 2024).

For example, Image Processing Transformer (IPT) achieves significant performance improvements from CNNs like RCAN, but also contains $\times 7$ the number of parameters, and also underperforms on small datasets. More recent Transformer-based models such as SwinIR, EDT, and HAT reduce the number of parameters to be comparable RCAN, whilst achieving better reconstruction performance when applied to non-specialised SISR, but these models generally require vast amounts of training data (Kozlov *et al.*, 2024). While Transformers are effective at enhancing global structure (low-frequency) content, they can also underperform when trying to recover fine details and edges (high-frequency) (Bai *et al.*, 2022). Overall, whilst having a high potential for outperforming CNNs and GANs, we chose to not use Transformers due to their higher performance cost, relative novelty, and potential underperformance risk.

2.5.4 2D vs 3D Super-Resolution

MRI captures the brain as a 3D volume, making 3D super-resolution preferable to SISR, since inter-slice relationships contain valuable information. Sanchez and Vilaplana (2018) extended the SRGAN structure to accept 3D data, and Wang *et al.* (2020) applied the ESRGAN structure to 3D MRI data. However, 3D models are more computationally expensive than 2D models and normally require the volume to be split into patches. U and M. (2024) merged three consecutive slices into one, which reduces the computational cost while allowing the model to build a limited view of 3D information, achieving promising results when applied to MRI images. We did not focus on pure 3D super-resolution due to the computational intensity, and while the slice interpolation approach by U and M. (2024) was intriguing, we wanted to focus this project on exploring different models and loss functions.

2.6 BraTS 2024 Dataset

The BraTS 2024 dataset is a combination of several smaller MRI datasets, designed for different sub-challenges in the BraTS 2024 challenge cluster. The primary reason why we selected this dataset is due to its ease of access, while other brain MRI sets would likely be appropriate, our institution hosts this dataset on an institutional GitLab repository making it easy to use. The dataset is split into the following categories:

Dataset	Challenge	Training Size	Validation Size
BraSyn	Synthesis (Global) - Missing MRI	1251	219
GLI	Segmentation - Adult Glioma Post Treatment	1350	188
GoAT	Segmentation - Generalizability	1351	451
LocalInpainting	Synthesis (Local) - Inpainting	1251	219
MEN-RT	Segmentation - Meningioma Radiotherapy	500	70
MET	Segmentation - Brain Metastases	652	88
Path	Pathology	N/A	N/A
PED	Segmentation - Pediatric Tumors	261	91
SSA	Segmentation - BraTS-Africa	60	0

The only datasets that contain T2F scans are BraSyn, GLI, GoAT, MET, PED and SSA. Each of the scans contain a lesion, and a ground truth segmentation map denoting the location and sections of the lesion. Each of the scans are also taken in T1C, T1N, T2F, and T2W, creating a directory structure as follows:

```
BraTS-DATASET-PATIENT-VISIT
├── BraTS-DATASET-PATIENT-VISIT-t1c.nii.gz
├── BraTS-DATASET-PATIENT-VISIT-t1n.nii.gz
├── BraTS-DATASET-PATIENT-VISIT-t2f.nii.gz
├── BraTS-DATASET-PATIENT-VISIT-t2w.nii.gz
└── BraTS-DATASET-PATIENT-VISIT-seg.nii.gz
```

We chose to focus this project singularly on the BraSyn dataset. The PED, SSA, and MET datasets did not contain enough scans to warrant use on their own, though we could have merged them into a single training set with BraSyn, 1251 scans was already pushing our hardware to the limit. While GLI contained the most scans, they were lower quality, with each slice along the axial plane in the BraSyn dataset being 240×240 as opposed to 180×218 which resulted in a clearer and less pixelated image.

The Brain MR Image Synthesis for Tumor Segmentation (BraSyn) dataset is used in the BraTS 2023 challenge (Li *et al.*, 2024) and is based on the RSNA-ASNR-MICCAI-BraTS-2021 challenge (Baid *et al.*, 2021). It contains a diverse set of scans from different institutions and scanners, making it useful in creating a model that tries to generalise the process of 1.5T to 3T enhancement across a number of settings. Each scan in the dataset contains 155 slices of 240×240 along the axial plane.

Chapter 3

Methodology

We developed a simulation pipeline that estimates to generate 1.5T-like scans from 3T images and applied it to the BraTS dataset to create LR, HR pairs, where the HR data consists of the original 3T scans from BraTS and LR scans were generated from the HR scans using our pipeline. Using these pairs as training data, we trained an ESRGAN model to enhance 1.5T images to 3T-quality.

3.1 Simulation Pipeline

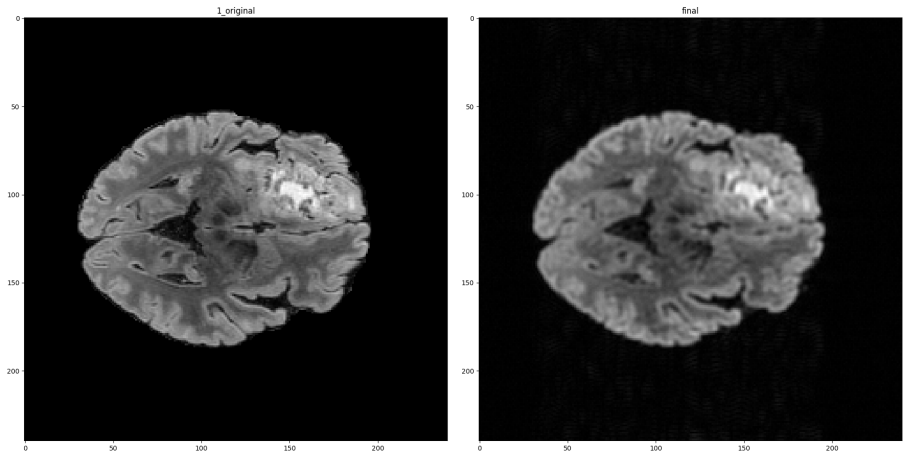

Figure 3.1: Example of slice before (*left*) and after (*right*) simulation pipeline.

Our simulation pipeline takes a high resolution 3D MRI scan volume $HR \in^{W \times H \times D}$ from the BraSyn subset of the BraTS dataset, where W, H, D represent spatial dimensions. The pipeline first transforms HR into the frequency domain via the Fourier transform:

$$K = \mathscr{F}(HR)$$

The pipeline transforms HR into $LR \in^{W \times H \times D}$ using a series of transformations $\mathcal{T}(K)$ on its k-space frequency component to simulate GRAPPA and Partial Fourier acquisition protocols.

$$\widetilde{K} = \mathcal{T}(K)$$

In reality, $\mathcal{T}(K)$ represents a set of transformations:

$$\mathcal{T}(K) = \mathcal{T}_3(\mathcal{T}_2(\mathcal{T}_1(K)))$$

Where:

- T_1 represents a cylindrical low-pass filter with a Tukey window.
- T_2 represents GRAPPA acquisition simulation.
- T_3 represents Partial Fourier.

A noise component ϵ is then added to the resultant k-space volume, and the volume is then converted back into the image domain through another Fourier transform to create the final volume LR.

$$LR = \mathscr{F}^{-1}(\widetilde{K} + \epsilon)$$

A visualisation for the process on a single 2D MRI slice can be seen in Figure 3.2.

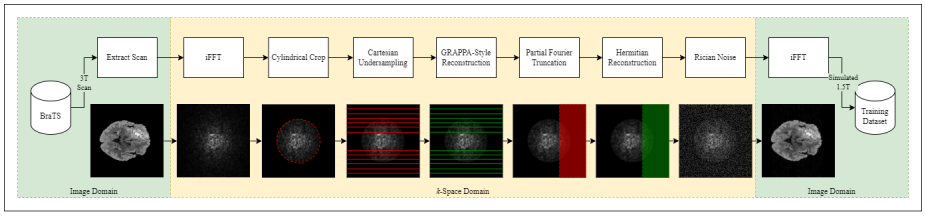

Figure 3.2: 3T to 1.5T Simulation Pipeline

3.1.1 Conversion into Frequency Domain

We first use a Fast Fourier Transform (FFT), denoted as $\mathscr{F}$, to convert the HR image into k-space. We then normalise the image using unitary FFT normalisation.

$$K = \frac{\mathscr{F}(HR)}{\sqrt{N_x \cdot N_y \cdot N_z}}$$

This normalisation keeps the overall intensity the same before and after the transform, so the image doesn't get artificially brighter or darker when converted.

3.1.2 Cylindrical Low-Pass Filter with Tukey Window

After converting the volume from the spatial domain into k-space, we first apply a cylindrical low-pass filter to the k-space $K_{z,y,x}$, zeroing out any values outside the filter. When thought of slice-wise, this effectively draws a centred circle

on each slice of 2D k-space and zeroes out values outside this circle. We then smooth the edges of this window by applying a Tukey function along the radius of this filter to reduce unwanted artefacts.

$$\mathcal{T}_1(K_{z,y,x}) = K_{z,y,x} \cdot w(r(z,y))$$

The first step of this process is computing the distance $r(z,y)$ of each point (where height and width are y and z) from the centre of 2D k-space c_y, c_z. These values are then scaled with a filter radius parameter R which we set to half the width of the image so that values outside the radius are greater than 1 and values inside are less than 1.

$$r(z,y) = \frac{\sqrt{(y-c_y)^2 + (z-c_z)^2}}{R}$$

Using $r(z,y)$, We then smooth out the edges using a Tukey window $w(r)$. There are three categories of smoothing, the passband, taper region, and stopband:

1. Values inside the passband ($r \leq 1-\alpha$) retain their full original intensity as these are well within the filter radius.

2. Values inside the taper region ($1-\alpha < r \leq 1$) lie close to the filter radius, therefore their intensities near the outer edge are tapered from 1 to 0 using a cosine function

3. Values inside the stopband ($r > 1$) are completely suppressed as these are outside filter radius.

We manually calibrate the steepness of the tapering with the smoothing parameter α which we set to 0.5.

$$w(r) = \begin{cases} 1, & r \leq 1-\alpha \\ \frac{1}{2}\left[1 + \cos\left(\pi \frac{r-1+\alpha}{\alpha}\right)\right], & 1-\alpha < r \leq 1 \\ 0, & r > 1 \end{cases}$$

The resulting filter and effect of $\mathcal{T}_1$ can be seen in Figure 3.3 below:

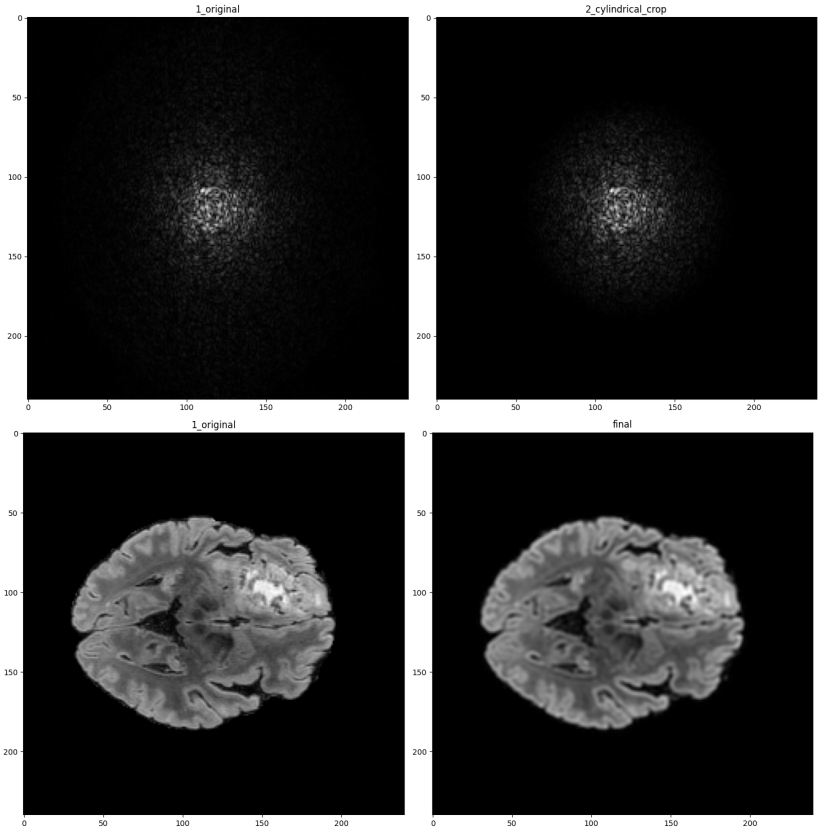

Figure 3.3: Filter (*left*) and resulting k-space (*right*) after applying $\mathcal{T}_1$.

k-Space Filter Justifications

The reason that we opted for a k-space filter similar to the one used by Ayaz *et al.* (2024) and not a kernel-based method similar to Shi *et al.* (2015) our approach is slightly more realistic for 3T to 1.5T image degradation. When scans are taken at 1.5T, the SNR is inherently lower (see Section 2.2.1), meaning that signal found at the edges of k-space which represent finer details are obscured by noise data whereas the opposite is true for more central portions. The result is that contrast and structure is preserved whilst finer detail is obscured in the final image for lower-field scans.

Even though the blurring effect and comparative lack of detail caused by lower-field scans appears similar to standard uniform blurring, they are not exactly the same. Uniform blurring techniques, such as ones used by Shi *et al.* (2015), pass a Gaussian kernel over the image, which smooths the image uniformly for the whole image as opposed to varying spatially depending on frequency content. It is also difficult to target exact frequency components using uniform blurring whereas filtering in k-space allows us direct control over the frequency bands to suppress. Another issue would be that tissue boundaries, such as the boundary between white matter and cerebrospinal fluid, could be blurred in an unrealistic manner due to the softening effect of uniform blurring. Using a k-space filter keeps these sharper in a more realistic manner

as boundaries tend to be found in the more central regions of k-space which are entirely preserved. Overall, using k-space degrades images in a more true-to-life manner by operating in k-space.

Cylindrical Low-Pass Filter Justifications

The specific low-pass filter that we used was cylindrical (or circular if observed slice-wise). The circular shape was mainly used to avoid streak artefacts which could be present if a square shape, or indeed any shape with distinct edges, were used (Figure 3.4). Additionally, frequency data is naturally radial in the sense that the distance from the origin at any angle represents the frequency magnitude (lower spatial frequencies towards the centre and higher towards the edges), so it would not make sense to apply a filter that treats frequencies unevenly along different directions. For example, a square filter includes higher frequencies at its corners than along the middle of its sides because the corners are further from the origin in k-space, allowing higher frequency components in those diagonal directions.

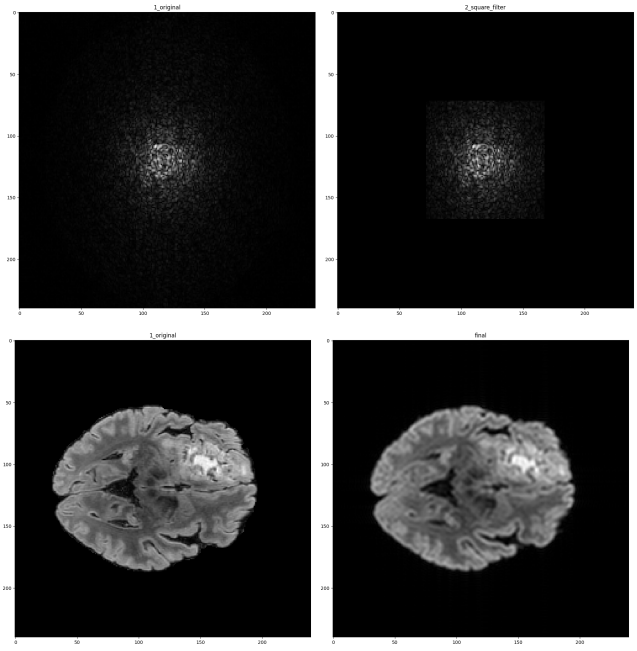

Figure 3.4: Artefacts caused by using a different square filter.

We chose to use the radius parameter R equal to half the width of the image as this matches the spatial resolution ratio between 3T and 1.5T plus added leeway for the Tukey window. 3T scans tend to contain more frequency content than 1.5T, meaning that their maximum frequency f_{max} in k-space is higher. We create an estimation for this ratio based off of their respective isotropic resolution, where 3T scans have a higher isotropic resolution than 1.5T.

$$r = \frac{f_{max,1.5T}}{f_{max,3T}} = \frac{v_{3T}}{v_{1.5T}}$$

This ratio is then multiplied by the original 3T radius R_{3T}, and the result is used as the 1.5T radius R.

$$R = r \times R_{3T}$$

The BraTS dataset contains 3T scans that have an isotropic resolution of 1mm^3. Whilst there is not one true isotropic resolution for 1.5T scans, Ayaz *et al.* (2024) used a 0.7mm and 1mm resolution for their HR and LR sets, so we also opted for a similar ratio of 1mm and 1.5mm. This gave us an R value of $\frac{2}{3}$.

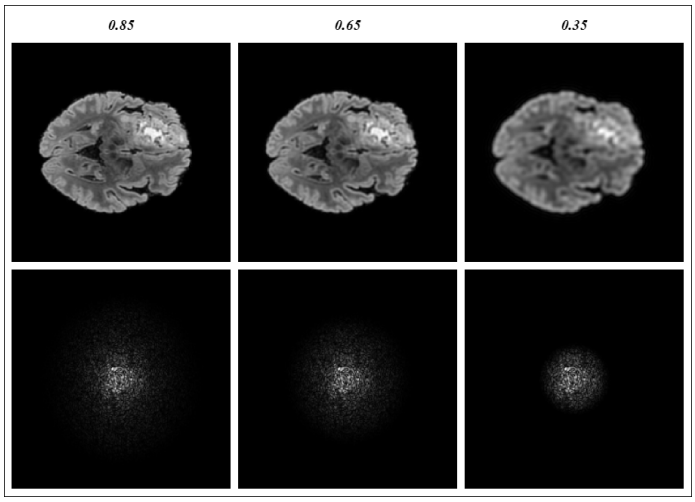

Figure 3.5: Effect of varying radii on the spatial image.

Tukey Window Justifications

The main reason for including a Tukey window on the radius of our low-pass filter was to avoid ringing artefacts caused by the harsh cut-off in k-space due to the filter. The purpose of the pipeline is to create a realistic simulation so that any artefacts are drawn from real acquisition techniques, so any artificial effects introduced by overly abrupt truncation were removed.

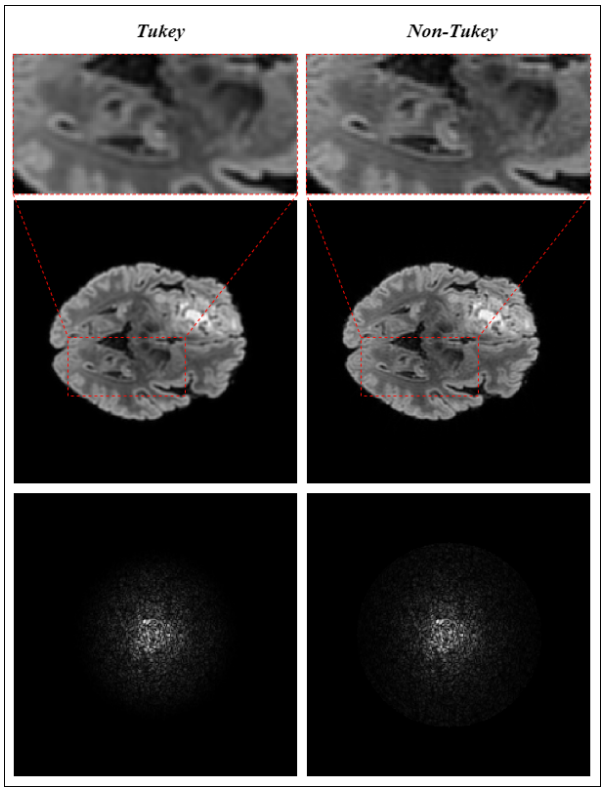

Figure 3.6: Example of ringing artefacts caused by not having a Tukey window.

The main reason as to why we used a Tukey window was because the window plateaus at the top and contains smoothing where the gradient tapers off at both extremes with the steepest gradient being in the centre. This means that the intensities scale in a non-abrupt manner. The plateau ensures that any values within the passband radius do not have their intensities changed, concentrating any smoothing effect on the border. Overall, any alternatives that contain a cosine taper and a plateau would be equally viable. In the end, the smoothing parameter $\alpha = 0.4$ was chosen through visual inspection of several slices as we observed the least ringing artefacts at this particular value whilst still maintaining the radius chosen using r.

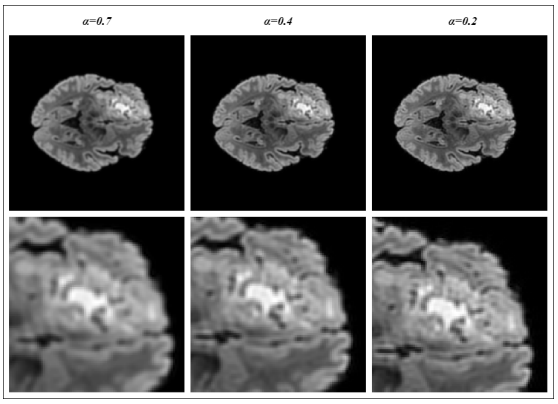

Figure 3.7: Examples of different artefacts caused by varying α.

3.1.3 GRAPPA Acquisition Simulation

After degrading the k-space using our filter $\mathcal{T}_1$, we then run the filtered k-space through a simulation of GRAPPA acquisition, denoted as $\mathcal{T}_2$.

GRAPPA in Clinical MRI

As mentioned in Section 2.1.7, GRAPPA is a commonly used parallel imaging protocol, used to reduce acquisition time by capturing an undersampled version of k-space with multiple receiver coils simultaneously. This section will explain it in more detail.

GRAPPA first requires a fully sampled strip of k-space, which is called the Autocalibration Signal (ACS), from all receiver coils. It also captures every N lines parallel to the ACS, which are kept constant across all coils. This process leaves missing points in k-space parallel to the ACS. This sampling pattern can be seen in Figure 3.8.

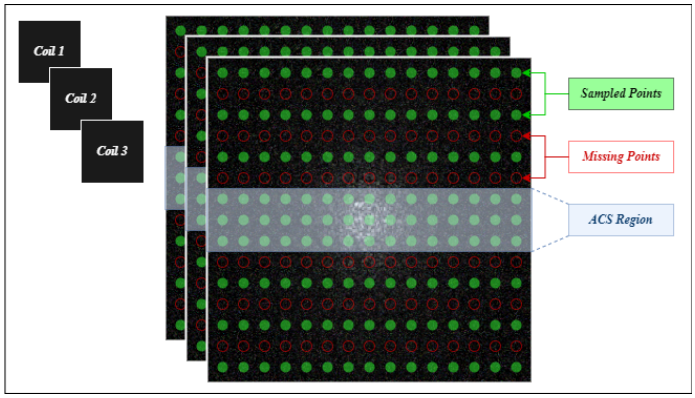

Figure 3.8:

GRAPPA assumes that a missing point in k-space $\widehat{ACS}(x, y)$ can be predicted

by using the weighted sum of its neighbours across all coils using a 3D kernel.

$$\widehat{ACS}(x, y) = \mathbf{w} \cdot \mathbf{n}(x, y)$$

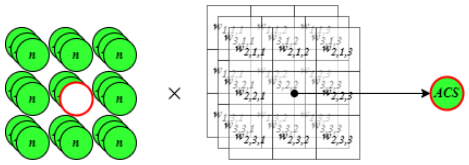

Figure 3.9: Diagram of single ACS prediction using kernel.

Here, $n(x, y)$ is a vector containing all acquired points from the local neighborhood around (x, y) across all coils, and $\mathbf{w}$ contains the corresponding learned kernel weights. This kernel is trained using linear regression, least-squares, or another appropriate method to learn the weight values in the kernel $\mathbf{w}$.

$$\mathbf{w} = \text{argmin}||Y_{ACS} - X_{ACS} \cdot \mathbf{w}||$$

Once the kernel weights are trained, it is passed over each of the missing k-space data points and used to fill in the missing values, creating complete k-space data.

The reason we chose to simulate this protocol in the pipeline was because we wanted to see if we could train a model to eliminate bad acquisition artefacts commonly used in 1.5T MRI such as SENSE or GRAPPA. Whilst there exists models which aim to super-resolve 1.5T to 3T-like quality, there has not been much focus on particular acquisition protocols which are common in 1.5T scans. These protocols can sometimes leave artefacts, which are worth correcting as a model trained to purely solve 1.5T to 3T super-resolution may struggle when faced with scans from standard MRI machines which have not been properly calibrated in 1.5T. Whilst there is evidence that 3T scans generally contain more artefacts than 1.5T due to their increased SNR (Bernstein *et al.*, 2006), in a super-resolution task, it is more clinically advantageous if a model can be trained to cancel out these acquisition artefacts rather than replicate them, hence why our LR set contains GRAPPA artefacts and the HR does not, which may not necessarily reflect real-life. More broadly, if we can successfully correct artefacts introduced by GRAPPA, this could eliminate the need for longer non-GRAPPA acquisitions, thereby improving imaging efficiency without sacrificing quality. The main reason we specifically focused on simulating GRAPPA over a popular alternative SENSE was because SENSE requires coil sensitivity maps to make predictions, which we do not have access to.

Our Process Overview

With the background knowledge of how GRAPPA functions, our simulation $\mathcal{T}_2$ can be explained. First we perform Cartesian undersampling, using a 3D mask of the same shape as the 3D k-space volume using a defined gap between each line and an ACS region of width, both of which are set to 1 whilst the missing values are set to 0.

The second stage involves reconstructing the missing lines using interpolation. We first extract the ACS region and use it to train the weights of a

2D kernel using least squares. The BraTS dataset does not contain data from multiple coils for each slice, so we treat k-space as a single coil. This means that our data for each slice is only 2D as opposed to 3D, meaning that we only train a 2D kernel, not a 3D one. Afterwards, this kernel is passed over each missing point in k-space and used to predict its value.

$$\mathcal{T}_2(K_{z,y,x}) = \mathcal{T}_{\text{reconstruct}}(\mathcal{T}_{\text{undersample}}(K_{z,y,x}))$$

Sampling Maps

In more detail, we first create two masks, one representing the Cartesian stripe pattern M_{cart} and one representing the ACS central stripe M_{ACS}. These are then merged into one map where the maximum value takes overwrites a smaller value, making the operation equivalent to a logical OR $\vee$ as only values 0 and 1 exist in the masks. The k-space is then multiplied by each mask.

$$\mathcal{T}_{\text{undersample}}(K_{z,y,x}) = (M_{cart} \vee M_{ACS}) \cdot K_{z,y,x}$$

For our first map M_{cart}, we start off with a zero-filled map. We then define a gap width between lines R and set every gth line to 1. In our final version we set $R = 4$:

$$M_{cart}(z, y, x) = \begin{cases} 0, & \text{if } y \bmod R = 0 \\ 1, & \text{otherwise} \end{cases}$$

For the ACS map M_{ACS}, we also start off with a zero-filled map. We define the ACS spine width $s = 64$, and set a "spine" region using the following where Y represents the height of the k-space:

$$y \in [\frac{Y - s}{2}, \frac{Y + s}{2}]$$

Using this spine definition and the zero-filled map, we then fill any points that lie in the spine region with 1:

$$M_{ACS}(z, y, x) = \begin{cases} 1, & \text{if } \frac{Y-s}{2} \leq y \leq \frac{Y+s}{2} \\ 0, & \text{otherwise} \end{cases}$$

As seen in Figure 3.10, our sampling map is very similar to a typical GRAPPA sampling map, including a similar ACS.

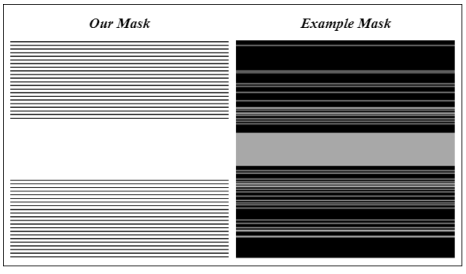

Figure 3.10: Our map (*left*) compared to a real GRAPPA sampling map (*right*).

Typically, the level of undersampling (equivalent to our parameter R) ranges from around 2-4 (Chang *et al.*, 2022), with values higher than 4 leading to a disproportionately higher level of artefact the more it is increased. As we wanted to simulate a relatively high level of artefact for our model to learn, we opted for $R = 4$. We based the ACS width parameter s on real-world ACS width in proportion to k-space, keeping in mind that our kernel will necessarily underperform as there is only data from one "coil". Typically ACS width ranges from 5% to 11% of the image for $R = 4$ (Chang *et al.*, 2022), so in our case of a 256×256 voxel image this would range between $s = 16$ and $s = 30$. However, we decided to increase this to $s = 64$ as we expect significant extra degradation as we have only one reference coil. This lead us to an appropriate level of artefact that didn't fully obscure anatomical details (Figure 3.11).

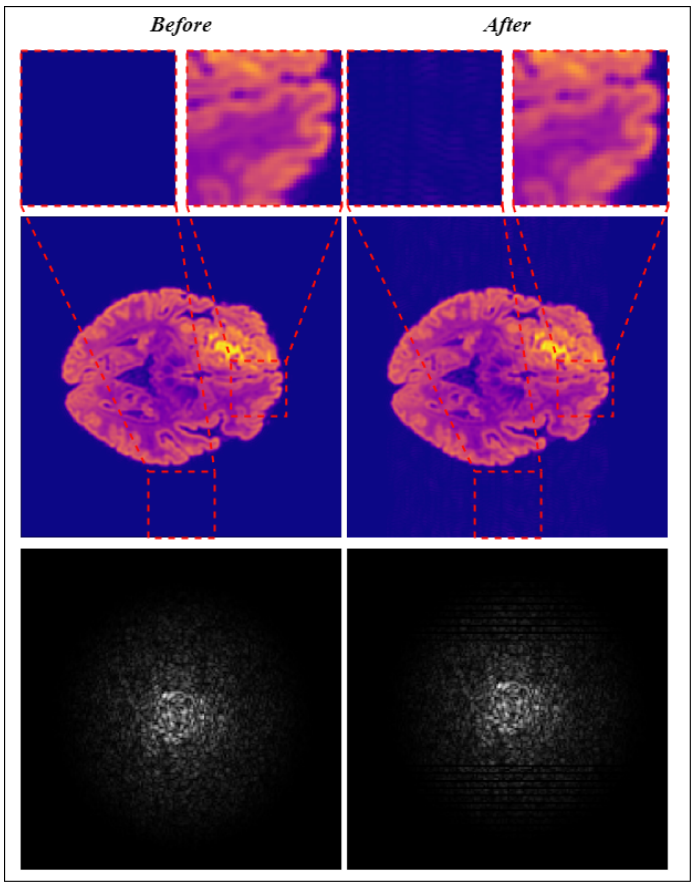

Figure 3.11: Image and k-space after applying our sampling maps.

GRAPPA-Style Reconstruction

For our reconstruction method $\mathcal{T}_{\text{reconstruct}}$, we essentially operate the same method as standard GRAPPA reconstruction mentioned in Section 3.1.3, but only using one coil. First, the ACS region $\mathcal{A}$ is retrieved from the previous step.

$$\mathcal{A}_{z,y,x} = M_{ACS} \cdot K_{z,y,x}$$

From $\mathcal{A}$, we take every odd-indexed line and treat it as missing, adding it to a target set $\mathcal{Y}$. For each such line, we extract the neighbourhood of lines centred on it (based on the kernel size k), which we flatten and store as a training example in $\mathcal{X}$. We use a kernel width of $k = 3$ so each training input contains one line above and one line below the target centre line. We opted for a kernel that encompasses the width of the entire line to compensate for the lack of coil information. Standard GRAPPA relies heavily on inter-coil correlations, so we compensate by using full lines which gives our kernel more context to make predictions. Using entire lines also mean that the predictions are less sensitive to noise of small artefacts than small windows.

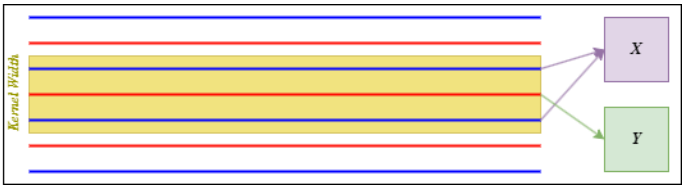

Figure 3.12: Example of $\mathcal{X}$ and $\mathcal{Y}$ given a kernel size of $k = 3$.

Using our training and test set, we train the weights w of a 2D kernel of size $k \times k$ using linear regression.

$$w = \text{argmin}||\mathcal{X} - \mathcal{Y}||_2^2$$

These weights are then applied across the k-space slice to estimate missing lines. For each missing line, the neighbouring lines $\mathcal{N}_{z,y,x}$ are multiplied by the learned w, then scaled using a signal multiplier $\alpha = 1000$, and finally filled in.

$$\hat{K}_{z,y,x} = \begin{cases} \alpha \cdot (\mathcal{N}_{z,y,x} \cdot w), & \text{if } \mathcal{K}_{z,y,x} = 0 \\ \mathcal{K}_{z,y,x}, & \text{otherwise} \end{cases}$$

Overall, our two-step process of undersampling and reconstructing k-space does create more intense aliasing artefacts than normal GRAPPA-acquired images.

Figure 3.13: Comparison of image transformed using our GRAPPA simulation (*left*) and a real GRAPPA image with artefacts (*right*), taken from MRI Master (2023).

3.1.4 Partial Fourier Acquisition Simulation

After simulating GRAPPA acquisition artefacts using $\mathcal{T}_2$, the final stage is to simulate partial Fourier acquisition effects using the transformation $\mathcal{T}_3$. Like in

$\mathcal{T}_2$, this consists of both an undersampling $\mathcal{T}_{\text{truncate}}$ and reconstruction phase $\mathcal{T}_{\text{reconstruct}}$.

$$\mathcal{T}_3(K_{z,y,x}) = \mathcal{T}_{\text{reconstruct}}(\mathcal{T}_{\text{truncate}}(K_{z,y,x}))$$

Hermitian Symmetry

Partial Fourier relies on a concept called Hermitian symmetry. In the image domain, each voxel is real-valued, meaning that they do not contain an imaginary component. However, Fourier transformed real-values are complex numbers, meaning that they contain both a real sa and an imaginary component b:

$$\mathcal{F}(k) = a + ib$$

To retrieve the complex conjugate $F^*(k)$ of a complex number, we simply swap the sign of the imaginary component:

$$\mathcal{F}^*(k) = a - ib$$

If a signal $f(x)$ is real-valued, then its Fourier transform $\mathcal{F}(k)$ satisfies:

$$\mathcal{F}(-k) = \mathcal{F}^*(k)$$

As k-space is the Fourier domain representation of a spatial MRI image, since the MRI image is real-valued (voxel value is equal to the greyscale intensity), its k-space representation obeys Hermitian symmetry. This means that one half of k-space is the complex conjugate mirror of the other half, theoretically meaning that only half of k-space needs to be sampled to form a full image.

k-Space Truncation

Our truncation phase simply creates a binary mask of the same shape as the k-space volume where a fraction of the mask is set to zero at an axis perpendicular to the GRAPPA ACS lines.

$$\mathcal{T}_{\text{truncate}}(K_{z,y,x}) = K_{z,y,x} \cdot M_{fourier}$$

We simulate these sampling strategies perpendicular to each other as this is reflective of what is typically used in MRI machines, as using them parallel to each other can cause an increased level of artefacts () and it also means that the protocols do not overlap.

In real life, GRAPPA is applied first, then partial Fourier reconstruction along orthogonal axes as this is the only way in which Hermitian symmetry holds. When GRAPPA is applied first, its reconstruction means that each line along the partial Fourier axis is fully sampled, despite only containing a partial extent. This allows Hermitian symmetry to be valid, and a reflection to be made.

If N is the length of the axis perpendicular to our GRAPPA ACS direction, in our case $N = 256$, and f represents the fraction of k-space sampled, we chose $f = \frac{5}{8}$, then our binary mask can be defined as follows:

$$M_{fourier} = \begin{cases} 1 & \text{if } i \geq N - (f \cdot N) \\ 0 & \text{otherwise} \end{cases}$$

The resulting k-space can be seen in Figure 3.14

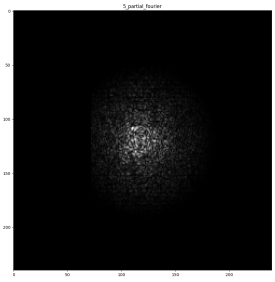

Figure 3.14: Example of k-space being truncated using $\mathcal{T}_{\text{truncate}}$.

The reason that we used the parameter $f = \frac{5}{8}$ is because it matches sampling fractions typically used in partial Fourier (Haldar and Liang, 2022), and it also strikes a balance, where visible artefacts can be seen in the resulting reconstructed image but do not overwhelm the true signal to a great extent (Figure 3.15).

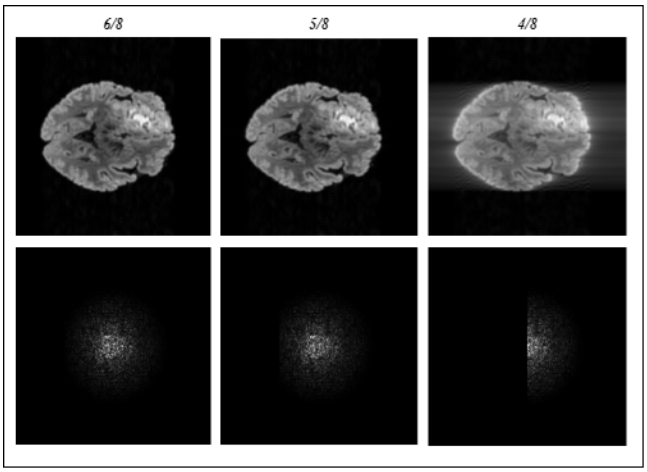

Figure 3.15: Partial Fourier reconstruction with $f = \frac{6}{8}$ (*left*), $f = \frac{5}{8}$ (*middle*), $f = \frac{4}{8}$ (*right*)

Hermitian Symmetry-Based Reconstruction

To perform the reflection, we define K^* as the complex conjugate of K. The Hermitian reconstructed k-space is given by:

$$\hat{K}_{z,y,x} = \begin{cases} K_{z,y,x} & \text{if } M_{z,y,x} = 1 \\ K^*_{z,y,x} & \text{if } M_{z,y,x} = 0 \end{cases}$$

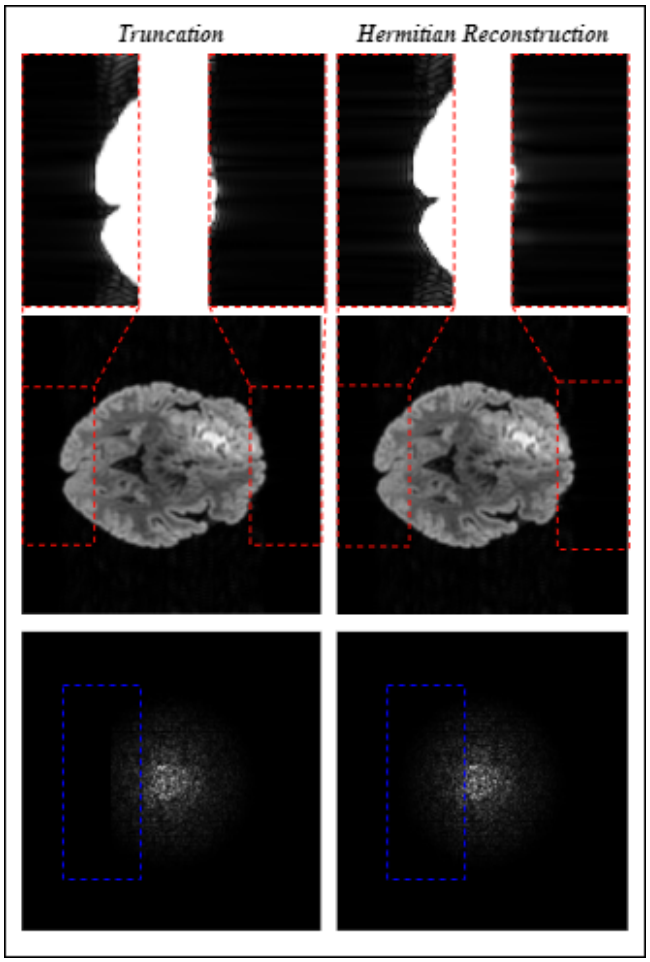

Figure 3.16: Before (*left*) and after reconstruction (*right*).

3.1.5 Noise

The final step of our pipeline is to simulate realistic acquisition noise ϵ from MRI scanners to mimic the noise mentioned in Section 2.2.1. We aim to simulate background noise, arising from factors such as coil resistance or electronic noise, not any structured noise caused by motion, as the presence of structured noise isn't as predictable as it is influenced by factors that are further disconnected from magnetic field strength.

Background Information on Gaussian and Rician Noise

As k-space is complex, it contains a real and imaginary component, evidence suggests that the noise in these components is independent and both follow a zero-mean Gaussian distribution (Gudbjartsson and Patz, 1995). This means that for each voxel, the intensity of the noise (not including the underlying signal) in either the real or imaginary part can be mapped onto a Gaussian distribution. In practice, this means that the noise can be either positive or

negative, with a higher likelihood that the noise intensity is closer to 0 but a small chance that it is higher. We can see an example of what this noise looks like in Figure 3.17

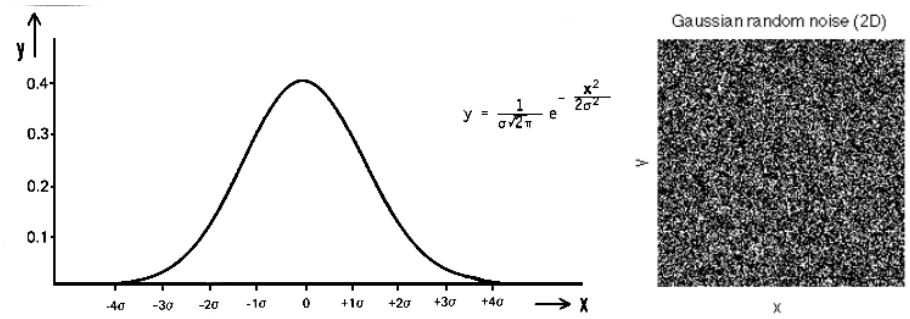

Figure 3.17: A Gaussian distribution (*left*) (Truax, 1999) and the noise map generated using it (*right*) (Stanford University, 2011).

However, when the k-space is converted into the image domain through a Fourier transform, we take the magnitude of both the real and imaginary components to form a single brightness value M.

$$M = \sqrt{(real + n_r)^2 + (imag + n_i)^2}$$

This results in M following a Rician distribution (Gudbjartsson and Patz, 1995), meaning that for each voxel, the intensity of the combined signal and noise can be mapped onto a Rician distribution. The Rician distribution is slightly skewed right and has a mean above zero, meaning that voxel intensities are more likely to be less intense. This is slightly complicated by the fact that areas with different SNR follow slightly different distributions. Additionally, it is worth mentioning that background areas with no signal actually follow a Rayleigh rather than a Rician distribution.

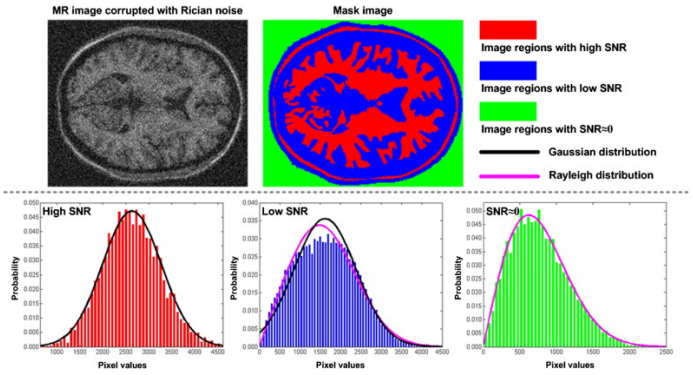

Figure 3.18: Characteristics of Rician noise in MRI. From top-left to top-right: a 2D T1-weighted MR image corrupted by 15% of Rician noise and a mask image separating the background, low-SNR and high-SNR regions. As shown, the Rician noise tends to be Gaussian distributed when SNR is high (bottom-left) and Rayleigh distributed when SNR is close to zero (bottom-right). In low-SNR regions, the Rician noise tends to be neither Gaussian nor Rayleigh distributed (bottom-middle). (Liu *et al.*, 2014)

Our simulation

In our simulation, we chose to add Gaussian noise to the real and imaginary components of k-space so that the spatial image would follow a Rician distribution in magnitude, matching the real-life characteristics mentioned earlier. The noise map ϵ that we added is defined below where $\sigma = 0.005$:

$$\epsilon = \mathcal{N}(0, \sigma^2) + i \cdot \mathcal{N}(0, \sigma^2)$$

As mentioned earlier, to acquire the final image, we add this noise map to the k-space and use a Fourier transform with unitary FFT normalisation to convert the image back into the spatial domain, creating our final image LR.

$$LR = \mathscr{F}^{-1}(\tilde{K} + \epsilon)$$

Figure 3.19 shows an example of a single MRI slice with and without our noise function added in k-space

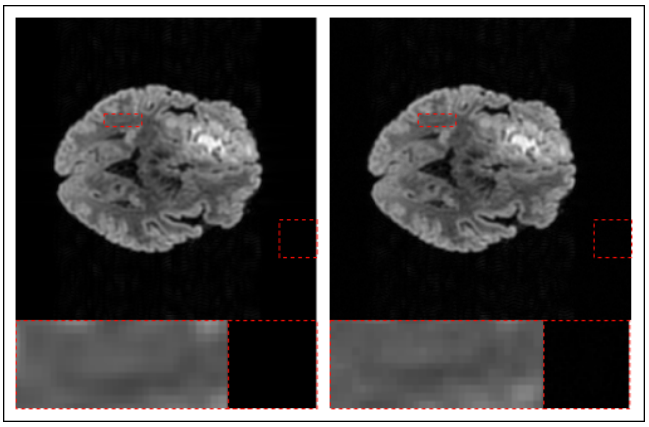

Figure 3.19: Noise before (*left*) an after (*right*) addition of Rician noise.

3.2 Model Architectures

This section outlines the architecture of ESRGAN and FSRCNN, which are the two deep-learning models that we tested for the super-resolution task.

3.2.1 ESRGAN

The ESRGAN model is built upon residual blocks, which are grouped together into Residual-in-Residual Dense Blocks (RRDBs), which are stacked in series to create the generator. It also contains a discriminator which mainly consists of convolution layers. Our architecture was very similar to the architecture used by Ayaz *et al.* (2024).

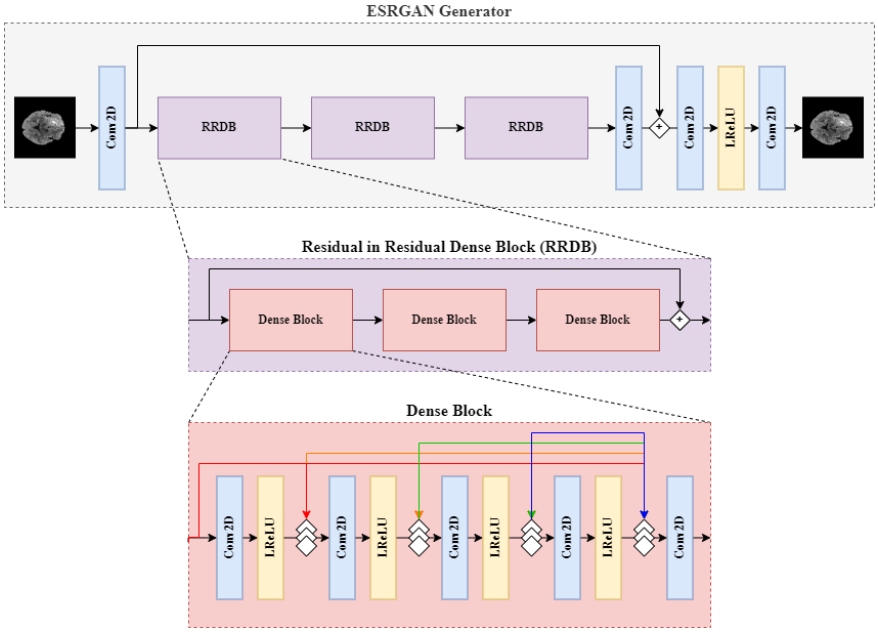

Figure 3.20: Internal Architecture of the ESRGAN generator.

Dense Block

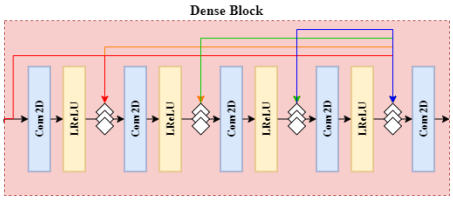

Figure 3.21: Dense Block Diagram

At the lowest level of the ESRGAN structure are dense blocks, which consist of 5 2D convolution layers with 3×3 kernels. The first 3 convolution layers expand the number of channels by 32 and the final layer matches the number of channels to the input tensor. Each of the first 4 convolution layers is followed by a leaky ReLU (LReLU) activation function and the final convolution does not pass through any activation function. Importantly, after each feature map from a convolution layer is passed through the LReLU activation, maps are concatenated along the channel dimension, maintaining the spatial dimensions. The dense block can be seen below where the input $x^{C \times H \times W}$ contains C input

channels:

$$
\begin{aligned}
x_1 &= \text{LReLU}_{0.2}(\text{Conv2D}_{3\times3}^{64\to32}(x)) \\
x_2 &= \text{LReLU}_{0.2}(\text{Conv2D}_{3\times3}^{96\to32}([x, x_1])) \\
x_3 &= \text{LReLU}_{0.2}(\text{Conv2D}_{3\times3}^{128\to32}([x, x_1, x_2])) \\
x_4 &= \text{LReLU}_{0.2}(\text{Conv2D}_{3\times3}^{160\to32}([x, x_1, x_2, x_3])) \\
x_5 &= \text{Conv2D}_{3\times3}^{192\to64}([x, x_1, x_2, x_3, x_4]) \\
y &= x + 0.2 \cdot x_5
\end{aligned}
$$

We set the LReLU slope to 0.2 and also add a final scaling factor to 0.2 based on the implementation in the official ESRGAN repository Wang (2018), based on the model described in the paper Wang *et al.* (2018).

The rationale behind using a dense block is that each block uses the original input and every prior convolution output, not just the final activated layer. In practice, this gives the model more context the model can see low and mid-level features (such as corners, edges, and angles) simultaneously alongside abstract features (like anatomical information). This is beneficial in super-resolution as reconstructing detailed textures requires an understanding the context of textures within a local area and within the entire image simultaneously (Wang *et al.*, 2018).

RRDB

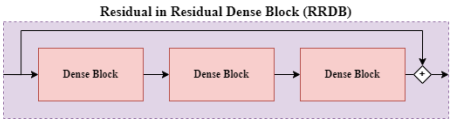

Figure 3.22: RRDB Diagram

An RRDB, also designed by Wang *et al.* (2018), consists of 3 stacked dense blocks together. A residual connection is added around the entire block, meaning that the input to the whole RRDB is added to a scaled output of the RRDB.

$$
\begin{aligned}
x_1 &= \text{RDB}_1(x) \\
x_2 &= \text{RDB}_2(x_1) \\
x_3 &= \text{RDB}_3(x_2) \\
y &= x + 0.2 \cdot x_3
\end{aligned}
$$

RRDBs are effective in super-resolution tasks as residual connections outside the blocks provide a shortcut for gradients during backpropagation, which reduces vanishing gradients. This means that if signal becomes too weak inside an RRDB due to redundant features and the gradients disappear, they won't vanish entirely as the residual connection allows gradients to flow around the dense blocks so that earlier layers can update gradients and continue training (Sharif *et al.*, 2024).

Generator

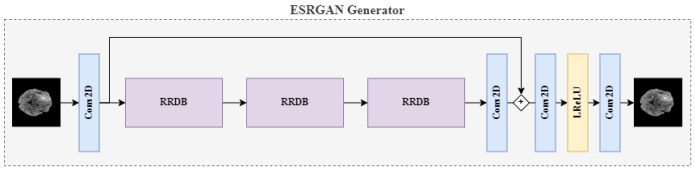

Figure 3.23: ESRGAN Structure

We employ a shallow ESRGAN for our generator, based on implementation by Ayaz *et al.* (2024). Our ESRGAN is 2D, designed to enhance a single LR slice with the same spatial dimensions as the corresponding HR slice. Our model also expects slices to have 1 channel dimension, as voxels in MRI are greyscale rather than RGB (which would require 3 channels). The model also expects input tensors x with spatial dimensions of 256×256. The first operation in the generator is a 2D convolution layer with a 3×3 kernel and 64 output channels generated from 1 input channel:

$$x_1 = \text{Conv2D}_{3\times3}^{1\to64}(x)$$

Feature maps in x_1 are saved for the residual connection later on. Afterwards, x_1 is passed through N consecutive RRDB blocks and finally through another 2D convolution layer with a 3×3 kernel and 64 output channels. We tested both $N = 1$ (the number of blocks that used by Ayaz *et al.* (2024)) and we tried $N = 3$ to try and improve performance.

$$x_2 = \text{Conv2D}_{3\times3}^{64\to64}(RRDB_N(...(RRDB_2(RRDB_1(x_1)))...))$$

We then activated a residual connection between the initial convolution output x_1 and the post-RRDB feature maps x_2.

$$x_3 = x_1 + x_2$$

After the residual connection, we processed x_3 through another identical 2D convolution block, mirroring implementation by Ayaz *et al.* (2024). In a standard ESRGAN, there would be an upsampling layer Wang *et al.* (2018), but our HR and LR have the same spatial dimensions, so upsampling isn't needed.

$$x_4 = \text{Conv2D}_{3\times3}^{64\to64}(x_3)$$

We then pass x_4 through a LReLU activation with a slope of 0.2, mirroring implementation by Ayaz *et al.* (2024).

$$x_5 = LReLU_{0.2}(x_4)$$

x_5 is passed through a final 2D convolution layer which takes the 64 input channels and matches it to the desired output channel size of 1, also containing a 3×3 kernel.

$$y\text{Conv2D}_{3\times3}^{64\to1}(x_5)$$

Discriminator

The discriminator is a separate model which takes an output from the generator (a simulated slice generated from LR or a slice from the HR set) and classifies the image as real or generated. The architecture can be seen in 3.24 which takes inspiration from Ayaz *et al.* (2024), resembling a standard CNN image classifier.

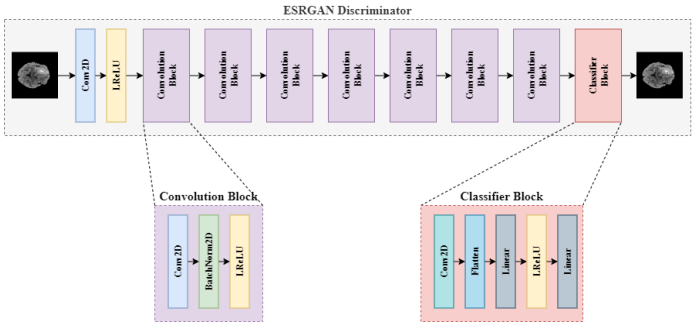

Figure 3.24: ESRGAN Discriminator Structure

The model begins with a 3×3 2D convolution that extracts 64 feature maps. It is then passed through an LReLU activation with slope 0.2.

$$x_1 = \text{LReLU}_{0.2}(\text{Conv2D}^{64}_{3\times3}(x))$$

Feature maps then pass through 7 convolutional blocks, each containing a 2D convolution layer (with a 3×3 kernel), batch normalisation, and LReLU activation with a 0.2 slope. Blocks are divided into two categories:

$$D_i(z) = \text{LReLU}_{0.2}(\text{BN}(\text{Conv2D}^{C_i, s=2}_{3\times3}(z))) \quad \text{(Downsampling block)}$$
$$F_i(z) = \text{LReLU}_{0.2}(\text{BN}(\text{Conv2D}^{2C_i, s=1}_{3\times3}(z))) \quad \text{(Feature extraction block)}$$

Downsampling blocks preserve the number of feature maps C_i but halve the spatial resolution with stride $s = 2$, reducing the resolution while preserving existing features, reducing compute whilst maintaining semantic information. Feature extraction blocks preserve spatial resolution by setting stride $s = 1$ but double the number of features maps $2C_i$, enabling the extraction of deeper semantic information from the image. This structure, used in VGG architecture (Simonyan and Zisserman, 2015), maintains stability whilst training and keeps a reasonable model size. We chose to apply 7 convolutional blocks (4 downsampling and 3 feature extraction), differing from Ayaz *et al.* (2024) as we don't employ a patch-wise approach, focusing on input sizes matching a standard MRI slice (256×256). Ayaz *et al.* (2024) used a patch-wise approach with patch sizes of 64×64. We made the assumption that a higher spatial resolution would naturally require more model depth, so we chose the maximum depth that would fit into our system memory.

$$x_2 = F_3(D_4(F_2(D_3(F_1(D_2(D_1(x_1))))))$$

Finally, the convolution block output is converted into a classification by a classifier block. We first pass the results through an average pooling layer,

resizing feature maps to 4×4 resolution. Feature maps are then flattened into a long 1D vector, and compressed into 100 dimensions using a fully-connected layer. The 100-dimensional vector passes through a LReLU and then a fully-connected layer to create a single classification score.

$$
\begin{aligned}
x_3 &= AdaptiveAvgPool2D_{4,4} \\
x_4 &= Flatten_{8192} \\
x_5 &= Linear_{8192 \to 100} \\
x_6 &= LReLU_{0.2}(x_3) \\
y &= Linear_{100 \to 1}
\end{aligned}
$$

3.2.2 FSRCNN

We also experimented with CNN super-resolution, choosing FSRCNN (Dong *et al.*, 2016) for its competitive performance in super-resolution tasks compared to other CNNs (see 2.5.1). We tested a CNN to validate our assumption that GANs would perform better, since GANs require much longer training.

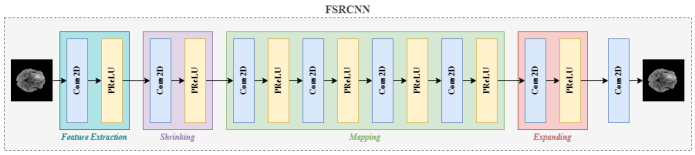

Figure 3.25: FSRCNN Structure

The model is relatively simple, containing five sections:

1. **Feature Extraction**: First the 1-channel image is passed through a 2D convolution layer with a kernel size of 5×5, outputting $d = 56$ channels. It then runs through a Parametric ReLU (PReLU) activation function.

$$x_1 = \text{PReLU}(\text{Conv2D}_{5\times5}^{1\to d}(x))$$

2. **Shrinking**: Next the d-channel feature maps are passed through another 2D convolution layer and PReLU activation. This convolution layer differs as it converts the d-channel feature maps into s features, and also only has a kernel size of 1×1. It is recommended by Dong *et al.* (2016) to set $d > s$ as this step reduces the number of parameters while maintaining a similar PSNR, in this case we set $s = 12$.

$$x_2 = \text{PReLU}(\text{Conv2D}_{1\times1}^{d\to s}(x_1))$$

3. **Mapping**: The next layer is called the mapping layer, which contains m stacked 2D convolution layers (all with 3×3 kernels, and s input and output features) and PReLU activation functions.

$$x_3 = \text{PReLU}(\text{Conv2D}_{3\times3}^{s\to s}(x_2))$$

$$x_4 = \text{PReLU}(\text{Conv2D}_{3\times3}^{s\to s}(x_3))$$

$$x_5 = \text{PReLU}(\text{Conv2D}_{3\times3}^{s\to s}(x_4))$$
$$x_6 = \text{PReLU}(\text{Conv2D}_{3\times3}^{s\to s}(x_5))$$

The specific combination $s = 12$, $d = 56$, and $m = 4$ is also tested in the paper and achieves the best trade-off between performance and parameters.

4. **Expanding**: Afterwards, the number of channels are expanded from s back to d using an expanding layer:

$$x_7 = \text{PReLU}(\text{Conv2D}_{1\times1}^{s\to d}(x_6))$$

5. **Deconvolution**: Finally, the output is de-convoluted back into a final single-channel output image:

$$x_8 = \text{PReLU}(\text{Conv2D}_{5\times5}^{s\to 1}(x_7))$$

3.3 Loss Functions

We used a composite loss function for our generator L_{gen}, which is the weighted sum of multiple loss functions designed to penalise different aspects of the image:

$$L_{gen} = \lambda_1 L_{pixel} + \lambda_2 L_{perc} + \lambda_3 L_{edge} + \lambda_4 L_{fourier} + \lambda_5 L_{style} + \lambda_6 L_{adv}$$

We also experimented with each of the loss functions individually in combination with the adversarial loss L_{adv}. To train the discriminator, we used a separate loss function L_{disc}.

Pixel Loss

We use a simple mean absolute error (MAE) to deduce the pixel loss between images. Pixel loss measures how different the intensities of each pixel are from SR to HR, without targeting specific features like edges or structures, representing a stable measurement for how different the two images are. MAE is calculated by taking the absolute difference between SR and HR, averaging these differences across all pixels:

$$L_{pixel} = \frac{1}{N}\sum_{i=1}^{N} |SR_i - HR_i|$$

We chose MAE over mean squared error (MSE), as MSE often produces blurrier GAN images (Ledig *et al.*, 2017), whilst MAE produces sharper images (Zhao *et al.*, 2017). We included this loss following Ayaz *et al.* (2024), who weighted it at $\lambda_1 = 0.3$ with positive results on a similar super-resolution task. This weighting was achieved using an ablation study to empirically determine the weight combination that maximizes sharpness of edge pixels, where the other two loss functions were perceptual loss $\lambda_2 = 1.0$ and edge loss $\lambda_3 = 0.7$. Additionally, L1 loss is a standard benchmark in super-resolution, so not including it would limit comparability.

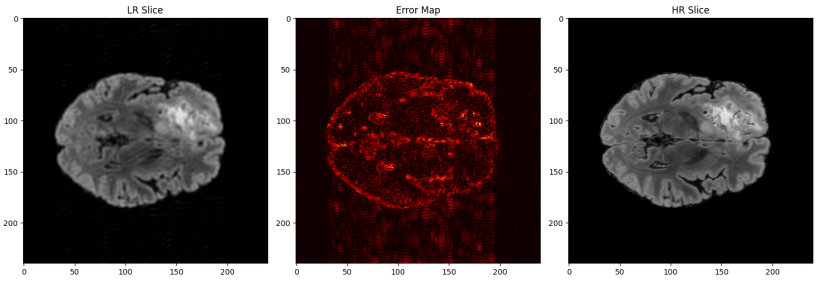

Figure 3.26: Pixel loss intensity map (*middle*) between LR (*left*) and HR (*right*).

Perceptual Loss

The perceptual loss function L_{perc} is included to measure the semantic differences between SR and HR by generating and comparing feature maps for both images using a pre-trained VGG19 CNN. These feature maps represent higher-level structures, such as the global shape of the brain, different tissue textures, and anatomical regions.

VGG19 expects a 3-channel image, but both SR and HR are 1-channel images, so we first stack each image across the channel dimension:

$$\widetilde{SR} = [SR, SR, SR], \quad \widetilde{HR} = [HR, HR, HR]$$

Each channel c in $\widetilde{SR}$ and $\widetilde{HR}$ is then normalised using the normalisation function v_c to match the domain of ImageNet so that any inputs to the model lie within the same distribution as the data it was pre-trained on.

$$v(x)_c = \frac{x_c - \mu_c}{\sigma_c}$$

Where:

$$\mu = [0.485, 0.456, 0.406]$$
$$\sigma = [0.229, 0.224, 0.225]$$

So:

$$SR_{norm} = v(\widetilde{SR}), \quad HR_{norm} = v(\widetilde{HR})$$

The normalised inputs SR_{norm} and HR_{norm} are then passed through VGG19 to extract high-level feature maps. The MAE between the two feature maps gives us the perceptual loss L_{perc}.

$$L_{perc} = \frac{1}{N} \sum_{i=1}^{N} |VGG(SR_{norm}) - VGG(HR_{norm})|$$

VGG19 was pre-trained using ImageNet-1K, meaning that the model's weights are calibrated using the ImageNet-1K dataset, which can be broken into 12 categories: mammal, bird, fish, reptile, amphibian, vehicle, furniture, musical instrument, geological formation, tool, flower, and fruit (Deng *et al.*, 2009). Whilst this is a domain mismatch as VGG19 is designed to generate feature

maps for 3-channel natural images as opposed to 1-channel MRI scans, CNNs trained with ImageNet are shown to generalise well to new domains as they can capture compositional structures, texture, and motifs, which tend to be the building blocks of any image type regardless of domain (Yosinski *et al.*, 2014). Additionally, this model was used by Ayaz *et al.* (2024) inside their composite loss function with a weighting of $\lambda_2 = 1.0$, which specifically focused on MRI super-resolution. Therefore, we didn't fine-tune our VGG19 model with MRI data, though this could be a potential improvement. Perceptual loss was also proposed by Wang *et al.* (2018) and was found to provide sharper edges and more visually pleasing results, making it valuable given the problem similarity.

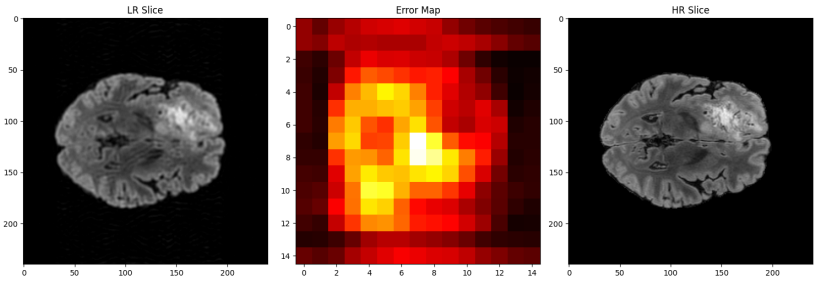

Figure 3.27: Perceptual loss intensity map (*middle*) between LR (*left*) and HR (*right*).

Edge Loss

To mitigate blurriness caused by pixel loss we included edge loss, which penalises differences in edge structure, leading to images with sharper boundaries. As mentioned earlier, Ayaz *et al.* (2024) uses edge loss weighted at $\lambda_3 = 0.7$, indicating its suitability for the task.

Let $HR \in^{W \times H}$ denote a high-resolution ground truth 3T image, and $SR \in^{W \times H}$ denote the super-resolved image generated by the model, where W represents the width and H represents the height of the image (256×256). We first compute the horizontal and vertical gradient magnitudes of HR. The gradient magnitude measures how quickly the intensity of an image is changing at any particular point. For example, flat areas such as white and grey matter have a smaller gradient magnitude, whilst boundaries between different matter types (white, grey, CSF and air) would have a high gradient magnitude.

The gradient magnitudes are computed via the Sobel operator, which consists of two kernels that approximate the first-order derivatives in the horizontal x and vertical y directions:

$$K_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}, \quad K_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Gradient magnitude maps in the horizontal G_x and vertical G_y directions are computed by a two-dimensional convolution operation:

$$G_x = HR * K_x, \quad G_y = HR * K_y$$

These are then combined to form a single combined gradient map G:

$$G = \sqrt{G_x^2 + G_y^2}$$

The loss function then computes the mean absolute error between SR and HR over the entire batch of size N, weighted by the edge magnitude map.

$$L_{edge} = \frac{1}{N} \sum_{w=1}^{W} \sum_{h=1}^{H} G_{w,h} \cdot |SR_{w,h} - HR_{w,h}|$$

This ensures that differences in intensities near edges are given a higher weighting, encouraging the model to prioritise the construction of boundaries, leading to a sharper super-resolved image.

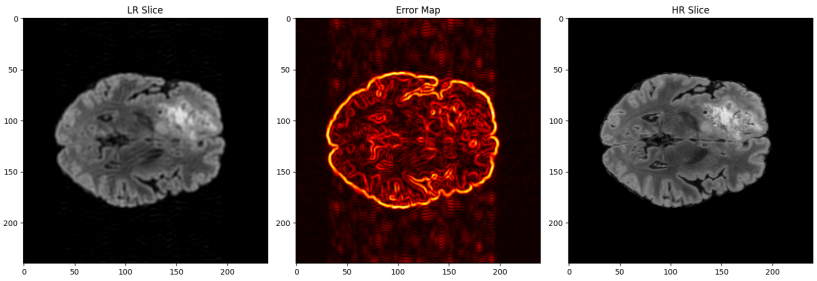

Figure 3.28: Edge loss intensity map (*middle*) between LR (*left*) and HR (*right*).

Fourier Loss

The Fourier loss function $L_{fourier}$ was included as an attempt to penalise differences in edge structure, essentially serving as a drop-in alternative to L_{edge}. It compares the MSE between Fourier transforms $\mathscr{F}$ of SR and HR.

$$L_{fourier} = \frac{1}{N} \sum_{i=1}^{N} (\mathscr{F}(SR_i) - \mathscr{F}(HR_i))^2$$

Instead of penalising differences in the spatial domain, Fourier loss penalises major differences in k-space between SR and HR. We intended this loss function to capture how textures and structures are distributed across frequencies, for example, low frequency content like global structure is positioned towards the middle of k-space and high frequency content such as specific textures are positioned at the edges. As pixel loss and perceptual loss have a tendency to align intensities well and match the general structure of an image (lower frequency content), but miss fine textures and edges (high frequency), it can lead to blurrier content. In theory, Fourier loss could allow a different perspective for the model to see higher frequency content, as if the magnitudes at the edges of k-space are very different, they would be penalised in the same way that lower frequency content is penalised. This could lead to clearer grey-white matter boundaries or more accurate lesion boundaries, acting as another method of edge loss. Additionally, our simulation pipeline was a series of k-space transformations, so we thought it would be interesting to see if the model

could learn these transformations. If it can, then it leads to another question as to whether there exists a consistent k-space mapping that translates 1.5T MRI scans from one machine to equivalent 3T scans from another, which could be investigated using real-world data.

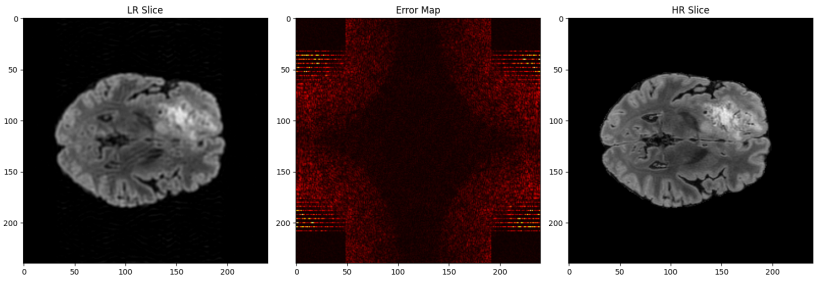

Figure 3.29: 3D Fourier difference map (*middle*) between an LR scan (*left*) and a HR scan (*right*).

Style Loss

We experimented with a style loss function designed to penalise differences in style, which can be thought of as a global texture. While local texture refers to repeating patterns in a specific area, global texture involves capturing these patterns across the entire image, similar to an image filter. This concept can be seen visually in Figure 3.30. We experimented with this type of loss as we considered the possibility that mapping images from 1.5T to 3T could be seen as a style change. The spatial resolution of both images is the same, so any changes would be in contrast, noise characteristics, and artefacts, which could possibly be better captured by global texture rather than metrics like pixel-wise content. Therefore, the model could learn to align these global textures, effectively translating the "style" of 1.5T images into 3T images.

Figure 3.30: Images that combine the content of a photograph with the style of several well-known artworks. Taken from Gatys *et al.* (2015).

The style loss between two images is calculated by comparing the Gram matrices of feature maps generated using VGG19. We stack the 1-channel LR and HR into 3-channel images, then normalise and pass into VGG19 in the exact same way as Section 3.3. This leaves us with:

$$VGG(SR_{norm}), \quad VGG(HR_{norm})$$

We then reshape these feature maps into Gram matrices:

$$G_{SR} = \frac{VGG(SR_{norm}) \cdot VGG(SR_{norm})^T}{C \cdot H \cdot W}$$

$$G_{HR} = \frac{VGG(HR_{norm}) \cdot VGG(HR_{norm})^T}{C \cdot H \cdot W}$$

Gram matrices represent how similar different feature maps are to each other. If two feature maps often activate together then the Gram matrix is large. To achieve a loss value, we compute the MAE between the two Gram matrices:

$$L_{style} = \frac{1}{N} \sum_{i=1}^{N} |G_{SR_i} - G_{HR_i}|$$

Figure 3.31 shows the style loss heatmap between a slice.

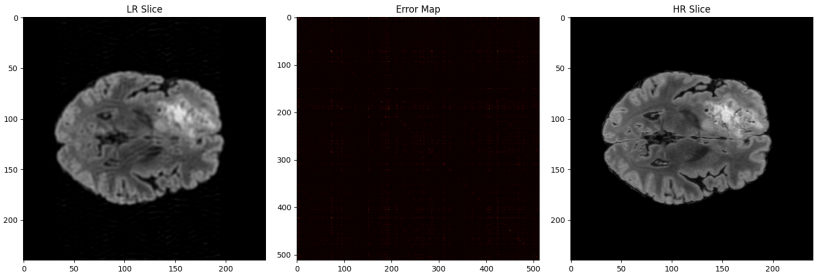

Figure 3.31: Style loss map between an LR scan (*left*) and a HR scan (*right*).

Adversarial Loss

The key component that makes a GAN work differently to other networks it the adversarial loss function Ł$_{adv}$. This enables the generator and discriminator to continuously compete with each other to generate a realistic fake image and determine if an image is real or fake respectively. We first pass the LR slices into the generator G at its current state to form super-resolved slices. We then pass those super-resolved slices into the current discriminator D, which generates a scalar value for each slice, the higher the value the more real-looking the image. Afterwards, we average these discriminator scores over the entire batch and take the negative, so that the generator is penalised when the discriminator easily identifies its outputs as fake.

$$L_{adv} = -\frac{1}{N}\sum_{i=1}^{N} D(G(LR_i))$$

We set the weighting of this loss function to $\lambda_6 = 0.001$ following the ablation study done by Ayaz *et al.* (2024), who also used a WGAN discriminator and a similar composite loss function structure, identifying this weighting to be optimal for maximising edge sharpness.

Wasserstein GAN with Gradient Penalty

To train the discriminator we use the loss function L_{disc}. This loss function first runs the generated SR and real HR sets and runs them through the discriminator at its current state, it then averages the scores given by the discriminator for each set.

$$\mu_{SR} = \frac{1}{N}\sum_{i=1}^{N} D(SR_i), \quad \mu_{HR} = \frac{1}{N}\sum_{i=1}^{N} D(HR_i)$$

The mean scores of real images μ_{HR} should naturally be much higher than μ_{LR} as the discriminator is more likely to understand these inputs to be real. Therefore, we aim to maximise the difference between scores, which is equivalent to maximizing the Wasserstein distance:

$$\mu_{HR} - \mu_{SR}$$

This is equivalent to minimising:

$$\mu_{SR} - \mu_{HR}$$

Therefore, our loss function L_{disc} uses the following:

$$L_{disc} = \mu_{SR} - \mu_{HR} + \lambda_{gp} \cdot L_{gp}$$

A weighted gradient penalty $\lambda_{gp} \cdot L_{gp}$ is also added to enforce the Lipschitz constraint required for WGANs. This attempts to enforce the discriminator's gradients with respect to its input to have a norm close to 1, preventing it from becoming too confident. We set the weight of this to the recommended value of $\lambda_{gp} = 10$ (Gulrajani *et al.*, 2017).

We use the Wasserstein GAN (WGAN) over a standard GAN as it improves training stability and removes mode collapse (generator producing outputs that are repetitive or near-identical), which are common issues when training GANs. Using WGANs also makes the training process more forgiving when changing model architecture or learning rate, allowing us to conduct experiments with less risk of the model breaking during training (Arjovsky *et al.*, 2017). This is something we aim to minimise as we operate under resource and time constraints, meaning that we need to make the most out of our limited training runs. The WGAN was also used by Ayaz *et al.* (2024) with positive results, showing its appropriateness for the problem at hand.

Calibration of Loss Function Weights

When combining multiple loss functions in the composite generator loss function L_{gen} we had two methods for deciding the appropriate weighting of $\lambda_1, \lambda_2, \ldots, \lambda_5$. To find an initial weighting that results in a balanced composite loss function, we created an untrained generator and generated a set of super-resolved images SR using the entire HR dataset. We assume that any output from the untrained generator is always noise as the generator has had no chance to calibrate its internal weights to fit HR. We then compare SR and HR using our composite loss function, and log the magnitude values for each individual loss function, plotting them on a line graph to visualise the difference in magnitudes between these weights (example graph shown in Figure 3.32).

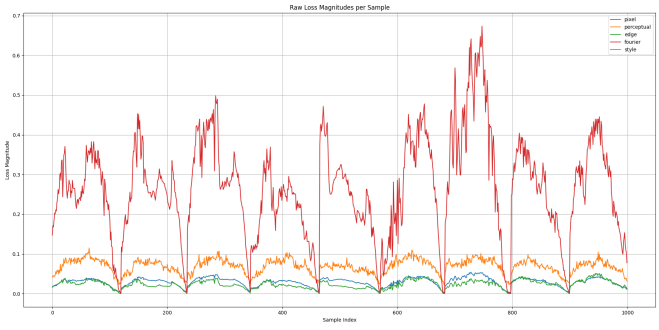

Figure 3.32: Magnitudes of loss after first pass.

This helps us to calibrate the magnitudes of each loss function so that no one loss function dominates training from the outset. By visualising the magnitudes of each loss term at the initial stage where no specific learning has actually

occurred, we can calibrate the weights so that each component contributes an equal amount to the final loss L_{gen}.

Once training has started, we also add the ability to log individual loss magnitudes so that we are aware if one loss function appears to have an equal magnitude to the others at the initial stage, but dominates the composite loss after a certain duration of training, in which case we can scale the weighting accordingly.

3.4 Training Procedure

The training procedure involves selecting, preprocessing, and augmenting the BraTS dataset. GAN training involves pre-training the generator, then using both the generator and discriminator to do adversarial training.

3.4.1 Data Selection

As mentioned in Section 2.6, we used all T_2-FLAIR data in the BraSyn which is a subset of the BraTS dataset. This included a predefined 60 : 10 : 30 training/validation/test split containing a total of 1251 scans in the training set, 219 scans in the validation set, and 570 in the test set. Unfortunately, we do not have access to the test set so we split the default validation set in half, leaving 110 scans in the validation set, 109 in the test set, and keeping the training split unchanged. This creates a train/validation/test split of 1251 : 110 : 109. We then split these volumes into slices along the axial plane to obtain 2D slices for training.

We opted to use this split for a few reasons. Firstly, we wanted to preserve the original splits as much as possible, which meant that we left the training split unaltered to preserve any statistical stratification on factors like scanner type, and in case the creators of this dataset had intentionally selected certain scans for each split. We also wanted to expose the model to the maximum possible training data diversity to ensure that it was not learning scan-specific characteristics. While we train the model a greater than normal number of slices, these slices only come from 1251 scans, which is not exceptionally diverse, so we did not want to risk lowering it any further as it could impact the performance of the model. That being said, Ayaz *et al.* (2024) only used a maximum test set size of 20 scans and a maximum training size of 60 scans, both of which are significantly smaller than any of our set size, suggesting that strong performance and generalization are still achievable even with a relatively small number of unique scans. We opted to split the validation set in half to form validation and test sets of equal size so that we have sufficient validation samples to avoid overfitting while tuning hyperparameters, and enough scans to reliably report generalised performance. It is common for validation and test splits to be of equal size, so this approach aligns with standard practice. In an optimal scenario, we would use k-fold cross validation to train and evaluate the model on the entire dataset, but k-fold cross validation multiplies the training time by k, and due to time and resource constraints this would have been unfeasible.

3.4.2 Data Preprocessing and Augmentation

After generating our LR dataset using the pipeline from Section 3.1, we split both the HR and LR data into slices along the axial plane. Each scan has the shape $240 \times 240 \times 155$ but the outermost slices were mostly blank (containing only zeros) or contained a very small amount of useful information (Figure 3.33).

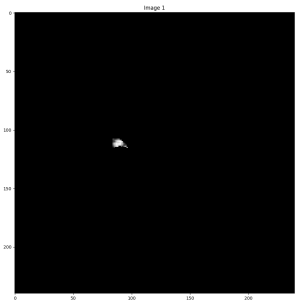

Figure 3.33: Example of slice at index 5 (*BraTS-GLI-00000-000-t2f*) with very little information.

Therefore, we removed any slices with an index below 10 or above 145 leaving us with scans of shape $240 \times 240 \times 135$. This was mainly to condense the dataset to reduce training time, however including blank slices is also bad practice as there is no useful signal for the model to learn meaning that any metrics such as MSE or PSNR can be artificially improved. To take this one step further, we also removed all slices with a standard deviation below 1×10^{-5}, and all slices where their minimum value is equal to their maximum.

We then normalise each slice x into $\hat{x}$ using min-max normalisation to rescale the data values to a standard range as neural networks are generally sensitive to the scale of data and work best with normalised inputs. Leaving the slices unnormalised may lead to numerical instability and may cause exploding or vanishing gradient issues.

$$\hat{x} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

We then upscale each slice from 240×240 to 256×256. This increases the size of the model and impacts training time negatively, but we do this as 256 is a power of 2, which aligns well with the convolution layer architectures that use strides and pooling. Additionally, increasing the size of the input gives the model a large canvas to learn and enhance fine details. If we were to change the size of the slices to align with a power of 2, then the closest options available would be to downscale to 128×128, which would limit the canvas, or to upscale to 512×512, which would not run properly with our model on our hardware.

We then applied a series of augmentation steps. Each slice would have a 50% chance of being flipped horizontally, a 50% chance of vertical flipping, and a 50% chance to rotate by either 90°, 180°, or 270°(selected randomly). These probabilities are independent of each other. Augmentation empirically improves model performance on unseen data as models are trained to recognise the same object under different transformations, and real-world images are also likely to appear in different orientations.

Finally, we shuffle the training dataset to prevent the model from learning any sequence patterns, leaving us with a training dataset with approximately 168885 slices of 256×256 images, and validation and test sets of approximately 14850 and 14715.

3.4.3 Pre-training Phase

GAN training is difficult as the generator and discriminator converge at different speeds, with the discriminator normally converging faster. If the discriminator becomes too accurate and overpowers the generator, then it assigns a near-zero score to all fake images, making the generator loss flat and gradients disappear. Without gradients, the generator can't improve while the discriminator continues to improve, meaning that any future change the generator makes is easily classified by the discriminator and reverted, effectively stalling the entire learning process (Ham *et al.*, 2020).

To remedy this, we pre-trained the generator using pixel loss for several epochs before starting adversarial training. If the generator can perform relatively well before the discriminator sees the images, this helps the discriminator to focus more on texture discrimination, because most other structural aspects of the super-resolved image are already resolved well. Pre-training is designed to let the generator learn most of the super-resolution process, whilst adversarial training can be thought of as fine-tuning. This approach is also used by Wang *et al.* (2018) in the original ESRGAN paper, indicating its applicability.

To pre-train, we use the Adam optimiser with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and an initial learning rate of $\eta = 1 \times 10^{-4}$, following Wang *et al.* (2018). We then performed training over either 3 or 4 epochs depending on the size of the dataset, waiting until the MAE loss was stagnant for 2 epochs before performing a visual sense check by running the model on a subset of LR slices and moving onto discriminator training. This same loop is also used to train our FSRCNN model.

3.4.4 Adversarial Training

After pre-training, we perform adversarial training using the same Adam optimiser for generator optimisation and an identical Adam optimiser for the discriminator. For each epoch, we first train the discriminator by generating a set of fake SR data using the generator, using L_{disc} in Section 3.3 to update the discriminator weights. We then run the updated discriminator on the SR data to generate a set of predictions to be used in L_{adv} alongside any other loss functions mentioned in Section 3.3 to form the composite loss function L_{gen}. We then compute the gradients using L_{gen} and update the generator weights accordingly. We run the adversarial loop for 20 epochs with an early stopping criterion based on SSIM with a patience of 10 epochs.

3.5 Evaluation Procedure

To evaluate model performance, we use standard methods such as Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) to evaluate visual similarity of the

reconstruction. We also employ custom methods to quantify the reconstruction of matter and tumour regions using pre-trained models. Tests are run each epoch on the validation set for tuning, and once on the unseen test set for final evaluation. Our preference would have been to use k-fold cross-validation to use the entire dataset for both training and evaluation, but this was not possible due to time and resource constraints.

3.5.1 PSNR

PSNR is a metric designed to quantify the quality of a reconstructed image compared to a reference image in terms of pixel-level difference. This is calculated using the following:

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX^2}{MSE} \right)$$

- MAX: Maximum pixel magnitude.
- MSE: Mean Squared Error between the original and the reconstructed image.

Generally, a higher PSNR (30dB+) indicates a better reconstructed image, but PSNR relies on the numerical difference between two images, not taking into account human perception. For example, all images in Figure 3.34 have the same MSE score and therefore same PSNR, but some are far more legible than others due to human perception. For example, sparse spikes of noise with high magnitudes (f) make for a far clearer image than a blurring effect (c).

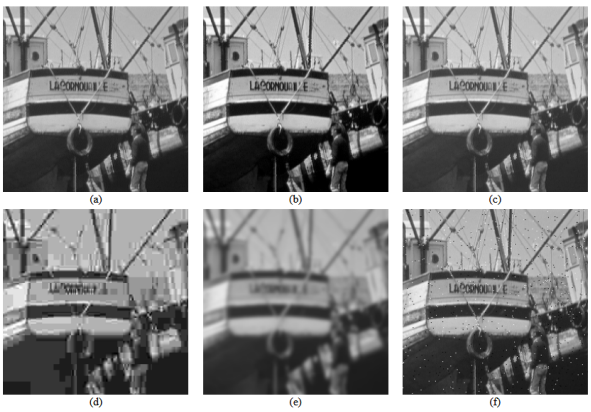

Figure 3.34: Comparison of "Boat" images with different types of distortions, all with MSE = 210. (a) Original image (8 bits/pixel; cropped from 512×512 to 256×256 for visibility); (b) Contrast stretched image, MSSIM = 0.9168; (c) Mean-shifted image, MSSIM = 0.9900; (d) JPEG compressed image, MSSIM = 0.6949; (e) Blurred image, MSSIM = 0.7052; (f) Salt-pepper impulsive noise contaminated image, MSSIM = 0.7748. Taken from Wang *et al.* (2004).

In an MRI scan, we would prioritise the reconstruction of lesions and matter boundaries, as well as certain textures for diagnostic purposes. PSNR has the potential to fall short by giving a high score to blurry images without clear lesion

boundaries but similar overall pixel intensity. These pitfalls do not invalidate the PSNR metric, as it can still be used as a baseline for initial comparison and is widely used, allowing us to compare our results to other works, but it is important to be aware of these limitations.

3.5.2 SSIM

A method that attempts to overcome the limitations of PSNR is SSIM, which compares a reconstructed image y and reference image x based on three factors that are important for human perception: luminance l, contrast c, and structure s. The structural aspect is especially important as human perception can easily compensate for non-structural distortions.

$$SSIM(x, y) = l(x, y) \cdot c(x, y) \cdot s(x, y)$$

The luminance comparison is calculated using the mean intensities for x and y (μ_x and μ_y) and a constant c_1:

$$l(x, y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1}$$

The contrast comparison is calculated using the standard deviations for x and y (σ_x and σ_y) and a constant c_2:

$$c(x, y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2}$$

The structure comparison is calculated using the covariance between x and y (σ_{xy}) and another constant c_3:

$$s(x, y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3}$$

The SSIM is computed for each location using a sliding window, where the final SSIM score is equal to the mean of all local SSIM values. If the two images have similar brightness, contrast, and structure, then SSIM approaches 1. If they differ significantly then SSIM will approach 0.

SSIM is generally better than PSNR at quantifying blurring levels (Anchuri, 2011) and can quantify noise somewhat better (Wang and Bovik, 2009). We directly degrade the HR image with noise and blurring, so we expect the super-resolution process to reduce both, making SSIM a suitable evaluation metric. These factors along with the structural quantification make SSIM a better approach for measuring clinically useful reconstructions, as inaccurate anatomical boundaries are more heavily penalised in SSIM than PSNR.

One drawback of SSIM is that it performs poorly when the mean intensity μ or variance σ is close to zero, which is common in MRI background regions, leading to unstable SSIM measurements. Additionally, when μ or σ is too high, distortions are ignored. Even though SSIM is better at capturing edge

similarity, when it computes the similarity around each pixel, the window it uses includes both the edge and surrounding similar areas. This means that if a surrounding region is very similar, the SSIM score for pixels near the edge gets pulled up, falsely suggesting high similarity even if the edge is poorly reconstructed (Pambrun and Noumeir, 2015).

3.5.3 LPIPS

LPIPS is a metric designed to align closer to human perceptual quality than both SSIM and PSNR by comparing the similarity of deep feature maps generated using a pre-trained CNN. As LPIPS does not use pixel values directly, instead using deep features, it is not susceptible to issues caused by high and low μ or σ values, and can perceive blur and noise much better than PSNR. It also doesn't use local windows like SSIM, so it overcomes issues scoring edges close to similar regions (Zhang *et al.*, 2018).

Figure 3.35 shows how to compute a distance score d_0 between two images x and x_0, given a pre-trained CNN F. In our evaluation, we use AlexNet as our pre-trained CNN.

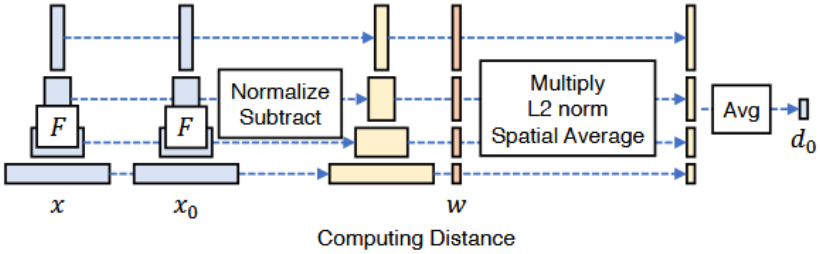

Figure 3.35: *Computing distance from a network*, taken from Zhang *et al.* (2018).

In the context of MRI comparison, LPIPS has the ability to provide a better metric to measure high-level structural and textural differences as opposed to pixel-level errors calculated using SSIM and PSNR, meaning that it could lead to scores that align closer to clinical relevance. However, LPIPS is not fine-tuned on MRI data, and as such has no specific anatomical knowledge. This means that the deep features do not align exactly with clinically meaningful variations. Overall, PSNR, SSIM, and LPIPS are useful in combination with each other to build a combined view on what areas of the reconstruction are similar.

3.5.4 Brain Tumour Segmentation

We wanted to evaluate the performance of our super-resolution in a clinical capacity, so we first focused on investigating whether our super-resolution models can improve the classification of tumour or lesion regions. To do so, we first ran all three datasets (LR, SR, and HR) through the brain-tumour segmentation model DeepSeg. We chose to use DeepSeg as it specialises in FLAIR scans, not requiring multiple modalities to work Zeineldin *et al.* (2020). Whilst different pre-trained DeepSeg models exist, we opted for their UNet architecture due to

its faster prediction time than other variants. DeepSeg generates a map for each slice containing labels for each voxel:

- **0**: Non-Tumour Area
- **1**: Tumour

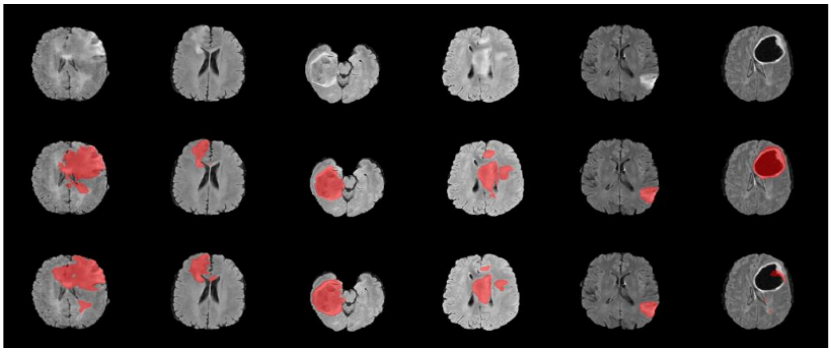

Figure 3.36: Example segmentations from DeepSeg UNet (*bottom*) compared with ground truth (*middle*) Zeineldin *et al.* (2020).

We compute the Dice coefficient between each set A and the ground truth B, using the following equation:

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

Where $|A \cap B|$ represents the number of pixels correctly predicted as a tumour and $|A|$ and $|B|$ are total predicted and actual tumour pixels. We use the Dice score as it directly measures the spatial overlap between classes and it also discounts true negatives (in this case the background) which would inflate the score. We first run the HR data through LATUPNET to generate a set of "ground truth" tumour segmentation maps (note that we did not have access to true ground truth segmentation maps for validation data) which are treated as B. Then we do the same with both the LR and SR data. This leaves us with a list of performance scores:

$$Dice_{LR} \quad Dice_{SR}$$

We then use the Wilcoxon signed-rank test to check if $Dice_{SR}$ is significantly better than $Dice_{LR}$. The Wilcoxon signed-rank test compares two groups to determine if one tends to have higher values than another. Since the LR and SR Dice scores are computed from the same slices and ground truths, they form a paired dataset, making the Wilcoxon test the appropriate choice.

To perform the test, we compute the differences between each pair of Dice scores ($Dice_{SR} - Dice_{LR}$), discard any zero differences, and rank the absolute values of the differences, with each rank still retaining its sign of the original difference. We then sum the negative and positive signed ranks separately into W_- and W_+, where the test statistic W is the lower value. The significance can be determined using this W value and a reference table.

We use Wilcoxon signed-rank as it does not assume that either set are normally distributed, which is safer as we expect our Dice scores to cluster near 1.0 for good models but might be more spread out for worse ones. It also provides an answer to our investigation as to whether our model can improve the classification of tumour regions by telling us if inputs enhanced by our model tend to have more accurate maps than unenhanced input. The experiment runs as a spiritual alternative to an expert score, where DeepSeg acts as a proxy for clinical assessment and is assumed to be reliable at producing tumour segmentations.

3.5.5 Brain Matter Segmentation

The second way we measure the clinical performance of our models is by measuring how well anatomical matter is mapped. We do so in a similar fashion to Section 3.5.4 by running LR, SR, and HR sets through the segmentation model FSL-FAST. We chose FSL-FAST as it is widely used in clinical and research settings, with the original publication by Zhang *et al.* (2001) cited over 5800 times in academic literature (as of July 2025). This produces, for each slice, three binary maps indicating the probability of presence (ranging from $0-1$) of one of the following specific brain matter types at each voxel:

- White Matter
- Grey Matter
- Cerebrospinal Fluid

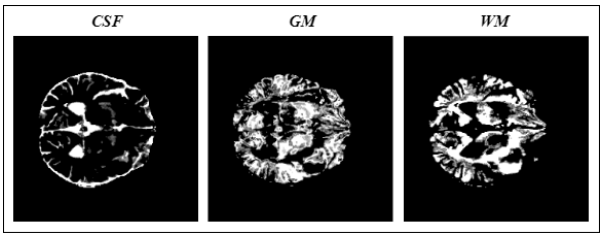

Figure 3.37: Image of FSL-FAST output.

Unfortunately, the BraTS dataset does not come with ground truth matter maps either, so we treat the HR maps as ground truth for the purpose of this method. For each matter map, we compute the dice scores between LR with reference to the HR ground truth, and do the same with SR giving us the dice scores:

$$Dice_{LR} \quad Dice_{SR}$$

We then use these to compute the Wilcoxon signed-rank, which tells us if using our super-resolution model produces significantly better matter labels than a set not using our model.

3.6 General Methodological Approach

Given our limited resources, we designed experiments under a constrained setup to identify the solution with best performance. We aimed to learn the following objectives:

1. Finding the appropriate model architecture and depth to understand the data without using unnecessary compute.
2. Finding appropriate hyperparameters that maximised performance of the model.
3. Using a loss function that penalised the most clinically useful factors when training.

We began with a restricted training dataset size that reduces training time. Using this subset, we first explore model architectures in two senses: whether the type of model is useful (i.e. whether a GAN actually improves performance or if just using CNN would suffice), and what the appropriate model depth is that balances performance and efficiency. During this, the loss function (set to a proven standard) is to be kept constant, but hyperparameters for each model can then be explored. We explore architectures before loss functions because it is important to verify whether the model has the capacity to learn meaningful improvements. If it does then the loss function can be thought of as fine-tuning the model outputs to be more clinically relevant, however loss functions cannot improve the performance in a meaningful way if the model is too shallow to capture relevant features in the slices. Once we have explored loss functions, we can then train any final models on the full dataset, with prior knowledge of appropriate models, hyperparameters, and loss functions. We acknowledge that loss functions, models, and hyperparameters are not independent and can affect each other, but we do not have the time to explore every combination, so we try to use a logical order.

3.7 Computational Resources

All training and evaluation were conducted on a local machine with the following specifications:

- **CPU**: *Intel Core i5 4690K*
- **GPU**: *NVIDIA RTX 3060 (12GB VRAM)*
- **RAM**: *16GB DDR3*

This setup was adequate for the task, however, caused some limitations on model size and importantly time taken to train due to a small batch size. To run a single pre-training epoch on the entire dataset takes around 5 hours and around 9 hours per epoch are taken in the main training loop, taking over a week of continuous training. This is due to the sheer size of the dataset and depth of the model, as well as the 256×256 resolution.

Chapter 4

Experimentation

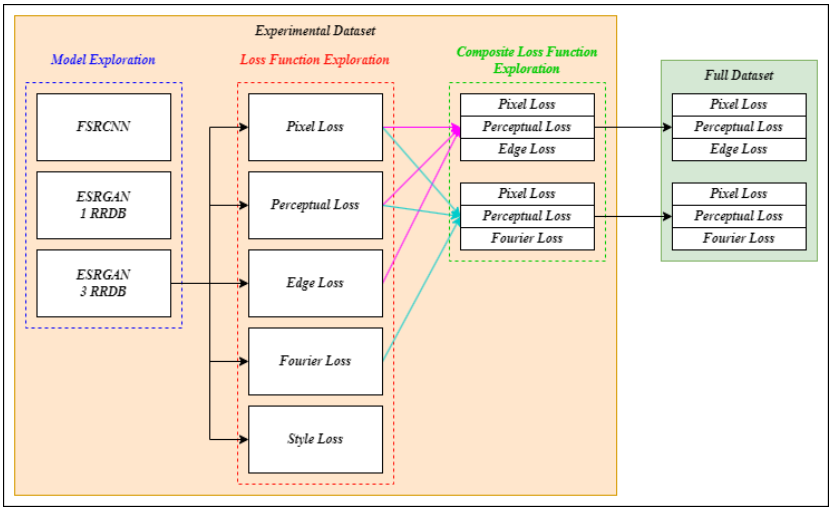

Figure 4.1: Experimental Timeline

4.1 Experimental Dataset

As mentioned in Section 3.6, we used a limited dataset for experimental stages. This was constructed simply by restricting the training dataset size to 10000 slices. We kept the validation set the same size and we also did not feel the need to use a test split, as our aim was simply to compare configurations relative to each other, not to generalise performance. A set of 10000 slices took about 1 hour to train a single epoch, making it relatively time efficient to train.

4.2 Model Architecture Exploration

To explore model architecture depth, we started off using a baseline model of an ESRGAN with 1 RRDB, with a composite loss function set to:

$$L_{gen} = 0.3L_{pixel} + 1L_{perc} + 0.7L_{edge} + 0.001L_{adv}$$

This model contains the same loss function and model depth used by Ayaz *et al.* (2024), making it a good reference point. Our degradation pipeline generates more artefacts than Ayaz *et al.* (2024), simulating GRAPPA and Partial Fourier where they only used a Tukey window, so we assumed that increasing the model depth might be necessary for the model to gain an understanding of the artefacts. Therefore, we also ran an experiment with the same model using 3 RRDBs to test this hypothesis. Additionally, we wanted to verify if a GAN improved performance significantly compared to its simpler counterpart, the CNN. If the CNN generates similar performance to a GAN, then the extra computation cost and training time is not justified, so we ran an experiment using FSRCNN.

Model	PSNR	SSIM	LPIPS
ESRGAN 1 RRDB	39.823	0.952 ± 0.132	0.046 ± 0.119
ESRGAN 3 RRDB	40.794	0.951 ± 0.129	0.055 ± 0.110
FSRCNN	36.26	0.945 ± 0.124	0.072 ± 0.111

The ESRGAN-based models outperformed the FSRCNN model in all metrics, which shows that the pixel-level similarity, structural similarity, and perceptual similarity was higher for ESRGAN-based models. Between the two ESRGAN models, the 1-RRDB ESRGAN did generate slightly higher perceptual similarity. This indicates that details like edge similarity and texture similarity were higher for a lower RRDB depth. However, the SSIM was similar throughout for different RRDBs and the 3-RRDB ESRGAN scored a higher PSNR. From this data, we took that the ESRGAN is a stronger candidate for this task than FSRCNN. The results suggest that the image generated 3-RRDB model was smoother and has sharper textures or edges that resemble human perception more closely than the 1-RRDB model. However, these figures are limited as we only look at the model after 5 epochs of full training, meaning that it is possible and likely that the models learn at different rates and reach different minima at this point in time, where later on, one model might outperform another.

When training the ESRGAN, we did use 3 pre-training epochs with L1 (pixel) loss to stop the discriminator overpowering the generator immediately (Figure 4.2). We observed a good training curve for both ESRGAN setups using the following hyperparameters:

Hyperparameter	Value
Batch size	4
Learning rate	1×10^{-4}
Generator Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Discriminator Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Pre-training epochs	3
Training epochs	5

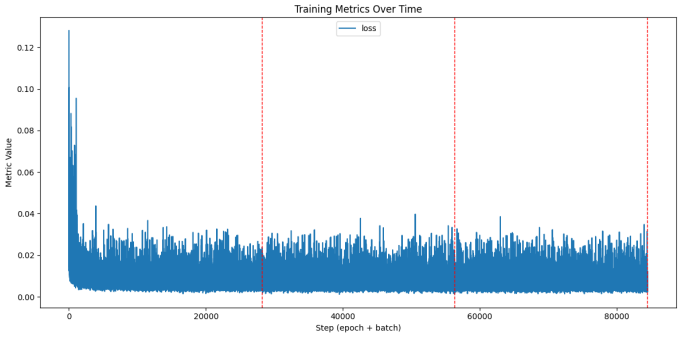

Figure 4.2: Generator loss over 3 pre-train epochs.

A batch size of 4 was chosen as this was the largest possible batch size that could fit into VRAM, all other hyperparameters followed Ayaz *et al.* (2024). Unfortunately, neither generator nor discriminator loss directly correlate with image quality, so instead we observe the PSNR and SSIM for the training and validation set over epochs to check for overfitting and underfitting, which is not observed as they roughly track each other (Figure 4.3).

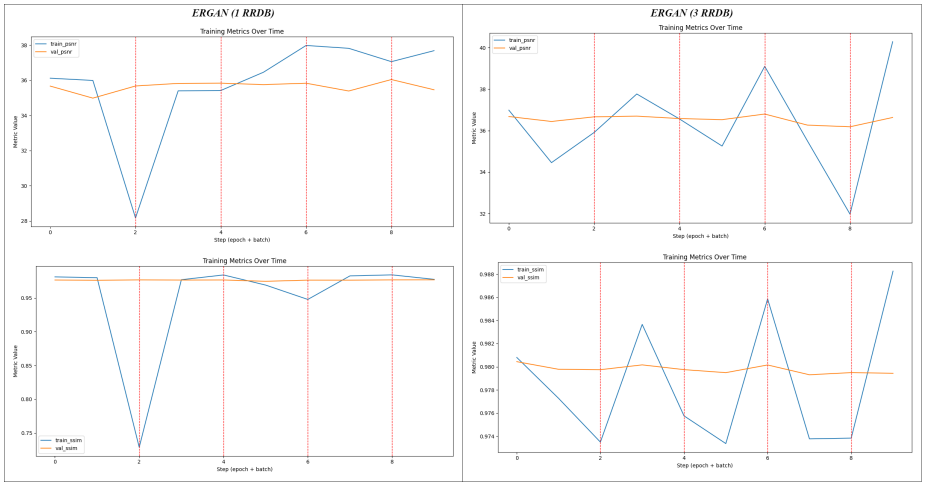

Figure 4.3: Training and validation PSNR and SSIM over epochs.

PSNR and SSIM remain roughly the same values over the 5 epochs of the main training loop, but when observing a slice, we still see qualitative improvements even when the metrics show no meaningful change.

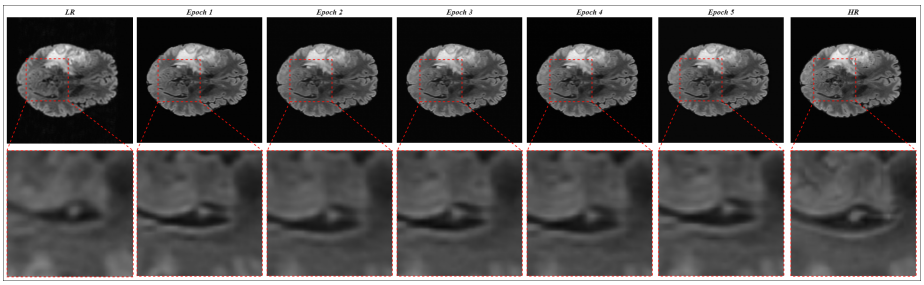

Figure 4.4: Evolution of slice super-resolution over epochs for 3 RRDB ESRGAN model.

For FSRCNN we found that the optimal combination of hyperparameters were:

Hyperparameter	Value
Batch size	4
Learning rate	1×10^{-4}
Optimizer	Adam
Training epochs	5

Overall, we noticed that the main difference was that the 3 RRDB model appeared to be able to better discern and remove artefacts introduced from the degradation pipeline as the depth likely enabled it to understand typical matter patterns and abnormality (likely accounting for the higher PSNR but lower LPIPS). A good example is shown below in Figure 4.5, which shows the 3 RRDB model eliminating dark spots created by ringing artefacts. Additionally, FSRCNN generated blurrier images than both, indicating the need for a GAN model. As we felt that the extra model depth helped the ESRGAN better distinguish between artefacts and signal, we decided to focus on the 3 RRDB model for further experimentation.

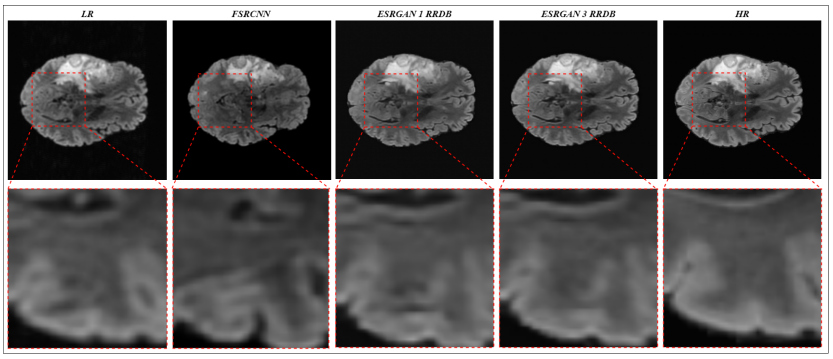

Figure 4.5: Side-by-side comparison of same slices super-resolved using FSRCNN, ESRGAN (1 RRDB), and ESRGAN (3 RRDB).

4.3 Loss Function Exploration

To gain a better understanding on what visual effect each loss function had when training, we decided to train our best performing architecture (ESRGAN with 3 RRDBs) over 5 epochs using each loss function in isolation with adversarial loss. We chose a small epoch number as this step was designed to help us understand what each loss function optimised, which is only used to decide which loss functions to use in a composite loss function, only intending to use these models as an informational stepping stone. As we already determined viable hyperparameters for the 3 RRDB ESRGAN, we used the same hyperparameters as in Section 4.2. Additionally, we kept the same 3 pre-training epochs as in Section 4.2.

Loss	PSNR	SSIM	LPIPS
Pixel	42.354	0.956 ± 0.126	0.044 ± 0.111
Perceptual	40.808	0.952 ± 0.132	0.049 ± 0.108
Edge	38.122	0.941 ± 0.140	0.064 ± 0.118
Fourier	38.409	0.936 ± 0.153	0.063 ± 0.116
Style	39.238	0.943 ± 0.118	0.047 ± 0.107

Table 4.1: Performance metrics for different loss functions across 5 epochs on a 3300-slice validation set.

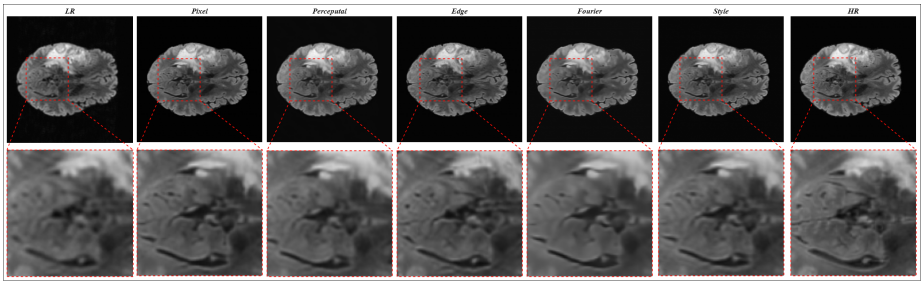

Figure 4.6: Comparison of image output after training on single loss functions.

4.3.1 Pixel Loss

Training a model using purely pixel loss for 5 epochs after pre-training returned the highest SSIM and lowest LPIPS, suggesting that the image generated by this loss function is perceptually the closest to HR than all the other loss functions by itself, returning the sharpest edges and textures. It also received the highest PSNR, suggesting that the overall noise was relatively low and the basic structures were well preserved.

In practice, we observed that it returned a surprising level of accuracy in the texture boundaries such as the edge of the brain and empty space, or on lesion boundaries. As is expected over 5 epochs, detail in texture was lacking and regions that were heavily obscured by ringing and blurring were not recoverable, though this was also true for most other loss functions. We generally saw

that ringing artefacts were reduced, however some strong ringing artefacts are perceived as edges (Figure 4.7).

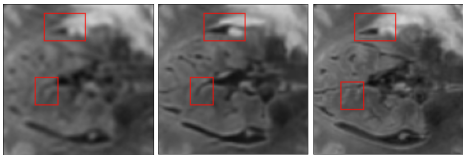

Figure 4.7: Imagined edges due to ringing artefacts.

Generally, pixel loss seemed to perform better on white matter than grey matter, struggling somewhat on central areas of the brain which contained a high amount of detail (and artefacts as a consequence).

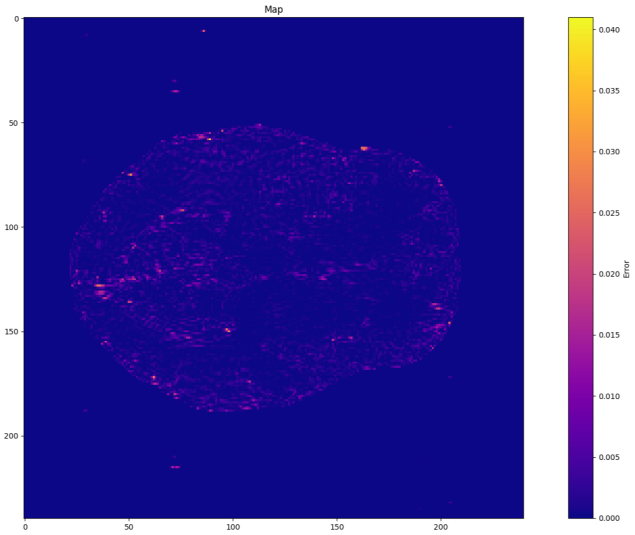

Figure 4.8: MAE map between SR and HR slice.

It is worth noting that the pre-training was done using L1 (pixel) loss, so its strong performance in terms of edges compared to other loss functions might be because the other loss functions are undoing the work of pixel loss in their 5 post-pre-training epochs and so are further back in the training process resulting in an averaging effect whilst this model does not have to re-learn anything. Overall, the pixel loss performance was relatively strong at defining strong edge boundaries, but was affected by artefacts.

4.3.2 Perceptual Loss

Perceptual loss contained a relatively high SSIM, but a lower LPIPS than pure pixel loss, suggesting that the images generated had similar structural features, but some of the textures are not as accurate. It also contained the second highest PSNR, just behind pixel loss, suggesting that in general, the uniform noise was low and basic structures are similar.

Generally, we observed a similar behaviour from pixel loss and perceptual loss across 5 epochs, with perceptual loss returning slightly less sharp images. We did observe that perceptual loss had a slightly higher sensitivity to both edges and artefacts, meaning that we often saw light residual ringing artefacts that were present in LR but not HR, and some more detailed textures, both of which were not present in the pixel loss model. However, most borders appeared blurrier than pixel loss. Similarly, regions with heavy blur and artefacts were not recoverable.

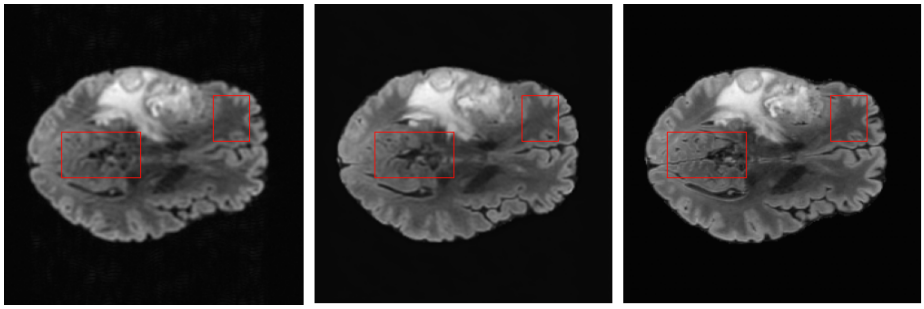

Figure 4.9: Unrecoverable detail (*left*) and residual ringing artefacts which are not present in pixel loss (*right*).

Perceptual loss seemed to perform better on white matter than grey matter, and struggled in central areas with lots of detail, much like pixel loss.

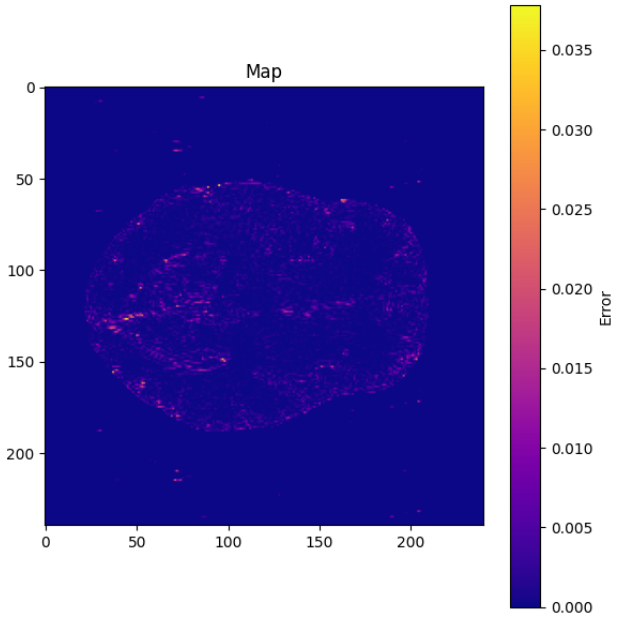

Figure 4.10: MAE map between SR and HR slice.

Overall, we saw similar performance positive performance from perceptual loss to pixel loss, with a slight underperformance. However, over more epochs perceptual loss functions tend to outperform pure L1 pixel loss in GANs (Ledig *et al.*, 2017).

4.3.3 Edge Loss

While perceptual loss and pixel loss perform similarly, edge loss is designed to prioritise penalising inaccurate edges. We saw that this impacted the SSIM, PSNR, and LPIPS negatively, receiving the lowest PSNR and worst LPIPS. To some extent this was expected, as prioritising edges would not create a balanced final image, so its human perception would be worse. As the loss function does not attempt to penalise each aspect of the image in a balanced manner, we also were not expecting the pixel-level difference to be low. We were more interested in the visual effects of this loss function.

The main difference that this loss function made visually was that it amplified edge structures. This is both useful in the sense that matter boundaries were far clearer, but as the LR image was riddled with ringing artefacts, it also amplified these to a larger extent than the perceptual loss did.

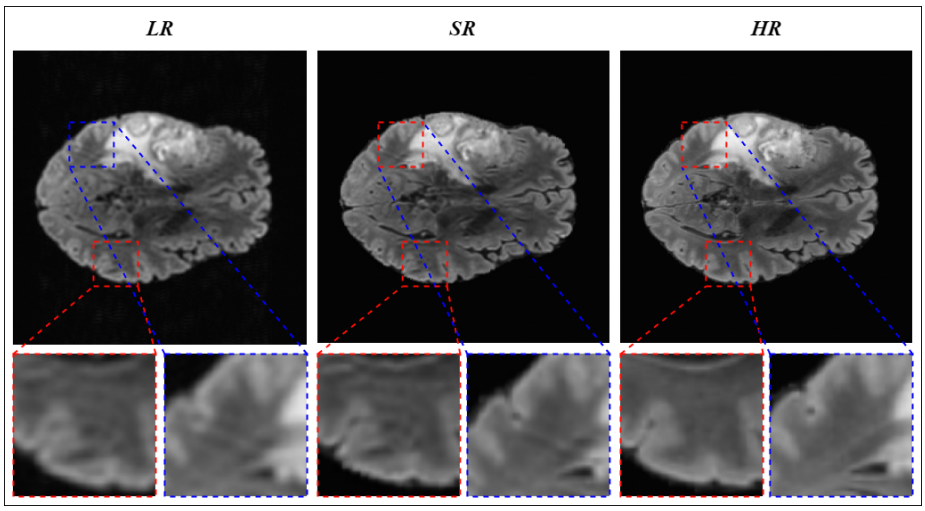

Figure 4.11: Amplification of ringing issues caused by edge loss.

The amplification of the ringing artefacts can also be seen in Figure 4.11:

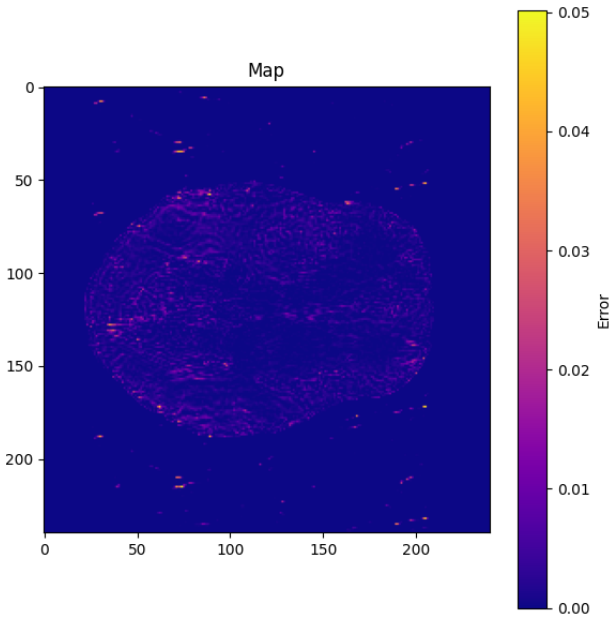

Figure 4.12: MAE map between SR and HR slice.

Ayaz *et al.* (2024) previously used a composite loss function with edge loss successfully, but their dataset had far less intense artefacts than ours. Nevertheless, we decided that it was worth exploring edge loss further in a composite loss function due to its evidence of past success; we also wanted to see if we could find a loss function that highlights edges and doesn't amplify artefacts as much.

4.3.4 Fourier Loss

One alternative loss function that we devised was Fourier loss. This was brought in to try and perform the same edge amplification as edge loss without the artefacts. As they perform a similar function, the PSNR and LPIPS scores are comparable, with the SSIM being slightly lower than edge loss.

Visually, Fourier loss is effective at preserving edges on the outside boundary of the brain but does smooth all textures creating a flat image with little texture. This creates an image where the main structural features and low-frequency content are kept, but finer details are removed. Importantly, Fourier loss is resilient to ringing artefacts, with a lack of overall texture as a trade-off.

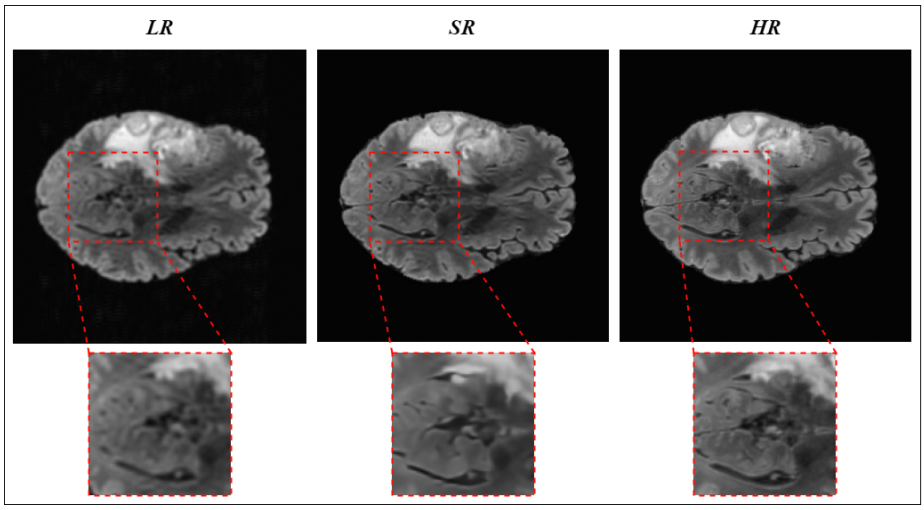

Figure 4.13: Lack of texture and artefacts caused by Fourier loss.

We see in the MAE map that Fourier loss tends to operate similar to pixel and perceptual loss, but any textures are highlighted as Fourier loss smooths them out, creating a difference between SR and HR:

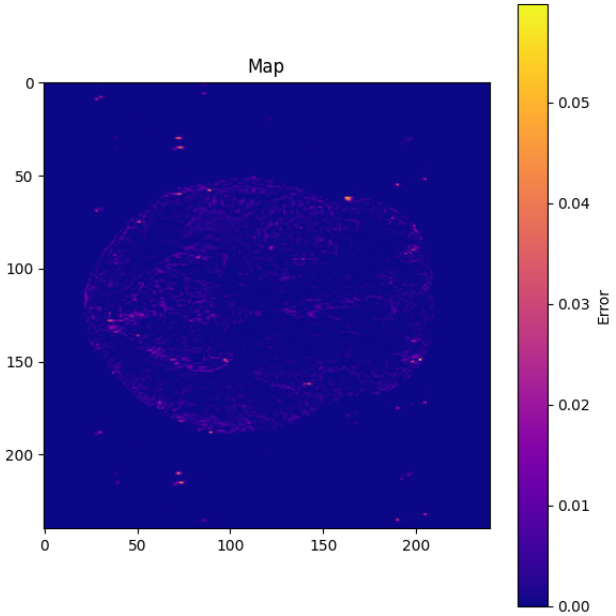

Figure 4.14: MAE map between SR and HR slice.

Overall, we thought that this type of loss would be worth further testing in our composite loss function as it acts in a similar way to edge loss, which has evidence of performing well in a composite loss function, without the drawback

of amplifying artefacts.

4.3.5 Style Loss

We wanted to see if 1.5T to 3T super-resolution could be thought of as a style transfer, for this purpose we tested style loss. On metrics alone, style loss achieves an LPIPS score that was far better than expected, with an SSIM and PSNR slightly above edge loss. The LPIPS score suggests that the images generated look visually similar under a human perception. However, we noticed that using style loss produces images that contain relatively realistic textures and strong boundaries, but these boundaries appeared to be far more inaccurate than any of the other loss functions. Often the shape of the edges was distorted and different shapes were created in the SR version that were not similar to either LR or HR.

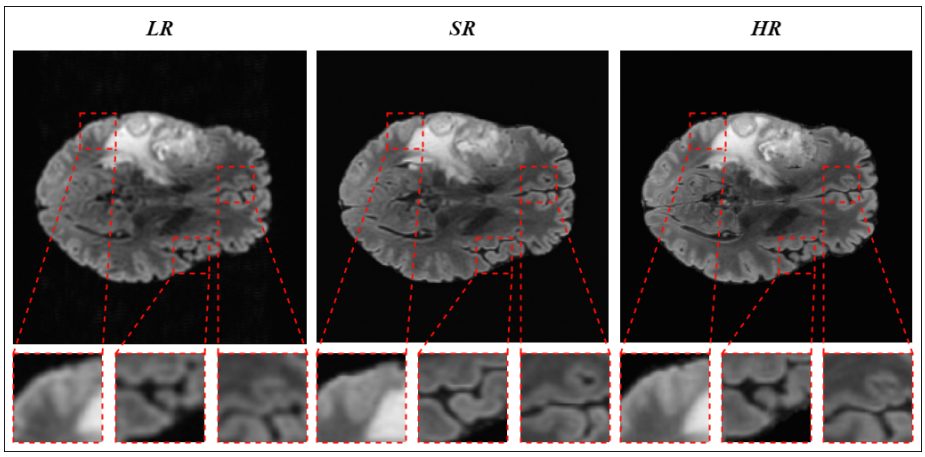

Figure 4.15: Lack of texture and artefacts caused by Fourier loss.

We opted not to use this loss function, despite its relatively high metrics, as we worried that the images produced would have distorted edges and boundaries, which would be more clinically risky in areas such as tumour detection and segmentation than inaccurate textures.

4.3.6 Preliminary Composite Loss

Using the information gathered from our loss function experimentation, we decided to experiment with two composite loss functions on the same 3-RRDB ESRGAN model. We used the same 3 pre-training epochs and the same hyperparameters as the previous section, also training for 5 full epochs after pre-training. The purpose of this stage was to verify that these composite loss functions are working before we dedicate time to train on the full dataset. The first loss function that we trained with was the same as Ayaz *et al.* (2024), which we chose as we had some evidence demonstrating its applicability:

$$L_{gen} = 0.3L_{pixel} + 1L_{perc} + 0.7L_{edge}$$

We had already checked this composite loss function in Section 4.2, as such, we did not need to repeat the experiment.

The second composite loss function that we trained aimed to swap the edge loss component from Ayaz *et al.* (2024) with Fourier loss to reduce the impact of ringing artefacts:

$$L_{gen} = 0.3L_{pixel} + 1L_{perc} + 0.0001L_{fourier}$$

Fourier loss contains a much higher magnitude than edge, pixel, and perceptual loss, hence the weighting needed to be adjusted to $\lambda_4 = 0.0001$. We were able to calibrate this using first-pass magnitudes of the first 1000 slices in the dataset. First we observed the magnitude of Fourier loss at $\lambda_4 = 1$ when compared to the first composite loss function (including edge loss), we were then able to scale λ_4 down to a similar magnitude to L_{edge}.

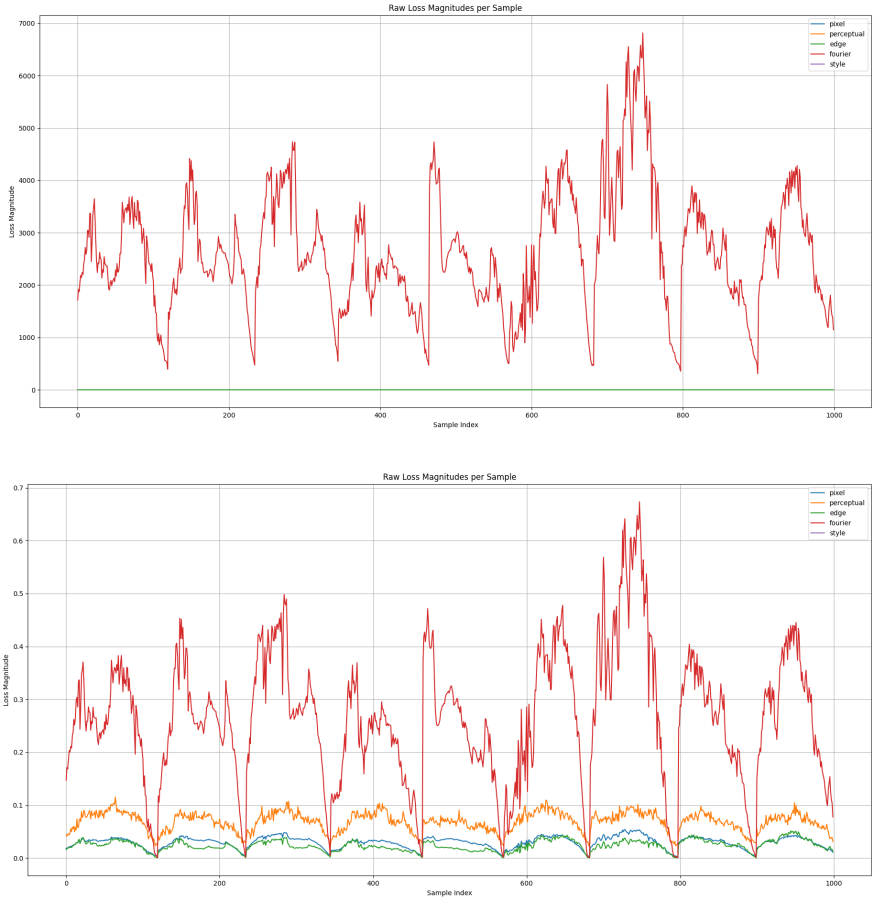

Figure 4.16: Loss magnitudes across the first 1000 slices after a single pass at $\lambda_4 = 1$ (top) and $\lambda_4 = 0.0001$ (bottom).

Note that the Fourier loss magnitude is higher on the first pass but scales down

to match L_{edge} after gradients are adjusted. We aimed for a λ_4 that matches the magnitude during training:

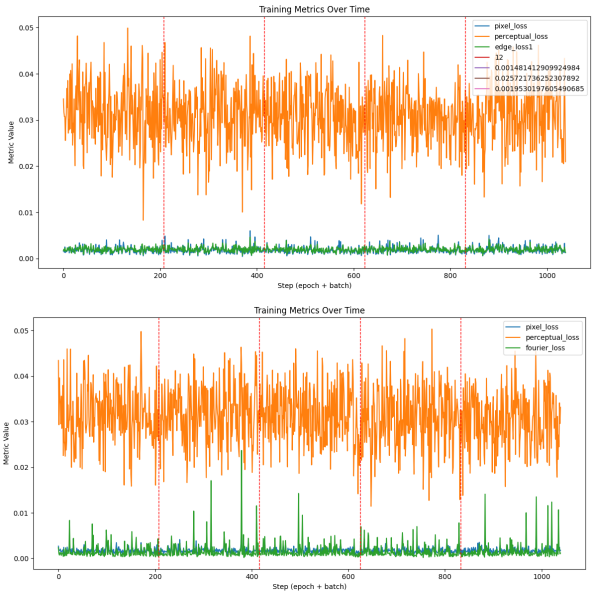

Figure 4.17: Loss magnitudes during training with edge loss (*top*) and Fourier loss (*bottom*).

Using these two loss functions over 5 full training epochs we were able to get these preliminary results:

Model	PSNR	SSIM	LPIPS
Edge	40.794	0.951 ± 0.129	0.055 ± 0.110
Fourier	39.592	0.951 ± 0.132	0.055 ± 0.055

These indicate to us that our composite losses performed comparatively to each other and the individual loss functions, warranting a training run on the full dataset.

Chapter 5

Final Model Evaluation

5.1 Training

We trained both models with the same hyperparameters as in Section 4.5, but with 20 full training epochs instead of 5.

Hyperparameter	Value
Batch size	4
Learning rate	1×10^{-4}
Generator Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Discriminator Optimizer	Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$)
Pre-training Epochs	3
Training Epochs	20
Early stopping patience	10

Following the results from Section 4.3.6, we decided to explore both loss functions:

- $L_{gen} = 0.3L_{pixel} + 1L_{perc} + 0.7L_{edge}$
- $L_{gen} = 0.3L_{pixel} + 1L_{perc} + 0.0001L_{fourier}$

The training process for each used a training set of around 168885 slices, taking around 9 hours per epoch. After pre-training each model took around 180 hours to train the full model. For both ESRGANs we observed that the generator and discriminator losses were balanced over the 20 epochs (Figure 5.1), indicating that adversarial training was steady.

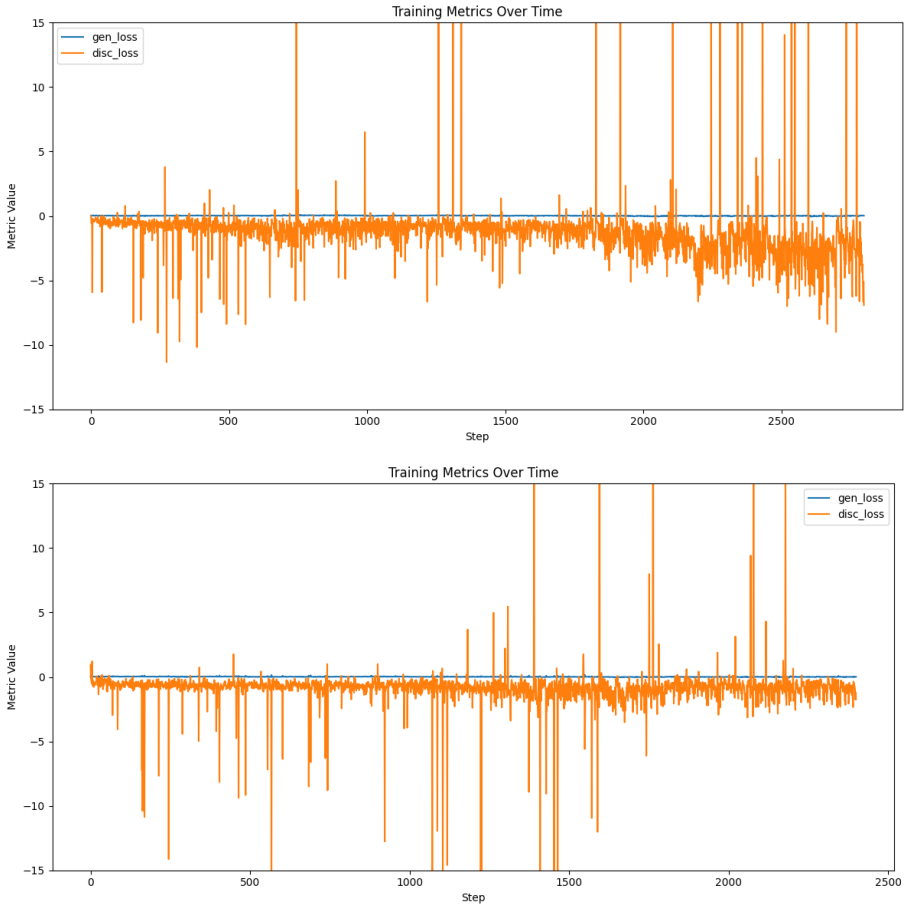

Figure 5.1: Training and validation loss for ESRGAN 1 (*above*) and ESRGAN 2 (*below*).

For ESRGAN 1, while SSIM and PSNR values fluctuated in the training set, we saw that they increased slowly over the epochs, though not by much. As SSIM and PSNR were already high after pre-training (around 0.97 and 36 respectively in the validation set), we did not expect much increase, as most of the changes that the GAN would be making would be very fine textural and structural changes which may not be reflected in PSNR or SSIM. This can be seen in Figure 5.2:

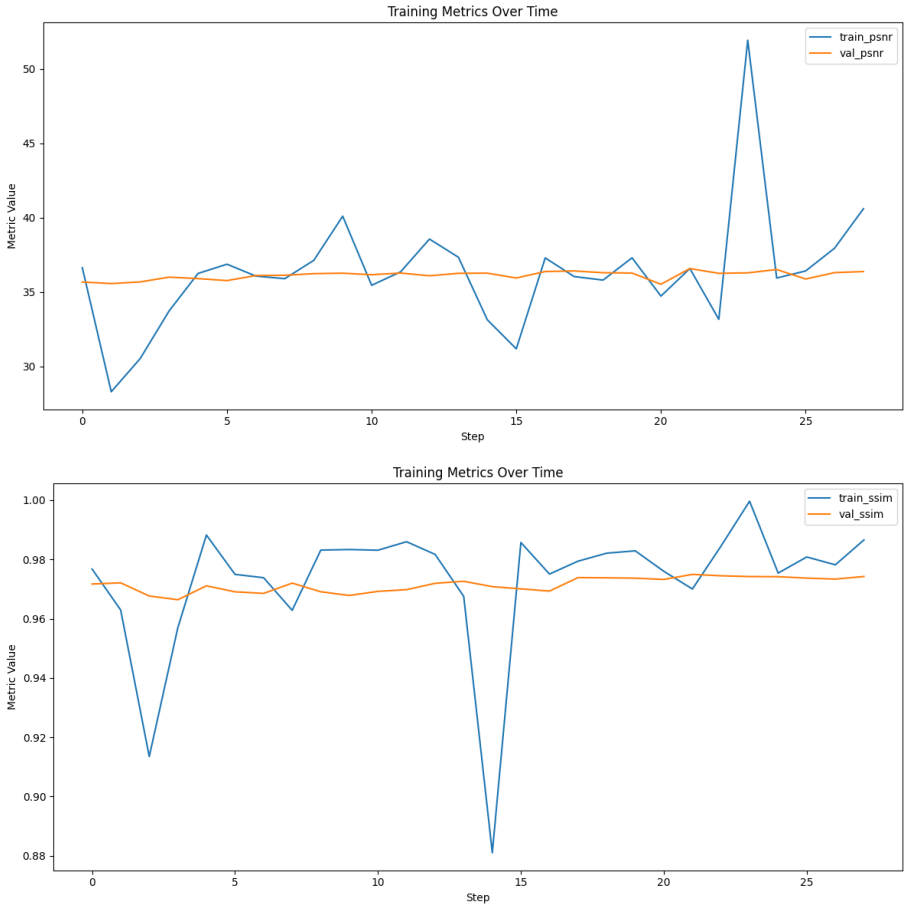

Figure 5.2: ESRGAN 1 PSNR (*above*) and SSIM (*below*) during training.

For ESRGAN 2, we saw similar results with the validation PSNR and SSIM also slightly improving, but overall the validation and training metrics staying balanced with each other. This shows a relatively healthy training process for both ESRGANs, as a higher training SSIM or PSNR would indicate an overfitting model. This can be seen in Figure 5.3:

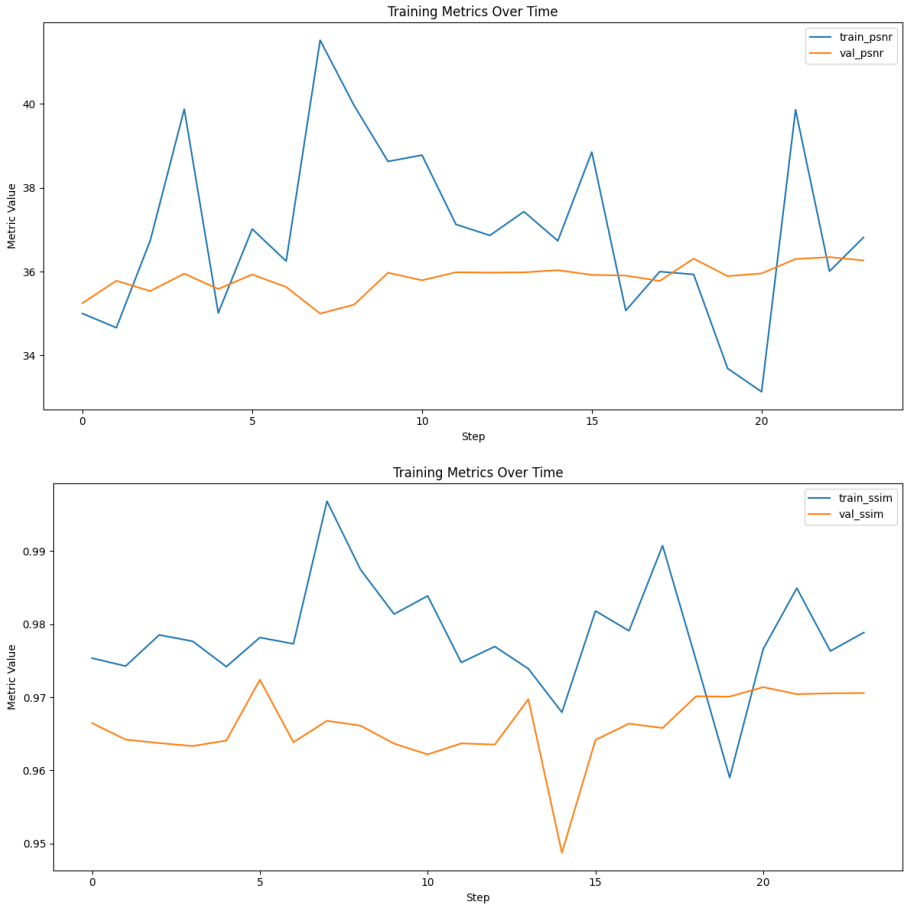

Figure 5.3: ESRGAN 2 PSNR (*above*) and SSIM (*below*) during training.

5.2 Quantitative Results

To perform a quantitative evaluation on both ESRGANs we conducted PSNR, SSIM, and LPIPS evaluation on an unseen test set (different to that which was used in Section 4) to compare the pixel-wise, and perceptual similarity between the generated SR images and the HR images. We then performed matter and tumour segmentation evaluations mentioned in Section 3.5.5 and 3.5.4.

5.2.1 Image Quality Metrics

Model	PSNR	SSIM	LPIPS
ESRGAN 1	39.096	0.961 ± 0.101	0.032 ± 0.069
ESRGAN 2	38.559	0.959 ± 0.109	0.034 ± 0.085

Using edge loss as opposed to Fourier loss resulted in slightly higher PSNR, SSIM and LPIPS scores, though differences were extremely similar on both

SSIM and LPIPS scores. The use of edge loss creates images that are slightly closer to the ground truth in pixel values than Fourier loss. Overall, a human observer would be unlikely to categorise edge or Fourier loss as visibly worse in degree when compared to the ground truth set. This is not to say that the outputs are similar themselves, as it is possible that different visual artefacts and effects contribute to their similar score.

They did outperform all other models used in Section 4 in LPIPS, and outperformed all but one model in SSIM score, though these cannot be compared directly, as models in Section 4 were evaluated using the validation set. Overall, this indicates that both models produce strong reconstructions of comparable perceptual quality. The choice between different loss functions must be guided by other characteristics.

5.2.2 Matter Segmentation

Tissue	*LR* Dice	ESRGAN 1 Dice	ESRGAN 2 Dice	*p*-value (*LR* vs ESRGAN 1)	*p*-value (*LR* vs ESRGAN 2)
WM	0.0125	0.6805	0.6745	$+1.510 \times 10^{-19}$	$+1.389 \times 10^{-19}$
GM	0.4435	0.6548	0.6555	$+4.129 \times 10^{-13}$	$+1.193 \times 10^{-13}$
CSF	0.0007	0.6926	0.6830	$+1.279 \times 10^{-19}$	$+1.279 \times 10^{-19}$

We see that both edge and Fourier loss offer similar Dice scores in comparison to the HR set when segmented using FSL-FAST. ESRGAN 1 does offer slightly higher Dice scores for CSF and white matter than ESRGAN 2, indicating that it constructs these tissues more accurately. Overall, enhancing LR using either ESRGAN before passing through FSL-FAST creates more accurate matter segmentations compared with the HR set than not using super-resolution. However, this is purely down to the fact that FSL-FAST performed very badly on our LR data, struggling to segment LR volumes accurately (Figure 5.4), only managing to segment grey matter with any sort of accuracy. This limitation is not necessarily a flaw in our method, as it highlights the potential benefit of using super-resolution to recover anatomical detail sufficient for standard clinical tools, though it is also worth noting that our degradation contained very strong artefacts and blurring. Either way it tells us that for our specific profile of artefacts, both of our ESRGANs can help clinical software to segment matter regions.

Dice scores of 0.65–0.69 suggest the model captures some tissue structure but are likely not useful clinically, where values above 0.80 are typically expected for reliable WM, GM, and CSF segmentation (Wong *et al.*, 2024). This indicates the super-resolution improvement does not yet yield accurate enough delineation. However, it is worth noting that these results compare SR segmentation map overlap with HR maps, not the ground truth. If the HR maps are not accurate then the Dice scores may underestimate the true performance of the model, since errors in the reference segmentations would propagate into the evaluation.

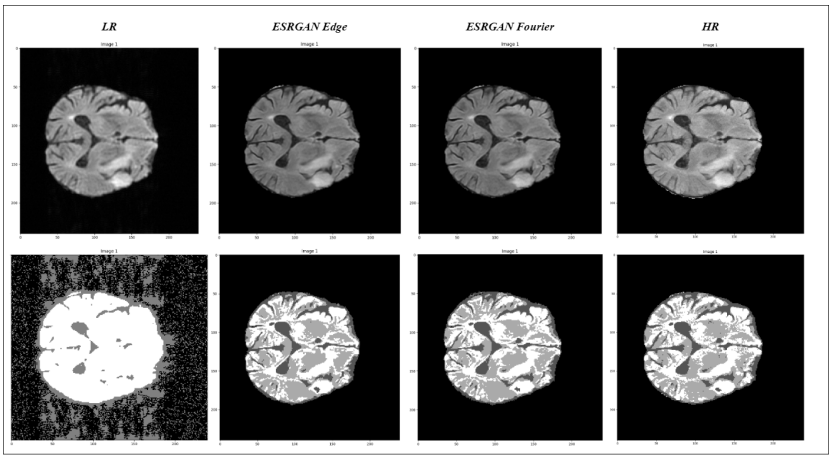

Figure 5.4: Comparison of FSL-FAST segmentation on LR, SR, and HR sets.

5.2.3 Tumour Segmentation

Model	Dice vs HR	p-value vs LR
LR	0.5054	N/A
SR - ESRGAN 1	0.8878	1.1429×10^{-18}
SR - ESRGAN 2	0.8582	1.2713×10^{-18}

Similar to our matter segmentation results, we see that edge and Fourier loss provide similar Dice scores when a tumour region is extracted using DeepSeg, which is significantly higher than the LR segmentation map. We see Dice scores above the 0.80 threshold indicating that our models yield clinically relevant tumour delineation, indicating that super-resolution enhancement can improve segmentation accuracy in downstream tasks. However, these results are limited by comparison to HR rather than ground truth and by DeepSeg's binary tumour masks, which capture only the outer boundary rather than specific subregions.

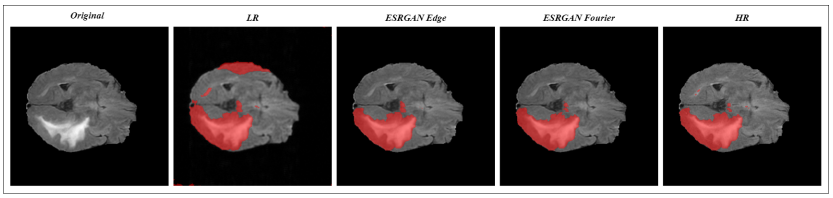

Figure 5.5: Comparison of DeepSeg segmentation on LR, SR, and HR sets.

5.3 Qualitative Results

While visual quality metrics are useful to evaluate the overall performance of both models in some regards, we also noticed several patterns that were demonstrated by models during super-resolution that could not be quantified. We summarise some of the key visual features in this section. Any images shown in this section are representative examples of any visual features, not the only examples of such features occurring.

Some areas in the HR set have ringing artefacts in their corresponding LR images, mainly due to proximity to matter boundaries. In an ideal world the ESRGANs would recognise that these are artefacts and remove them from the SR images, and we see some degree of that happening in both ESRGANs, but it is common to see residual patterns appearing in SR images (Figure 5.6 bottom left).

Both ESRGANs struggled to reconstruct regions heavily affected by blur and artefacts, particularly where the HR image contained many fine textures in close proximity, as seen in Figure 5.6 (top right). However, the models did show good performance on brain surface folds, reconstructing blurry boundaries sharply and accurately (Figure 5.6 bottom right). We expected this strong performance as Fourier or edge loss was designed to target these areas and the contrast difference in LR images is naturally very high, making it easy to locate the boundary.

Our models appeared to struggle slightly on tumour regions, often hallucinating some ridges and textures, mainly due to ringing artefacts in the LR image (Figure 5.6 top left). Tumour reconstructions also appeared less sharp than other areas. We suspected this was because tumour represented a small proportion of overall training data whilst often containing highly varying textures.

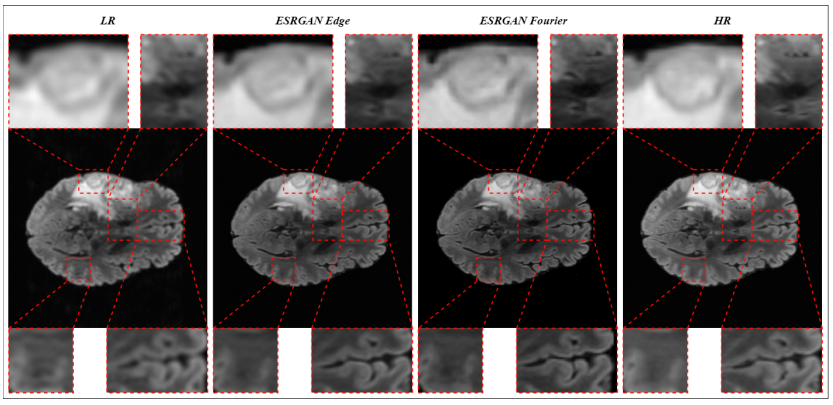

Figure 5.6: Example slice containing common reconstruction artefacts.

We can see in Figure 5.7 that LR slices containing clear edges and boundaries with less fine textures are reconstructed exceedingly well by both ESRGAN models. Both are able to accurately discern between ringing artefacts and true boundaries, understanding the context enough to enhance the latter and remove artefacts.

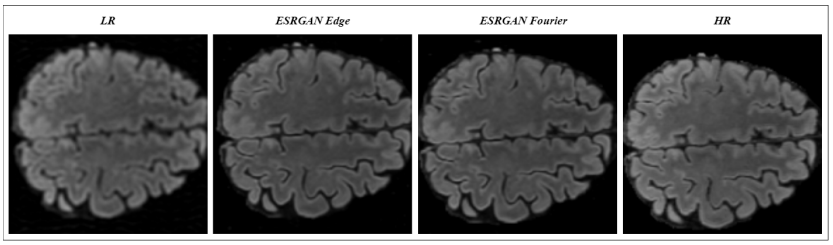

Figure 5.7: Performance on slices containing a large amount of folds.

Our models seem to struggle exactly replicating fine patterns on slices tending toward the base of the brain. We can see in Figure 5.8 that there are fine wave-like textures at the top of the HR image, which still partially exist in the LR degradation, but our ESRGANs flatten and blur these textures noticeably. In this example, the ESRGAN with Fourier loss does do a slightly better job at reconstructing these textures, but both are missing texture. This indicates that our model does struggle with the level of degradation that we applied to HR, and, clinically might struggle to recreate textures from artefact-heavy scans in slices closer to the stem.

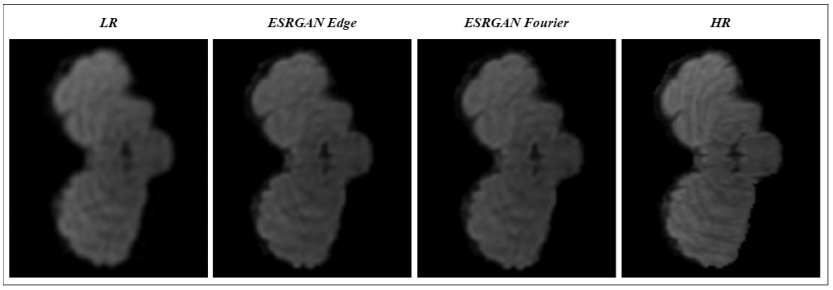

Figure 5.8: Failure to replicate textures on lower slices.

Where the HR image contained noticeable visual noise, neither ESRGAN reconstructs the SR image with as much noise, instead the expected noise is blurred (Figure 5.9). This can be seen as both a positive and negative, as noise in MRI images is generally not desirable as it can mask detail, but our SR images can only contain the level of detail present in the HR image with noise. It is also likely that the explanation for this is that the degradation pipeline removed the noise from HR, and our model wasn't deep enough to understand that it needs to reconstruct it. Overall, this suggests that our models tend to favour cleaner reconstructions over faithfully reproducing noise present in the original scans.

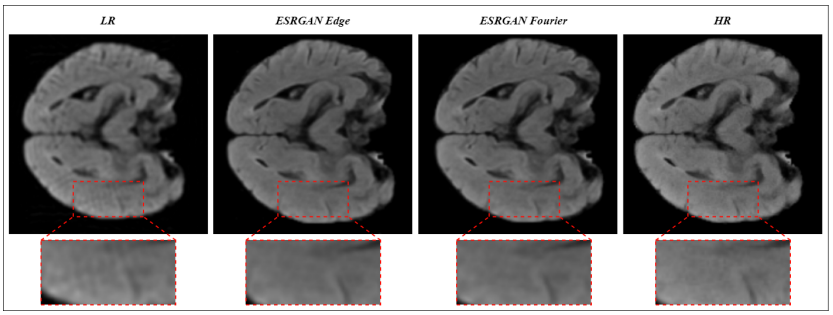

Figure 5.9: Example of lack of noise restoration.

5.4 Robustness to Degradation

Our degradation pipeline contained more artefacts, blurring, and noise than any other degradation pipeline that we were able to find in literature. In reality, data is unlikely to be as degraded as ours but we wanted to test our model on a harsher dataset. However, we also evaluate how our model performs on "easier" data by focusing on only degrading by each method in isolation.

5.4.1 Cylindrical Low-Pass Filter with Tukey Window in Isolation

We wanted to test the performance of our model without any sort of parallel imaging or partial Fourier artefacts as these are not always present. To do this, we generated a duplicate test set without $\mathcal{T}_2$ or $\mathcal{T}_3$, leaving only the cylindrical filter with a Tukey window. This removed most of the ringing artefacts, leaving a blurred image. We kept the Rician noise as it is inherent to 1.5T scans.

Model	PSNR	SSIM	LPIPS
ESRGAN 1	38.410	0.972 ± 0.073	0.029 ± 0.017
ESRGAN 2	37.295	0.978 ± 0.014	0.032 ± 0.018

When the LR image dataset doesn't contain GRAPPA or Partial Fourier artefacts, both LPIPS and SSIM scores improved, with the PSNR decreasing slightly. We see a major improvement in SSIM score from ESRGAN 2, which jumps from 0.959 to 0.978, outperforming ESRGAN 1. This shows that using Fourier loss with a simpler degradation seems to pick up structural changes effectively, perhaps more so than edge loss. Using only $\mathcal{T}_1$ is actually closer to degradation simulations used in other studies such as Ayaz *et al.* (2024), and is likely to be more visually similar to MRI scans in clinical settings, as we used a very high level of artefact to simulate our 1.5T scans. That being said, using ESRGAN 1 did create a lower PSNR and slightly higher LPIPS score, indicating that the pixel-level difference and human perceptual quality differences are slightly worse.

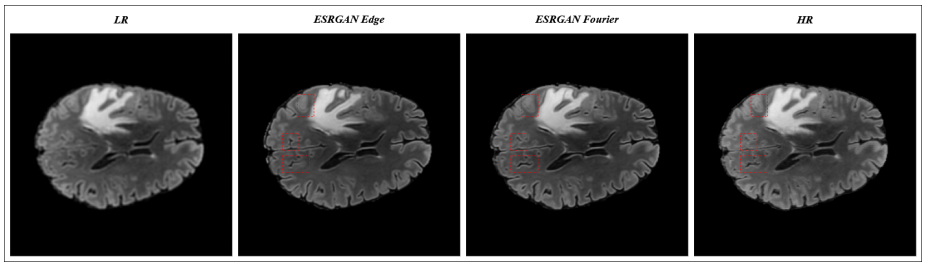

Figure 5.10: Outputs of ESRGAN 1 and ESRGAN 2 on slice with only cylindrical low-pass filter with Tukey window and Rician noise.

Without GRAPPA and Partial Fourier artefacts, SR images from both models look far closer to HR than using the full degradation pipeline, and both models are very similar for the most part. Upon visual inspections, we did notice a number of cases where ESRGAN 2 produced more accurate boundaries than ESRGAN 1, which had a tendency to erase boundaries if they were small enough (Figure 5.10).

Overall, both losses perform far better when $\mathcal{T}_2$ and $\mathcal{T}_3$ are not included, and we were impressed by the SSIM improvements and accurate border reconstruction shown by ESRGAN 2. Given that real-world LR images are likely to resemble this simplified degradation pipeline, our results demonstrate that a model trained with Fourier loss can be highly effective

5.4.2 GRAPPA Degradation in Isolation

We wanted to test the performance of both ESRGANs using an LR set generated only using GRAPPA artefacts $\mathcal{T}_2$. To do so we did need to reduce the signal multiplier to $\alpha = 15$ as the original signal multiplier created reconstruction lines that were far too intense for the sample k-space, creating unnecessarily high artefact levels. We achieved the following image quality metrics.

Model	PSNR	SSIM	LPIPS
ESRGAN 1	30.615	0.921 ± 0.095	0.086 ± 0.074
ESRGAN 2	29.913	0.930 ± 0.057	0.077 ± 0.058

These metrics are unusually low for both models, and the primary reason is that both models created black regions occasionally on the reconstructed images (Figure 5.11). Excluding these regions, the reconstructions were generally accurate, although performance with edge loss was noticeably poorer.

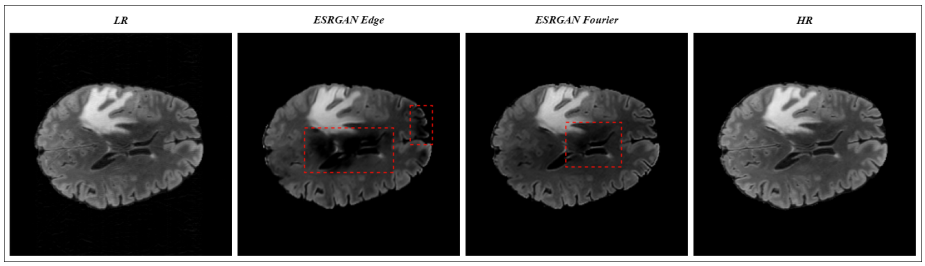

Figure 5.11: Outputs of ESRGAN 1 and ESRGAN 2 on slice with only GRAPPA simulation and Rician noise.

5.4.3 Partial Fourier Degradation in Isolation

Similar to the previous stages, we wanted to simulate LR using only Partial Fourier artefacts $\mathcal{T}_3$, achieving the following results:

Model	PSNR	SSIM	LPIPS
ESRGAN 1	31.458	0.933 ± 0.088	0.063 ± 0.047
ESRGAN 2	31.544	0.949 ± 0.037	0.062 ± 0.032

We see the same issue as before in Section 5.4.2, where all visual quality metrics are decreased. This is also due to hallucinated black regions in both ESRGAN, which are more prevalent when using ESRGAN 1 (Figure 5.12).

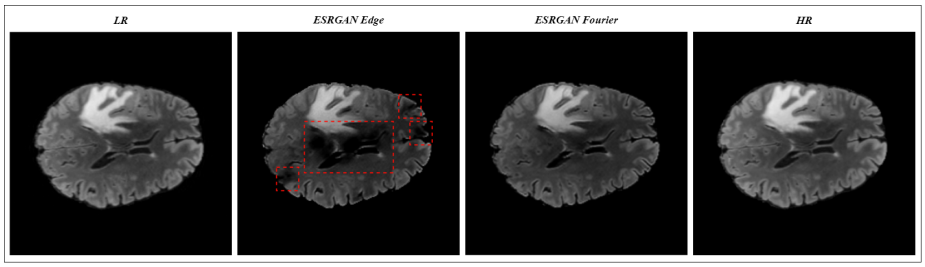

Figure 5.12: Outputs of ESRGAN 1 and ESRGAN 2 on slice with only Partial Fourier simulation and Rician noise.

Overall, this indicates that our models can fail when the LR images contain artefacts without accompanying blur. While cases like these shouldn't be common in typical clinical settings, they highlight a limitation in generalisability for degradation driven purely by acquisition protocol differences when used on a model not trained for that specific domain. This means that for images lacking significant blur, dedicated training with artefact-specific degradations may be necessary to achieve reliable performance.

5.5 Performance Across Slice Position

We evaluated whether our model performed better on different areas of the brain, so we averaged SSIM, PSNR, and LPIPS for each axial slice in the dataset.

As expected, performance was generally consistent across slice positions. We generally observe the best performance across all metrics in the outer slices which are closer to the neck and the top of the skull (Figure 5.13). In contrast, the middle slices showed a noticeable trough in all three metrics. This may be because these slices tend to cover a larger area, include more edges and boundaries between different tissue types, and contain a higher frequency of complex textures.

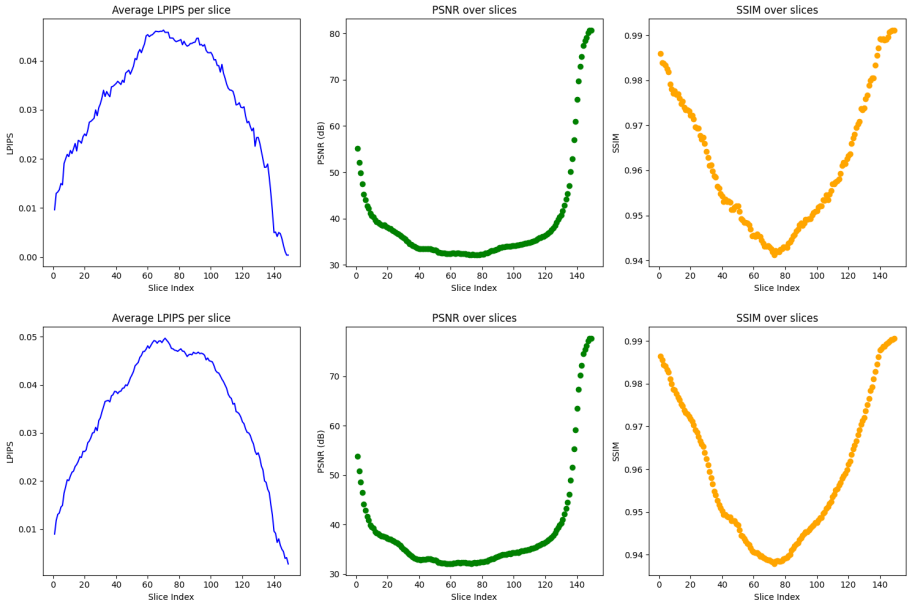

Figure 5.13: Comparison of ESRGAN 1 and ESRGAN 2 average PSNR, SSIM, and LPIPS over slices 0-150.

Chapter 6

Discussion

6.1 Synthesis of Findings

We were able to train both ESRGANs steadily for 20 epochs with balanced generator and discriminator losses. Given that PSNR and SSIM were already high after pre-training, improvements were small afterwards, with most changes being expressed in finer detail as opposed to visual quality metrics. ESRGAN 1 (containing edge loss) yielded a slightly better PSNR and SSIM on the full degradation LR set, while ESRGAN 2 showed comparable perceptual quality (LPIPS).

Both models greatly improved FSL-FAST brain matter segmentation Dice scores compared to LR inputs, but this was mainly down to the LR input being so degraded that FSL-FAST could not provide any useful segmentation information. Additionally, the accuracy achieved when compared to the original HR data segmented using FSL-FAST was not particularly impressive. However, our models significantly improved performance of DeepSeg's binary tumour segmentation models when compared to degraded data, yielding Dice scores exceeding the 0.8 threshold when compared to HR, indicating a limited clinical usefulness.

Qualitatively, both models excelled at reconstructing boundaries between the brain surface and empty space, and performed well where LR contained simple blur that did not obscure the HR textured to a great extent. Both models struggled to reconstruct areas where ringing artefacts heavily obscured regions, tumours, and complicated textures. Fine detail reproduction in lower brain slices was also weaker than central slices.

When the degradation intensity was reduced, we found strong performance on simple blur and noise degradations, particularly for ESRGAN 2. Both models exhibited poorer performance on degradation with only GRAPPA and Partial Fourier artefacts without blurring, producing black-region artefacts and large metric drops. This problem was more extreme with ESRGAN 1.

6.2 Contextualisation in Literature

Before comparing our results with any from previous literature, it is worth noting that we use a highly unique method of degradation, producing more

complex and a higher level of artefacts than any other works to the best of our knowledge.

We first compare our results with Ayaz *et al.* (2024), who use the Human Connectome Project dataset as opposed to BraTS for training, with a significantly smaller training set (30 vs our 1251 volumes). They use a test set containing volumes from HCP, OASIS, and MRBrainS18, containing 17 vs our 109 volumes, giving our set more diversity. Whilst our degradation pipeline consisted of three steps to produce LR, theirs only contained the initial Tukey window filter step, making the resulting image far less artefact-heavy. Their model was also designed to be executed on smaller 64×64 patches rather than ours which operates on entire slices. Based on these reasons, we estimated that our results would be worse, but it would be difficult to quantify enough to perform a direct comparison.

Model	Dataset	PSNR	SSIM
ESRGAN 1	BraTS 2024 - BraSyn $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3$	39.096	0.961 ± 0.101
ESRGAN 2	BraTS 2024 - BraSyn $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3$	38.559	0.959 ± 0.109
ESRGAN 1	BraTS 2024 - BraSyn $\mathcal{T}_1$	38.410	0.972 ± 0.073
ESRGAN 2	BraTS 2024 - BraSyn $\mathcal{T}_1$	37.295	0.978 ± 0.014
FSRCNN	HCP+OASIS+MRBrainS18	39.27	0.991 ± 0.0004
RRDB	HCP+OASIS+MRBrainS18	43.21	0.995 ± 0.0002
DeepUResnet	HCP+OASIS+MRBrainS18	38.58	0.987 ± 0.0006
ESRGAN	HCP+OASIS+MRBrainS18	40.08	0.992 ± 0.0004

As expected our models performed worse in SSIM and PSNR values than theirs, but results were within the same ballpark, suggesting that the performance is somewhat comparable.

A systematic review by Muhammad *et al.* (2024) details the performance of different MRI super-resolution models on their respective datasets. The degradation for these images is simply downscaling spatial resolution by either $\times 2$, $\times 3$, or $\times 4$, which is very different to our methods. We focus on models in the review which are trained on the BraTS 2019, 2018, 2017, and 2015 datasets with a $\times 2$ super-resolution, as these are likely to be the most similar in terms of dataset content and degradation level.

Model	Dataset	PSNR	SSIM
ESRGAN 1	BraTS 2024 - BraSyn $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3$	39.096	0.961 ± 0.101
ESRGAN 2	BraTS 2024 - BraSyn $\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3$	38.559	0.959 ± 0.109
ESRGAN 1	BraTS 2024 - BraSyn $\mathcal{T}_1$	38.410	0.972 ± 0.073
ESRGAN 2	BraTS 2024 - BraSyn $\mathcal{T}_1$	37.295	0.978 ± 0.014
MDGUN (Yang *et al.*, 2022)	BraTS 2019	35.969	0.9703
MS-GAN (Zhu *et al.*, 2019)	BraTS 2019	27.90	0.8300
Andrew *et al.* (2021)	BraTS 2017	29.023	0.8840
SwinMR (Huang *et al.*, 2022)	BraTS 2017	32.07	0.9450
FSCWN (Shi *et al.*, 2019)	BraTS 2015	36.53	0.9583
CDSR (Wang *et al.*, 2019)	BraTS 2015	39.84	0.9904
MFER (Li *et al.*, 2023)	BraTS 2015	40.44	0.9897

Again, direct comparison is not possible, but we see that our models do have similar performance to other solutions.

6.3 Practical Implications

Overall, both ESRGANs have demonstrated a relatively competitive level of performance in line with other GAN MRI super-resolution models. This means that medical facilities in areas like Sub-Saharan Africa with older 1.5T scanners which contain artefacts and blurring could theoretically make use of our GAN to achieve 3T-like image quality for enhanced lesion clarity and diagnosis. Our super-resolution models have demonstrated some ability to improve automated tumour segmentation (at least to a similar quality to 3T data), though did not demonstrate impressive improvement in automated matter segmentation. This means that while super-resolution may help improve downstream tasks that rely on clear lesion boundaries, such as tumour detection, it is less likely to meaningfully enhance tasks that depend on accurate tissue differentiation, like WM, GM, and CSF segmentation.

However, we recommend using with caution as our models can hallucinate some edges when ringing artefacts are heavily obscuring edges, and tumour reconstructions are not always as sharp as other enhancements. These factors can lead to incorrect boundaries being visualised, which means that tumour areas may not be accurately mapped in diagnostic situations. Additionally, heavy ringing can cause residual texture effects, meaning that fine textures can be imagined incorrectly by our models, which could mislead radiologists or automated segmentation systems that rely on subtle texture for assessment. It is also important to understand that texture reconstruction appears to be worse in several cases on outer slices, though the PSNR, SSIM, and LPIPS on average is higher, so any strange matter or textures found in outer slices shouldn't be taken as gospel and must be subject to further review.

Importantly, our models appear to require a certain level of blurring to avoid black regions in the reconstructed images, these should naturally be present in 1.5T scans but it is important to ensure that any input 1.5T images match this level. These effects are unlikely to mislead radiologists as they are immediately obvious upon inspection. In this scenario, we recommend altering the radius of the low-pass filter to match levels of blur of input images, and removing $\mathcal{T}_2$ or $\mathcal{T}_3$ if these artefacts aren't present in real-life input images. Using this, a new simulated training dataset can be generated and our models can be retrained.

6.4 Future Work

This project was conducted over a period of around 4 months, but as we were operating using limited hardware, each training run took a substantial amount of time. As such, we did not implement all potential ideas.

6.4.1 Dataset

One simple implementation that we would have experimented with splitting the training dataset into one main subset and two comparatively smaller subsets. The primary subset would be degraded using only the low-pass filter $\mathcal{T}_1$, while the smaller subsets would be degraded using $\mathcal{T}_1$ combined with $\mathcal{T}_2$, and $\mathcal{T}_1$ combined with $\mathcal{T}_3$, respectively. This is because 1.5T scans are not guaranteed to contain either $\mathcal{T}_2$ or $\mathcal{T}_3$ artefacts. We expect that this would solve black region

issues mentioned in Sections 5.4.2 and 5.4.3. This was not done earlier due to time constraints.

6.4.2 Loss Functions

We had several drop-in replacement ideas for the perceptual loss function L_{perc}, which passed the SR and HR sets into the VGG19 CNN, using the MAE as the loss score. VGG19 is trained using the ImageNet dataset (Simonyan and Zisserman, 2015), which contains images not focused on the MRI domain. While the ImageNet weights do transfer well onto many other problems, it is worth investigating whether fine-tuning VGG19 on an MRI dataset may help the perceptual loss function prioritise features specific to MRI images.

In a similar vein, a theoretical segmentation loss function could be used to penalise large differences in matter type (WM, GM, and CSF) or tumour type by using feature maps from CNN-based MRI segmentation models. This can work in a similar way to L_{perc} using MAE. This type of loss function could be used as a drop-in replacement for L_{perc} or added alongside others in the composite loss function. The hope is for the generator to preserve anatomically meaningful structures and tissue boundaries.

6.4.3 Evaluation

Unfortunately, each epoch took around 9 hours using our training dataset, taking about a week to train. This meant that k-fold cross validation was out of the scope as it would take k times this amount of time. Using k-fold cross validation would allow us to train our models on the entire dataset and evaluate based on the entire dataset, giving us a model exposed to more data, which could generalise better as a result, and an evaluation using significantly more data, allowing for a more accurate assessment of model performance, with reduced bias from any particular train/test split.

Additionally, the matter segmentation evaluation (Section 5.2.2) was unusually poor at segmenting our LR slices, skewing Dice scores to favour our SR set. With more time, we could evaluate using multiple brain-matter segmentation models in addition to FSL-FAST, helping us build a more holistic evaluation.

6.4.4 Architecture Improvements

As mentioned briefly in the background, MRI data is inherently 3D. Therefore, a 3D super-resolution should theoretically be able to produce higher reconstruction performance by using inter-slice relationships. Our model could replace convolution, batch normalisation, and other layers with 3D variants, adapting the model to use an entire MRI volume as input. This obviously makes the model far larger, meaning that a single pass takes up more VRAM than a 3D model. Unfortunately, we were not able to use any of these approaches due to VRAM limitations.

Chapter 7

Reflection

I began this task with absolutely no knowledge of MRI physics. As such, even simple concepts such as 1.5T and 3T magnetic field strength were outside of my domain. Naturally I understood that some level of research was needed to understand the task, however I was not aware of how complex it would be to simply understand how a scan works. Initially, I imagined that simulating acquisition artefacts would simply boil down to blurring images and adding noise, but I soon became aware of the concept of k-space, which I was not able to understand conceptually, spurring me to dive into a week-long period dedicated to understanding MRI acquisition physics. This personal research was invaluable, not only helping me understand why prior works used certain steps in their degradation pipelines, but also allowing me to justify design choices for my own degradation pipeline, something that would never have been possible without domain knowledge. However, the field is still relatively new to me, and decisions are likely reflective of some naivety. I noticed that while I can understand details very well, sometimes I lose the bigger picture. For example, once I gained some in-depth MRI physics knowledge, I felt the need to apply every detail into my degradation pipeline without as much consideration into whether it would make the final image resemble 1.5T more. In reality, it is likely that including GRAPPA and Partial Fourier artefact simulation was unnecessary and a simpler approach using a low-pass filter in isolation would have yielded more realistic 1.5T simulation. But once I learned how GRAPPA and Partial Fourier worked I was far too eager to add them in without proper justification as to whether they were needed. In the future, I recognise that learning technical details is important, but it is equally important to critically assess their relevance to the overall project.

Most of the limitations in this project were caused by a lack of time as training took so long. Before taking this project, I had undertaken some small-scale machine learning projects, however the scale of this project was more complex and required far more experimentation and research than other projects. I also had no experience using GANs, or the field of super-resolution. As such, I was not prepared for how long training loops would take to complete, or how much hyperparameter tuning would be needed. On top of this, a lot of my time was spent monitoring the training and doing nothing simultaneously, wasting some time in the initial stages of the project. Towards project's tail-end, I had a number of personal commitments, triggering me to create a schedule

midway through. Taking into account training time for our models and report-writing, I found out that I had less time than I initially imagined. This led me to start writing my report and monitoring training simultaneously, which if done earlier would have saved time significantly and potentially have alleviated some of the limitations caused by time constraints. While I don't think that my time management was bad, as I correctly scheduled this project to be done within the deadline, next time I would improve it by considering multitasking and my knowledge of GAN training time from the beginning.

Approaching this task, I did not consider that the size of the BraTS dataset would affect the project in as significant of a way as it ended up doing. Early on, I decided that I wanted to give my models as much data variety as possible, which led me to use as large of a dataset as reasonably possible. Other models in the literature used far fewer volumes (such as Ayaz *et al.* (2024) using 60 scans) that intuitively did not seem like enough for the task. However, whether using a very large training dataset led to a true benefit was difficult to prove definitively and still is unclear. Ironically, I might have done more experimentation to prove this if I restricted the training dataset size and iteratively added more data whilst keeping the maximum validation set size, but this did not happen as I spent most of my time training my models on the full training dataset. Additionally, if I had not rushed into the assumption that I would use all slices from all volumes I could have investigated if training with a random subset of slices from each volume produced a similar performance. This is mainly an issue due to my time constraints, optimising for time could have potentially led to a better solution. Ultimately, it is unlikely that using more BraTS data harmed my project directly and it is possible that it helped, but the issue was that I hadn't fully investigated the effects. Moving forward I would spend more time investigating the benefit of dataset size.

All else equal, I believe that I conducted the research and task to a high standard. While the evaluation data demonstrates the quality of my results, the topics and skills I learned evidence the thoroughness of my research process. Although CMT311 and my own personal projects gave me a strong foundation in CNN architectures, training, and the mathematics underlying machine learning, I had no prior understanding of GANs. However, by the end of the project I felt confident in both GAN theory and practical implementation, despite GAN training being notoriously unstable. This was also my first larger-scale ML-focused academic project. I also developed my academic presentation skills, as this was the first dissertation I fully typeset using LaTeXrather than Microsoft Word. Prior to my Master's degree, I had no experience with typesetting, mathematical notation, or LaTeX, only beginning to learn these skills during the course. This dissertation was therefore my first opportunity to apply them on a larger scale.

On the whole, this project has been very engaging for me. Before choosing to pursue computer science and machine learning, I had an affinity for studying physics and this project allowed me to reconnect with that interest using skills I have developed throughout my degree. To date, this project has likely been the most complex challenge that I have completed, forcing me to research complicated topics, present and report data in a clear manner, reflect critically on my choices, and think creatively. These skills will undoubtedly stay with me as I leave the academic environment and move into the professional world.

Bibliography

Anazodo, U., Ng, J., Ehiogu, B., Obungoloch, J., Fatade, A., Mutsaerts, H., Diop, M., Opadele, A., Alexander, D., Dada, M., Ogbole, G., Nunes, R., Figueiredo, P., Aribisala, B., Awojoyogbe, B., Aduluwa, H., Sprenger, C. and Dako, F. 2022. A framework for advancing sustainable MRI access in Africa. *NMR in Biomedicine* 36.

Anchuri, A. 2011. *Image blur metrics*, [Online]. MS Report, Stanford University, PSYCH 221, Ref: Dr. Joyce Farrell, Available at: `https://acorn.stanford.edu/psych221/projects/2010/AdityaAnchuri/main.pdf`. Accessed: 2025-08-29.

Andrew, J., Mhatesh, T. S. R., Sebastin, R. D., Sagayam, K. M., Eunice, J., Pomplun, M. and Dang, H. 2021. Super-resolution reconstruction of brain magnetic resonance images via lightweight autoencoder. *Informatics in Medicine Unlocked* 26, p. 100713. Available at: `https://www.sciencedirect.com/science/article/pii/S2352914821001945`.

Arjovsky, M., Chintala, S. and Bottou, L. 2017. *Wasserstein GAN*, [Online]. Available at: `https://arxiv.org/abs/1701.07875`.

Arnx, A. 2019. First neural network for beginners explained (with code). *Towards Data Science (Medium)* Available at: `https://medium.com/data-science/first-neural-network-for-beginners-explained-with-code-4cfd37e06eaf`.

Ayaz, A., Boonstoppel, R., Lorenz, C., Weese, J., Pluim, J. and Breeuwer, M. 2024. Effective deep-learning brain MRI super resolution using simulated training data. *Computers in Biology and Medicine* 183, p. 109301. Available at: `https://www.sciencedirect.com/science/article/pii/S0010482524013866`.

Bachmann, R., Reilmann, R., Schwindt, W., Kugel, H., Heindel, W. and Krämer, S. 2006. FLAIR imaging for multiple sclerosis: a comparative MR study at 1.5 and 3.0 Tesla. *European Radiology* 16(4), pp. 915–921. Available at: `https://www.ncbi.nlm.nih.gov/pubmed/16365731`.

Bai, J., Yuan, L., Xia, S.-T., Yan, S., Li, Z. and Liu, W. 2022. Improving vision transformers by revisiting high-frequency components. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G. M. and Hassner, T., eds., *Computer Vision – ECCV 2022*. Cham: Springer Nature Switzerland, pp. 1–18, Available at: `https://github.com/jiawangbai/HAT`.

Baid, U., Ghodasara, S., Mohan, S., Bilello, M., Calabrese, E., Colak, E., Farahani, K., Kalpathy-Cramer, J., Kitamura, F. C., Pati, S., Prevedello, L. M., Rudie, J. D., Sako, C., Shinohara, R. T., Bergquist, T., Chai, R., Eddy, J., Elliott, J., Reade, W., Schaffter, T., Yu, T., Zheng, J., Moawad, A. W., Coelho, L. O., McDonnell, O., Miller, E., Moron, F. E., Oswood, M. C., Shih, R. Y., Siakallis, L., Bronstein, Y., Mason, J. R., Miller, A. F., Choudhary, G., Agarwal, A., Besada, C. H., Derakhshan, J. J., Diogo, M. C., Do-Dai, D. D., Farage, L., Go, J. L., Hadi, M., Hill, V. B., Iv, M., Joyner, D., Lincoln, C., Lotan, E., Miyakoshi, A., Sanchez-Montano, M., Nath, J., Nguyen, X. V., Nicolas-Jilwan, M., Jimenez, J. O., Ozturk, K., Petrovic, B. D., Shah, C., Shah, L. M., Sharma, M., Simsek, O., Singh, A. K., Soman, S., Statsevych, V., Weinberg, B. D., Young, R. J., Ikuta, I., Agarwal, A. K., Cambron, S. C., Silbergleit, R., Dusoi, A., Postma, A. A., Letourneau-Guillon, L., Perez-Carrillo, G. J. G., Saha, A., Soni, N., Zaharchuk, G., Zohrabian, V. M., Chen, Y., Cekic, M. M., Rahman, A., Small, J. E., Sethi, V., Davatzikos, C., Mongan, J., Hess, C., Cha, S., Villanueva-Meyer, J., Freymann, J. B., Kirby, J. S., Wiestler, B., Crivellaro, P., Colen, R. R., Kotrotsou, A., Marcus, D., Milchenko, M., Nazeri, A., Fathallah-Shaykh, H., Wiest, R., Jakab, A., Weber, M.-A., Mahajan, A., Menze, B., Flanders, A. E. and Bakas, S. 2021. *The RSNA-ASNR-MICCAI BraTS 2021 benchmark on brain tumor segmentation and radiogenomic classification*, [Online]. Available at: `https://arxiv.org/abs/2107.02314`.

Bernstein, M. A., Huston III, J. and Ward, H. A. 2006. Imaging artifacts at 3.0T. *Journal of Magnetic Resonance Imaging* 24(4), pp. 735–746. Available at: `https://onlinelibrary.wiley.com/doi/abs/10.1002/jmri.20698`.

Blaimer, M., Breuer, F., Mueller, M., Heidemann, R., Griswold, M. and Jakob, P. 2004. SMASH, SENSE, PILS, GRAPPA: how to choose the optimal method. *Topics in magnetic resonance imaging : TMRI* 15, pp. 223–36.

Bottomley, P. A., Foster, T. H., Argersinger, R. E. and Pfeifer, L. M. 1984. A review of normal tissue hydrogen NMR relaxation times and relaxation mechanisms from 1–100 MHz: dependence on tissue type, NMR frequency, temperature, species, excision, and age. *Medical Physics* 11(4), pp. 425–448. Available at: `https://doi.org/10.1118/1.595535`.

Chang, Y., Pham, H. A. and Li, Z. 2022. A dual-interpolator method for improving parallel MRI reconstruction. *Magnetic Resonance Imaging* 92, pp. 108–119. Available at: `https://www.sciencedirect.com/science/article/pii/S0730725X22001035`.

Chuang, C.-C., Lee, Y.-T., Chen, C.-M., Hsieh, Y.-S., Liu, T.-C. and Sun, C.-W. 2012. Patient-oriented simulation based on Monte Carlo algorithm by using MRI data. *Biomedical engineering online* 11, p. 21.

Corbin, N., Miraux, S., Ozenne, V., Émeline Ribot and Trotier, A. 2025. *Fast imaging and acceleration techniques*, [Online]. Available at: `https://radiologykey.com/fast-imaging-and-acceleration-techniques/`. CRMSB, CNRS, Université de Bordeaux, France.

Cummings, E., Macdonald, J. A. and Seiberlich, N. 2022. Chapter 6 - Parallel imaging. In: Akçakaya, M., Doneva, M. and Prieto, C.,

eds., *Magnetic Resonance Image Reconstruction*, Academic Press, vol. 7 of *Advances in Magnetic Resonance Technology and Applications*, pp. 129–157. Available at: https://www.sciencedirect.com/science/article/pii/B9780128227268000166.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Li, F.-F. 2009. ImageNet: A large-scale hierarchical image database. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. pp. 248–255.

Dong, C., Loy, C. C., He, K. and Tang, X. 2015. *Image super-resolution using deep convolutional networks*, [Online]. Available at: https://arxiv.org/abs/1501.00092.

Dong, C., Loy, C. C. and Tang, X. 2016. *Accelerating the super-resolution convolutional neural network*, [Online]. Available at: https://arxiv.org/abs/1608.00367.

Edelstein, W. A., Glover, G. H., Hardy, C. J. and Redington, R. W. 1986. The intrinsic signal-to-noise ratio in NMR imaging. *Magnetic Resonance in Medicine* 3(4), pp. 604–618. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.1910030413.

Felix, R., Schörner, W., Laniado, M., Niendorf, H. P., Claussen, C., Fiegler, W. and Speck, U. 1985. Brain tumors: MR imaging with Gadolinium-DTPA. *Radiology* 156(3), pp. 681–688.

Frost, R., Porter, D. A., Feiweier, T. and Jezzard, P. 2010. Homodyne reconstruction of partial fourier readout-segmented EPI for diffusion imaging. In: *Proceedings of the 18th Annual Meeting of ISMRM*. Stockholm, Sweden: International Society for Magnetic Resonance in Medicine, p. 1625, Available at: https://archive.ismrm.org/2010/1625.html.

Fu, T., Zhang, J., Sun, R., Huang, Y., Xu, W., Yang, S., Zhu, Z. and Chen, H. 2024. Optical neural networks: progress and challenges. *Light: Science & Applications* 13(1), p. 263. Available at: https://doi.org/10.1038/s41377-024-01590-3.

Gatys, L. A., Ecker, A. S. and Bethge, M. 2015. A neural algorithm of artistic style. *CoRR* abs/1508.06576. Available at: http://arxiv.org/abs/1508.06576.

Global Insight Services. 2025. *Next-gen MRI scanners market analysis and forecast to 2034*, [Online]. Available at: https://www.globalinsightservices.com/reports/next-gen-mri-scanners-market. Accessed: 2025-08-25.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. 2014. *Generative adversarial networks*, [Online]. Available at: https://arxiv.org/abs/1406.2661.

Griswold, M. A., Jakob, P. M., Heidemann, R. M., Nittka, M., Jellus, V., Wang, J., Kiefer, B. and Haase, A. 2002. Generalized autocalibrating partially parallel acquisitions (GRAPPA). *Magnetic Resonance in Medicine* 47(6), pp. 1202–1210. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.10171.

Gudbjartsson, H. and Patz, S. 1995. The Rician distribution of noisy MRI data. *Magnetic Resonance in Medicine* 34(6), pp. 910–914. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.1910340618.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V. and Courville, A. 2017. *Improved training of Wasserstein GANs*, [Online]. Available at: https://arxiv.org/abs/1704.00028.

Hajnal, J. V., Decoene, B., Lewis, P. D., Baudouin, C. J., Cowan, F. M., Pennock, J. M., Young, I. R. and Bydder, G. M. 1992. High signal regions in normal white matter shown by heavily T2-weighted CSF-nulled IR sequences. *Journal of Computer Assisted Tomography* 16(4), pp. 506–513.

Haldar, J. P. and Liang, Z.-P. 2022. Chapter 5 - Early constrained reconstruction methods. In: Akçakaya, M., Doneva, M. and Prieto, C., eds., *Magnetic Resonance Image Reconstruction*, Academic Press, vol. 7 of *Advances in Magnetic Resonance Technology and Applications*, pp. 105–125. Available at: https://www.sciencedirect.com/science/article/pii/B9780128227268000142.

Ham, H., Jun, T. J. and Kim, D. 2020. Unbalanced GANs: Pre-training the generator of generative adversarial network using variational autoencoder. *CoRR* abs/2002.02112. Available at: https://arxiv.org/abs/2002.02112.

Hashemi, R. H., Bradley, W. G., Chen, D. Y., Jordan, J. E., Queralt, J. A., Cheng, A. E. and Henrie, J. N. 1995. Suspected multiple sclerosis: MR imaging with a thin-section fast FLAIR pulse sequence. *Radiology* 196(2), pp. 505–510. Available at: https://doi.org/10.1148/radiology.196.2.7617868. PMID: 7617868.

Hu, J., Gu, X. and Gu, X. 2020. Dual-pathway DenseNets with fully lateral connections for multimodal brain tumor segmentation. *International Journal of Imaging Systems and Technology* 31.

Hua, R., Huo, Q., Gao, Y., Sui, H., Zhang, B., Sun, Y., Mo, Z. and Shi, F. 2020. Segmenting brain tumor using cascaded V-Nets in multimodal MR images. *Frontiers in Computational Neuroscience* 14. Available at: https://www.frontiersin.org/journals/computational-neuroscience/articles/10.3389/fncom.2020.00009.

Huang, J., Fang, Y., Wu, Y., Wu, H., Gao, Z., Li, Y., Ser, J. D., Xia, J. and Yang, G. 2022. *Swin transformer for fast MRI*, [Online]. Available at: https://arxiv.org/abs/2201.03230.

Hyun, C. M., Kim, H. P., Lee, S. M., Lee, S. and Seo, J. K. 2018. Deep learning for undersampled MRI reconstruction. *Physics in Medicine and Biology* 63(13), p. 135007. Available at: http://dx.doi.org/10.1088/1361-6560/aac71a.

Jamwal, A. 2025. *MRI systems market size, share, and growth forecast from 2025 - 2032*, [Online]. Available at: https://www.persistencemarketresearch.com/market-research/mri-systems-market.asp. Accessed: 07-06-2025.

Kozlov, S., Kolesnytskyi, O., Korolenko, O., Zhukov, A., Bondarenko, D., Smetaniuk, O., Kalizhanova, A. and Komada, P. 2024. Transformers in image super-resolution: a brief review. In: Romaniuk, R. S., Smolarz, A. and Wójcik, W., eds., *Photonics Applications in Astronomy, Communications, Industry, and High Energy Physics Experiments 2024*. International Society for Optics and Photonics, SPIE, vol. 13400, p. 134000J, Available at: `https://doi.org/10.1117/12.3054855`.

Kuhl, C. K., Textor, J., Gieseke, J., von Falkenhausen, M., Gernert, S., Urbach, H. and Schild, H. H. 2005. Acute and subacute ischemic stroke at high-field-strength (3.0T) diffusion-weighted MR imaging: Intraindividual comparative study. *Radiology* 234(2), pp. 509–516. Available at: `https://doi.org/10.1148/radiol.2342031323`. PMID: 15601894.

Kumar, P. S. and Dharun, V. S. 2016. A study of MRI segmentation methods in automatic brain tumor detection. *International Journal of Engineering and Technology* 8, pp. 609–614.

Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11), pp. 2278–2324.

Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z. and Shi, W. 2017. *Photo-realistic single image super-resolution using a generative adversarial network*, [Online]. Available at: `https://arxiv.org/abs/1609.04802`.

Li, H. B., Conte, G. M., Hu, Q., Anwar, S. M., Kofler, F., Ezhov, I., van Leemput, K., Piraud, M., Diaz, M., Cole, B., Calabrese, E., Rudie, J., Meissen, F., Adewole, M., Janas, A., Kazerooni, A. F., LaBella, D., Moawad, A. W., Farahani, K., Eddy, J., Bergquist, T., Chung, V., Shinohara, R. T., Dako, F., Wiggins, W., Reitman, Z., Wang, C., Liu, X., Jiang, Z., Familiar, A., Johanson, E., Meier, Z., Davatzikos, C., Freymann, J., Kirby, J., Bilello, M., Fathallah-Shaykh, H. M., Wiest, R., Kirschke, J., Colen, R. R., Kotrotsou, A., Lamontagne, P., Marcus, D., Milchenko, M., Nazeri, A., Weber, M.-A., Mahajan, A., Mohan, S., Mongan, J., Hess, C., Cha, S., Villanueva-Meyer, J., Colak, E., Crivellaro, P., Jakab, A., Albrecht, J., Anazodo, U., Aboian, M., Yu, T., Baid, U., Bakas, S., Linguraru, M. G., Menze, B., Iglesias, J. E. and Wiestler, B. 2024. The brain tumor segmentation (BraTS) challenge 2023: Brain MR image synthesis for tumor segmentation (BraSyn). *arXiv preprint arXiv:2305.09011v6* Available at: `https://arxiv.org/abs/2305.09011v6`. Preprint.

Li, Y., Liu, Y., Liu, Y., Qiao, Y. and He, J. 2023. Multi-level feature extraction and edge reconstruction fused generative adversarial networks for image super resolution. pp. 113–120.

Liao, B., Chen, Y., Wang, Z., Smith, C. D. and Liu, J. 2022. *A comparative study on 1.5T-3T MRI conversion through deep neural network models*, [Online]. Available at: `https://arxiv.org/abs/2210.06362`.

Liu, R. W., Shi, L., Huang, W., Xu, J., Yu, S. C. H. and Wang, D. 2014. Generalized total variation-based MRI Rician denoising model with spatially

adaptive regularization parameters. *Magnetic Resonance Imaging* 32(6), pp. 702–720. Available at: https://www.sciencedirect.com/science/article/pii/S0730725X14000861.

Lu, Z. and Chen, Y. 2019. *Single image super resolution based on a modified U-net with mixed gradient loss*, [Online]. Available at: https://arxiv.org/abs/1911.09428.

Moratal, D., Vallés-Luch, A., Marti-Bonmati, L. and Brummer, M. 2008. k-space tutorial: An MRI educational tool for a better understanding of k-space. *Biomedical imaging and intervention journal* 4, p. e15.

MRI Master. 2023. *MRI parallel imaging artifacts*, [Online]. MRI Master, Available at: https://mrimaster.com/parallel-imaging-artifact/.

Muhammad, A., Aramvith, S., Duangchaemkarn, K. and Sun, M.-T. 2024. Brain MRI image super-resolution reconstruction: A systematic review. *IEEE Access* 12, pp. 156347–156362.

Otazo, R., Kim, D., Axel, L. and Sodickson, D. K. 2010. Combination of compressed sensing and parallel imaging for highly accelerated first-pass cardiac perfusion MRI. *Magnetic Resonance in Medicine* 64(3), pp. 767–776. Available at: https://doi.org/10.1002/mrm.22463. PMCID: PMC2932824.

Pambrun, J.-F. and Noumeir, R. 2015. Limitations of the SSIM quality metric in the context of diagnostic imaging. In: *2015 IEEE International Conference on Image Processing (ICIP)*. pp. 2960–2963.

Puzio, T., Matera, K., Karwowski, J., Piwnik, J., Białkowski, S., Podyma, M., Dunikowski, K., Siger, M., Stasiołek, M., Grzelak, P. and Bobeff, E. J. 2025. Deep learning-based automatic segmentation of brain structures on MRI: A test-retest reproducibility analysis. *Computational and Structural Biotechnology Journal* 28, pp. 128–140. Available at: https://www.sciencedirect.com/science/article/pii/S200103702500128X.

Rosenblatt, F. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review* 65 6, pp. 386–408. Available at: https://api.semanticscholar.org/CorpusID:12781225.

Rumelhart, D. E., Hinton, G. E. and Williams, R. J. 1986. Learning representations by back-propagating errors. *Nature* 323, pp. 533–536. Available at: https://api.semanticscholar.org/CorpusID:205001834.

Sanchez, I. and Vilaplana, V. 2018. *Brain MRI super-resolution using 3D generative adversarial networks*, [Online]. Available at: https://arxiv.org/abs/1812.11440.

Sharif, S. M. A., Naqvi, R. A. and Loh, W.-K. 2024. Two-stage deep denoising with self-guided noise attention for multimodal medical images. *IEEE Transactions on Radiation and Plasma Medical Sciences* 8(5), p. 521–531. Available at: http://dx.doi.org/10.1109/TRPMS.2024.3380090.

Shi, F., Cheng, J., Wang, L., Yap, P.-T. and Shen, D. 2015. LRTV: MR image super-resolution with low-rank and total variation regularizations. *IEEE Transactions on Medical Imaging* 34(12), pp. 2459–2466.

Shi, J., Li, Z., Ying, S., Wang, C., Liu, Q., Zhang, Q. and Yan, P. 2019. MR image super-resolution via wide residual networks with fixed skip connection. *IEEE Journal of Biomedical and Health Informatics* 23(3), pp. 1129–1140.

Simonyan, K. and Zisserman, A. 2015. *Very deep convolutional networks for large-scale image recognition*, [Online]. Available at: `https://arxiv.org/abs/1409.1556`.

Smith, A. S., Weinstein, M. A., Modic, M. T., Pavlicek, W., Rogers, L. R., Budd, T. G., Bukowski, R. M., Purvis, J. D., Weick, J. K. and Duchesneau, P. M. 1985. Magnetic resonance with marked T2-weighted images: improved demonstration of brain lesions, tumor, and edema. *AJR Am J Roentgenol* 145(5), pp. 949–955.

Sodickson, A. D. and Sodickson, D. K. 2016. Chapter 18 - Introductory magnetic resonance imaging physics. In: Newton, H. B., ed., *Handbook of Neuro-Oncology Neuroimaging (Second Edition)*, San Diego: Academic Press, pp. 157–166. Second edition ed., Available at: `https://www.sciencedirect.com/science/article/pii/B9780128009451000185`.

Soher, B. J., Dale, B. M. and Merkle, E. M. 2007. A review of MR physics: 3T versus 1.5T. *Magnetic Resonance Imaging Clinics of North America* 15(3), pp. 277–290. Available at: `https://www.sciencedirect.com/science/article/pii/S1064968907000773`. Body MR Imaging: 1.5T versus 3T.

Stanford University. 2011. *Noise review*, [Online]. `https://physbam.stanford.edu/cs448x/old/Noise_Review.html`. Accessed: 2025-08-05.

Tajima, T., Akai, H., Yasaka, K., Kunimatsu, A., Yamashita, Y., Akahane, M., Yoshioka, N., Abe, O., Ohtomo, K. and Kiryu, S. 2023. Usefulness of deep learning-based noise reduction for 1.5T MRI brain images. *Clinical Radiology* 78(1), pp. e13–e21. Available at: `https://doi.org/10.1016/j.crad.2022.08.127`.

Tao, S., Trzasko, J. D., Shu, Y., Weavers, P. T., Huston III, J., Gray, E. M. and Bernstein, M. A. 2016. Partial fourier and parallel MR image reconstruction with integrated gradient nonlinearity correction. *Magnetic Resonance in Medicine* 75(6), pp. 2534–2544. Available at: `https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.25842`.

Truax, B. 1999. *Handbook for acoustic ecology*, [Online]. Online HTML/CD-ROM edition, Available at: `https://www.sfu.ca/sonic-studio-webdav/handbook/Gaussian_Noise.html`. Second Edition, image/figure reproduced online.

U, N. and M., A. P. 2024. MRI super-resolution using similarity distance and multi-scale receptive field based feature fusion GAN and pre-trained slice interpolation network. *Magnetic Resonance Imaging* 110, pp. 195–209. Available at: `https://www.sciencedirect.com/science/article/pii/S0730725X24001346`.

Wang, J., Chen, Y., Wu, Y., Shi, J. and Gee, J. 2020. Enhanced generative adversarial network for 3D brain mri super-resolution. In: *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*. pp. 3616–3625.

Wang, L., Du, J., Gholipour, A., He, Z. and Jia, Y. 2019. Brain MRI super-resolution reconstruction using a multi-level and parallel conv-deconv network. In: *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. pp. 885–891.

Wang, X. 2018. *ESRGAN official repository*, [Online]. `https://github.com/xinntao/ESRGAN/blob/master/RRDBNet_arch.py`. Accessed: 2025-07-25.

Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Loy, C. C., Qiao, Y. and Tang, X. 2018. ESRGAN: Enhanced super-resolution generative adversarial networks. *CoRR* abs/1809.00219. Available at: `http://arxiv.org/abs/1809.00219`.

Wang, Z. and Bovik, A. C. 2009. Mean squared error: Love it or leave it? a new look at signal fidelity measures. *IEEE Signal Processing Magazine* 26(1), pp. 98–117.

Wang, Z., Bovik, A. C., Sheikh, H. R. and Simoncelli, E. P. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing* 13(4), pp. 600–612.

Wolfram Research. 2024. *TukeyWindow — Wolfram language documentation*, [Online]. `https://reference.wolfram.com/language/ref/TukeyWindow.html`. Accessed: 2025-07-22.

Wong, S., Steinbach, L., Zhao, J., Stehling, C., Ma, C. B. and Link, T. M. 2009. Comparative study of imaging at 3.0T versus 1.5T of the knee. *Skeletal Radiology* 38(8), pp. 761–769. Available at: `https://doi.org/10.1007/s00256-009-0683-0`.

Wong, Y. M., Yeap, P. L., Ong, A. L. K., Tuan, J. K. L., Lew, W. S., Lee, J. C. L. and Tan, H. Q. 2024. Machine learning prediction of dice similarity coefficient for validation of deformable image registration. *Intelligence-Based Medicine* 10, p. 100163. Available at: `https://www.sciencedirect.com/science/article/pii/S2666521224000309`.

Yang, G., Zhang, L., Zhou, M., Liu, A., Chen, X., Xiong, Z. and Wu, F. 2022. Model-guided multi-contrast deep unfolding network for mri super-resolution reconstruction. In: *Proceedings of the 30th ACM International Conference on Multimedia*. ACM, MM'22, pp. 3974–3982, Available at: `http://dx.doi.org/10.1145/3503161.3548068`.

Yosinski, J., Clune, J., Bengio, Y. and Lipson, H. 2014. How transferable are features in deep neural networks? *CoRR* Available at: `http://arxiv.org/abs/1411.1792`.

Zeineldin, R. A., Karar, M. E., Coburger, J., Wirtz, C. R. and Burgert, O. 2020. DeepSeg: deep neural network framework for automatic brain tumor segmentation using magnetic resonance flair images. *International Journal of Computer Assisted Radiology and Surgery* 15(6), p. 909–920. Available at: `http://dx.doi.org/10.1007/s11548-020-02186-z`.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E. and Wang, O. 2018. The unreasonable effectiveness of deep features as a perceptual metric. *CoRR* abs/1801.03924. Available at: `http://arxiv.org/abs/1801.03924`.

Zhang, Y., Brady, M. and Smith, S. 2001. Segmentation of brain MR images through a hidden Markov random field model and the expectation-maximization algorithm. *IEEE Transactions on Medical Imaging* 20(1), pp. 45–57.

Zhao, H., Gallo, O., Frosio, I. and Kautz, J. 2017. Loss functions for image restoration with neural networks. *IEEE Transactions on Computational Imaging* 3(1), pp. 47–57.

Zhu, J., Yang, G. and Lio, P. 2019. *How can we make GAN perform better in single medical image super-resolution? a lesion focused multi-scale approach*, [Online]. Available at: `https://arxiv.org/abs/1901.03419`.

Appendix A

Source Code

The full source code for this project can be found on GitHub:
`https://github.com/theobaur13/MRI-Super-Resolution`