Data Standardization, Multi-Domain Learning and GRAPPA preprocessing for Improved MRI

Scientific thesis for the procurance of the degree B.Sc.
from the Department of Electrical and Computer Engineering at the Technical University of Munich.

Supervised by Prof. Dr.sc ETH Zürich Reinhard Heckel

Professur für Machine Learning

Submitted by Mohamed Amine Ketata

Submitted on Munich, 24.06.2021

Abstract

Accelerating Magnetic Resonance Imaging (MRI) is one of the major goals in many MRI applications. It has been addressed by using under-sampled k-space reconstruction techniques that have enabled image reconstruction with reduced acquisition time and costs. While traditional methods are based on handcrafted image models and classical signal processing methods, more recently, new machine learning approaches for MR image reconstruction have been proposed. A recent competition, the fast MRI challenge, provided a framework to study and benchmark deep learning based image reconstruction methods. In this work, we consider the methods proposed by the winning team of the fastMRI challenge 2020 version, the AIRS medical team. Their work focused on two key factors: data standardization and multi-domain data processing. We re-implement these methods in order to validate the reported results. Additionally, we study the effect of applying a classical image reconstruction algorithm prior to forwarding the data through a neural network model. Our experiments show that the data standardization and multi-domain data processing methods decrease the performance with different amounts compared to the baseline model. On the other hand, preprocessing the data with the GRAPPA algorithm increases the performance of the baseline model consistently across all metrics.

Contents

Chapter 1

Introduction

Magnetic resonance imaging (MRI) is a powerful diagnostic tool for many types of diseases. However, due to the long acquisition time in MRI, which may exceed 30 minutes, and its high exam costs, MRI is currently not practically applicable in a wide range of applications. Therefore, accelerating MRI has been an active area of research since the 1970s, when MRI was first invented. One very important development in this context is *compressed sensing* (CS), introduced in 2006 in the papers [CRT06] and [LDP07]. The basic idea of CS techniques is to speed up the MR acquisition by taking fewer measurements than has previously been required to reconstruct diagnostic quality images. However, this leads to violating the Nyquist-Shannon sampling theorem and therefore aliasing artifacts are introduced which must be eliminated in the course of image reconstruction. In the last few years, machine learning methods have been found to be very effective at this process and many new methods have been developed for MR reconstruction. Recently, the fastMRI dataset, a large-scale, public dataset of raw and processed MRI data has been released along with the paper [ZKS+18] with the goal of accelerating the development of the machine learning based reconstruction of accelerated MR images. In addition to the release of this dataset, two versions of the fastMRI competition have been organized since 2019. Before discussing the proposed methods in more detail, we begin with a brief introduction to the image reconstruction task, the dataset and the baseline model.

1.1 Task Description

MR imaging is an indirect process; instead of direct, spatially resolved measurements, frequency and phase measurements are performed, based on which cross-sectional images of the subject's anatomy are produced. These measurements correspond to points in the multidimensional Fourier-space representation of an imaged body, known as k-space. The spatially resolved image m can be estimated from the

full k-space y by performing an inverse multidimensional Fourier transform

$$\hat{m} = F^{-1}(y), \tag{1.1}$$

where $\hat{m}$ is a noise-corrupted estimate of the true image m. In real-world applications, multiple receiver coils are used, each of which produces a separate k-space measurement matrix. Each coil measures Fourier components of the imaged volume multiplied by a complex-valued position-dependent coil sensitivity map S_i. The measured k-space signal y_i for coil i in an array of n_c coils is given by

$$y_i = F(S_i m) + noise. \tag{1.2}$$

The fastMRI dataset provides input/output pair tuples (y, m) to train machine learning models. The dataset contains amongst other things the following types of data from MRI acquisition of knees and brains:

- Raw multi-coil k-space data: unprocessed complex-valued multi-coil MR measurements.
- Ground-truth images: real-valued images reconstructed from fully-sampled multi-coil acquisitions using the root-sum-of-squares method. These images are produced by first applying the inverse Fourier Transform to the k-space data from each coil

  $$m_i = F^{-1}(y_i), \tag{1.3}$$

  where y_i is the k-space data from the i-th coil and m_i is the i-th coil image. Then, the individual coil images are combined voxel by voxel as follows

  $$m_{rss} = \left(\sum_{i=0}^{n_c} |m_i|^2 \right)^{\frac{1}{2}}. \tag{1.4}$$

The multi-coil reconstruction task consists therefore of reconstructing images approximating the ground-truth from under sampled multi-coil data.

1.1.1 Metrics

In this section, we review the definitions of the metrics used to compare the different approaches for the MRI reconstruction task, as recommended in the fastMRI paper [ZKS+18].

- **Normalized Mean Squared Error**

  The *normalized mean square error* (NMSE) between a reconstructed image or image volume represented as a vector $\hat{v}$ and a reference image or volume v is defined as

  $$NMSE(\hat{v}, v) = \frac{\|\hat{v} - v\|_2^2}{\|v\|_2^2}. \tag{1.5}$$

- **Peak Signal-to-Noise Ratio**

  The *peak signal-to-noise ratio* (PSNR) represents the ratio between the power of the maximum possible image intensity across a volume and the power of distorting noise and other errors:

$$PSNR(\hat{v}, v) = 10log_{10}\frac{max(v)^2}{MSE(\hat{v}, v)}, \tag{1.6}$$

  where $MSE(\hat{v}, v) = \frac{1}{n}||\hat{v} - v||_2^2$ and n is the number of entries in the target volume v. Higher values of PSNR (as opposed to lower values of NMSE) indicate a better reconstruction.

- **Structural Similarity**

  The *structural similarity* (SSIM) index measures the similarity between two images by exploiting the inter-dependencies among nearby pixels. SSIM is inherently able to evaluate structural properties of the objects in an image and is computed at different image locations by using a sliding window. The resulting similarity between two image patches $\hat{m}$ and m is defined as

$$SSIM(\hat{m}, m) = \frac{(2\mu_{\hat{m}}\mu_m + c_1)(2\sigma_{\hat{m}m} + c_2)}{(\mu_{\hat{m}}^2 + \mu_m^2 + c_1)(\sigma_{\hat{m}}^2 + \sigma_m^2 + c_2)}, \tag{1.7}$$

  where $\mu_{\hat{m}}$ and μ_m are the average pixel intensities in $\hat{m}$ and m, $\sigma_{\hat{m}}^2$ and σ_m^2 are their variances, $\sigma_{\hat{m}m}$ is the covariance between $\hat{m}$ and m and c_1 and c_2 are two variables to stabilize the division; $c1 = (k_1L)^2$ and $c2 = (k2L)^2$. For SSIM values reported in this work, we choose a window size of 7×7, we set $k_1 = 0.01$, $k_2 = 0.03$, and define L as the maximum value of the target volume, $L = max(v)$.

- L1 Error

  We use the L1 loss

$$L_1(\hat{v}, v) = ||\hat{v} - v||_1 \tag{1.8}$$

  to train our models.

1.2 Baseline Model

We use the U-Net model which was first propsed in the paper [RFB15] as our baseline model, whose architecture is depicted in Figure 1.1. The U-Net consists of two deep convolutional networks, a down-sampling path followed by an up-sampling path. The down-sampling path consists of blocks of two 3×3 convolutions each followed by instance normalization and Rectified Linear Unit (ReLU) activation functions. The blocks are interleaved by down-sampling operations consisting of max-pooling layers with stride 2 which halve each spatial dimension. The

up-sampling path consists of blocks with a similar structure to the down-sampling path, interleaved with bilinear up-sampling layers which double the resolution between blocks. Each block consists of two 3×3 convolutions with instance normalization and ReLU activation layers. In contrast to the down-sampling path, the up-sampling path concatenates two inputs to the first convolution in each block: the up-sampled activations from the previous block, together with the activations that follow the skip connection from the block in the down-sampling path with the same resolution (horizontal arrows in Figure 1.1). At the end of the up-sampling path, we include a series of 1×1 convolutions that reduce the number of channels to one without changing the spatial resolution. In the multi-coil MRI reconstruction task, we have one set of under sampled k-space measurements from each coil, and a different zero-filled image can be computed from each coil. The zero-filled image of coil i is obtained by first inserting zeros at the location of all unobserved k-space values, applying a two-dimensional Inverse Fourier Transform (IFT) to the result, and finally computing the absolute value. The result is center cropped to remove any readout and phase oversampling. Using the notation from Section 1.1, the zero-filled image from coil i is given by $\tilde{m}_i = C(|F^{-1}(P(m))|)$, where C is the linear operator corresponding to the center cropping and F^{-1} is the two-dimensional IFT. These coil images are combined using the root-sum-of-squares algorithm (Equation (1.4)) to get the combined image $\tilde{m}_{rss}$, which is used as the input to the model. The entire network is trained on the training data in an end-to-end manner to minimize the mean absolute error with respect to corresponding ground truth images, which are computed as described in Section 1.1. Let $B_\theta(m)$ be the function computed by the U-Net model, where θ represents the parameters of the model. Then the training process corresponds to the following optimization problem

$$minimize_\theta \frac{1}{2} \sum_{i=0}^{n_{data}} \|B_\theta(\tilde{m}_{rss}^{(i)}) - m^{(i)}\|_1, \tag{1.9}$$

where $m^{(i)}$ are the ground truth images. During training, we randomly sampled a different mask for each training example in each epoch independently (using the protocol described in Section 4.9 in the paper [ZKS+18])

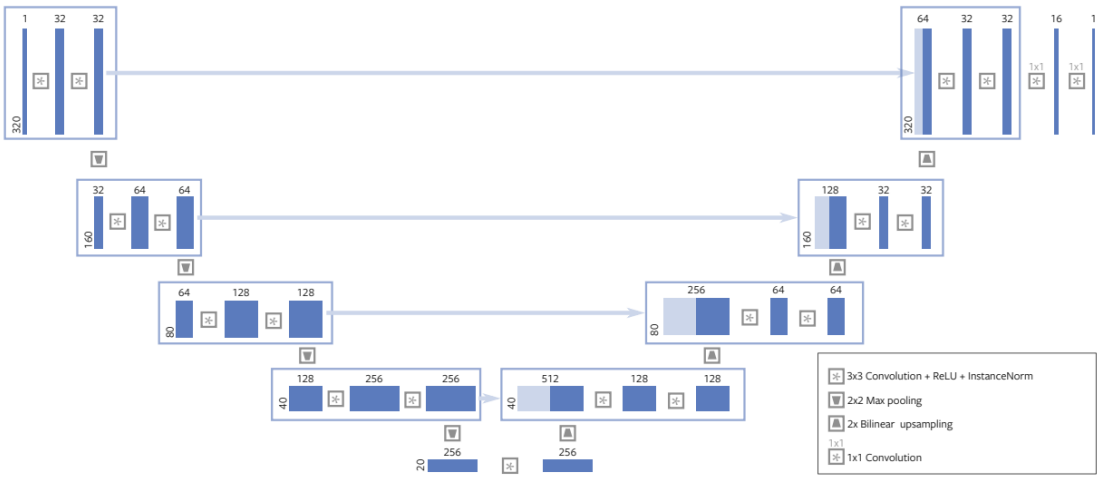


Figure 1.1: Baseline U-Net arcitecture. Taken from the paper [ZKS+18].

Chapter 2

Methods

2.1 Multi-channel data standardization

The U-Net baseline model, as described in Section 1.2, does not take coil sensitivity information into account, and therefore neither includes a direct parallel image reconstruction nor accounts for sparsity or other correlations among coils. Here, we describe a novel data standardization technique for multi-coil MRI data based on coil sensitivity maps.

2.1.1 Coil sensitivity maps estimation

First, we estimate the coil sensitivity maps corresponding to the undersampled coil images using the ESPIRiT approach, which was first proposed in the paper [ULM+14]. We used the implementation provided by the Berkeley Advanced Reconstruction Toolbox (BART) described in the paper [UVO+13], which contains implementations of standard methods for coil sensitivity estimation and undersampled MR image reconstruction incorporating parallel imaging and compressed sensing. The ESPIRiT approach harmonizes parallel imaging and compressed sensing in a unified framework. Figure 2.1 shows the absolute value of the inverse *discrete Fourier transform* (DFT) of fully-sampled complex-valued k-space signals for 4 coils of a 15-element knee coil array, and Figure 2.2 shows coil sensitivity map magnitudes given by ESPIRiT, corresponding to the same 4 fully-sampled k-space signals. Note that each coil is typically highly sensitive in one region, and its sensitivity falls off significantly in other regions.

2.1.2 Sensitivity maps based combination

Instead of combining the coil images using the root-sum-of-squares algorithm, we take coil sensitivity information into account and combine the different coil images

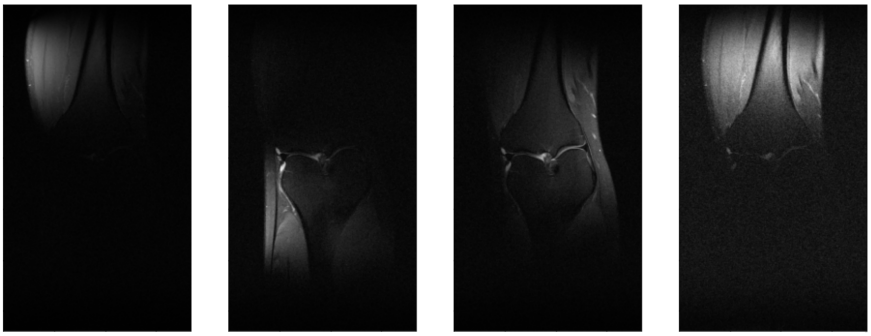


Figure 2.1: Individual coil spatial images from fully sampled data. This figure shows that each coil is highly sensitive in one region and its sensitivity falls off significantly in other regions.

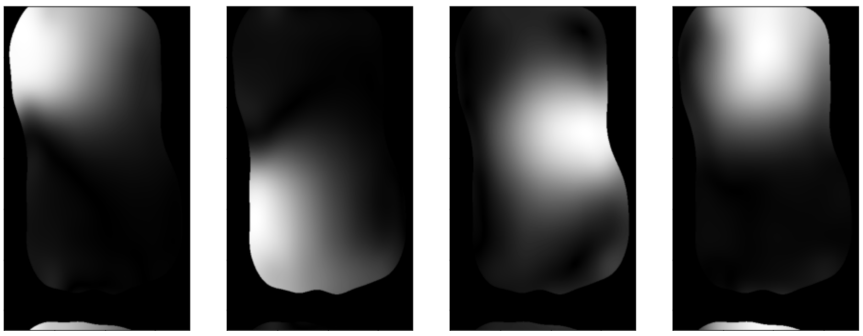


Figure 2.2: Coil sensitivity map magnitudes given by ESPIRiT. This figure shows that the coil sensitivity map has non-zero values in the region where the coil is sensitive.

as follows

$$\tilde{m}_{comb} = \sum_i \bar{S}_i \tilde{m}_i, \tag{2.1}$$

where $\tilde{m}_i$ is the zero-filled image of coil i, $\bar{S}_i$ is the complex conjugate coil sensitivity map corresponding to coil i and estimated by ESPIRiT and the multiplication is entry-wise. Then, depending on the used model, either $\tilde{m}_{comb}$ or its absolute value $|\tilde{m}_{comb}|$ is then used as the input to the model. The intuition behind this particular data standardization method is that this sort of standardization could be particularly effective when we have variation in the number of coils (from 2 to 24 in the brain dataset) and with different coil geometries and sensitivity information. In order to also consider sensitivity estimation errors or artifacts, one could compute the residual images given by

$$\tilde{m}_{i,res} = \tilde{m}_i - S_i \tilde{m}_{comb}. \tag{2.2}$$

Figure 2.3 shows the absolute value of the combined image m_{comb} and three residual images $m_{i,res}$ corresponding to the same fully-sampled 15-element knee coil array used in the previous two figures. The figure shows that the residual images are very noisy. Also, in our experiments with the knee dataset which has a fixed number of coils (15) among all volumes, concatenating these residual images with the combined image along the channel axis and passing the resulting multi-channel image $[m_{comb}, m_{1,res}, ..., m_{15,res}]$ as input to the model slightly decreased the performance compared to when just passing the combined image as input. On the other hand, it is not clear how to deal with these residual images, when we have different numbers of coils in different volumes in the dataset, as is the case with the brain dataset. One possibility is to pass each of the combined and residual images independently through the network and then combine the different outputs. Another way to deal with this variation is to pad every multi-channel image with zero-valued channels until it has the maximum number of channels present in the dataset. Because of these considerations, the results reported in this work use only the combined image $\tilde{m}_{comb}$ (or its absolute value) as input to the model.

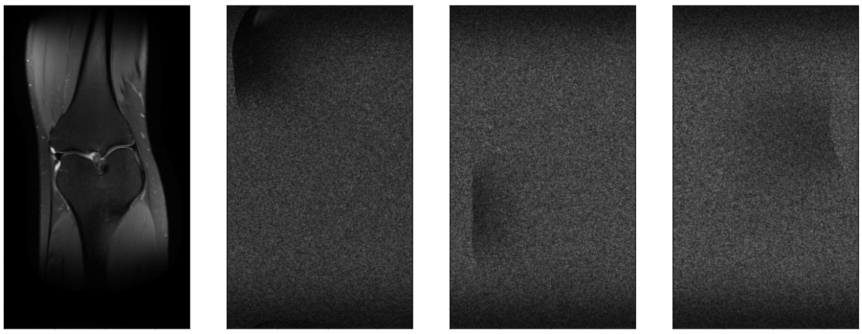


Figure 2.3: Absolute values of combined and residual images as given by equation (2.1) and equation (2.2), respectively. This figure shows that the combined image captures the information from all coils well and that the residual images are very noisy.

2.2 Multi-domain model

The U-Net baseline model, as described in Section 1.2, is only trained on the image domain, i.e., the model tries to estimate the true underlying images from images in which the detailed structures are distorted or have disappeared. To overcome this limitation, we implement a variation of the U-Net model, which operates on both the image and k-space domains. In the rest of this work, we refer to this model as MDU-Net for "Multi Domain U-Net". This model is similar to the model proposed by the AIRS medical team, but some implementation details might not be the same. A comparison of this model with the baseline model is presented later in Section 3.1

2.2.1 Model architecture

The overall structure of the MDU-Net is the same as of the U-Net baseline model. However, each convolutional layer of the MDU-Net operates on both the image and k-space domains, sharing features between the two domains. Along one path, the kernels of a given layer operate on the input image directly. Along the other path, the same kernels operate on the Fourier transformed input image, which represents the k-space. The output of the second path is then transformed back to the image domain using the Inverse Fourier Transform and then concatenated with the output of the convolution on the image domain along the channel axis. This constitutes the output of a convolutional layer of the MDU-Net, which is then passed to the next layer. A schematic representation of such a convolutional layer can be seen in Figure 2.4. An important observation to note here is that sharing parameters between both domains achieved slightly better performance than using independent parameters, in addition to keeping the number of parameters as low as possible.

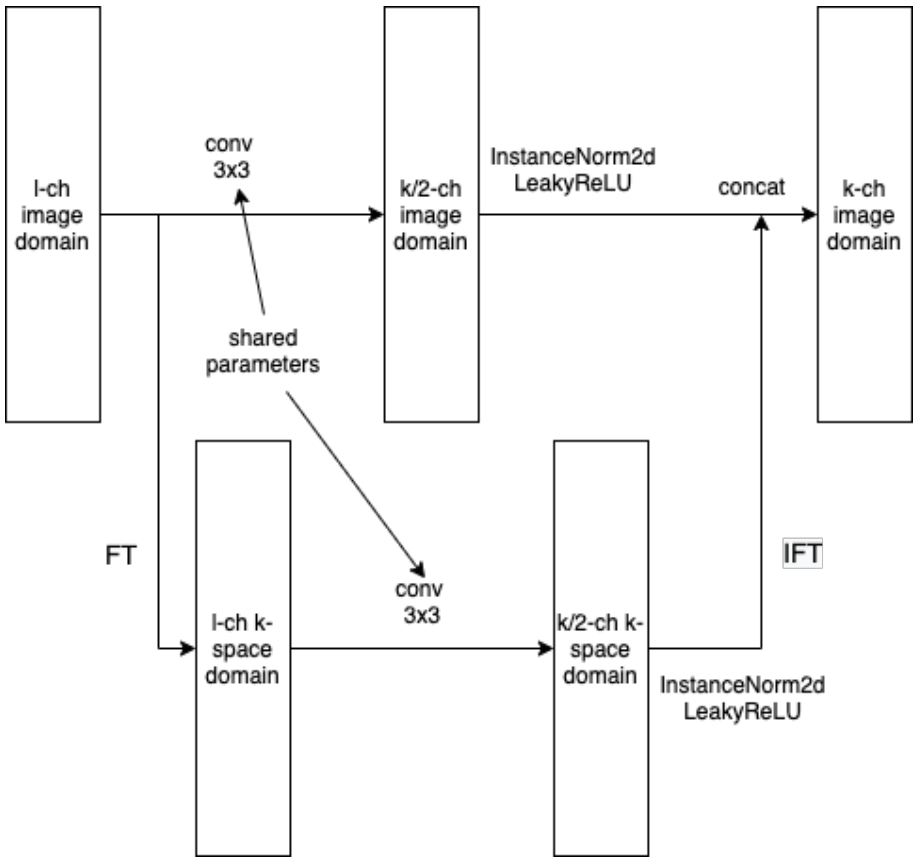


Figure 2.4: Schematic representation of a convolutional layer of MDU-Net

2.2.2 Implementation Details

In order to enable the integration of the Fourier Transform and the Inverse Fourier Transform into our model, we need to have model building blocks capable of handling complex-valued data. We use an implementation based on the framework proposed in the paper [TBZ+18], where the authors investigate additional benefits of using complex neural networks such as richer representational power due to having access to not only the amplitude, but also the phase information.

- **Complex Convolution**

  In order to perform the equivalent of a traditional real-valued 2D convolution in the complex domain, we convolve a complex filter matrix W = A + iB by a complex vector h = x + iy where A and B are real matrices and x and y are real vectors since we are simulating complex arithmetic using real-valued entities. As the convolution operator is distributive, convolving the vector h by the filter W we obtain:

  $$W \times h = (A \times x - B \times y) + i(B \times x + A \times y) \tag{2.3}$$

  A schematic representation of a complex convolution can be seen in Figure 2.5.

  Remarks:

  (i) For a standard convolutional layer which takes l feature maps as input, has k feature maps as output and uses $n \times m$ sized filters, the total number of parameters, including the biases, is $(n \times m \times l + 1) \times k$. A convolutional layer of the MDU-Net can be considered as two such convolutional layers (one corresponding to the real part of the convolutional kernel and the other one to its imaginary part), except that each layer outputs $\frac{k}{2}$ feature maps. So, the total number of parameters of a convolutional layer of the MDU-Net is $2 \times (n \times m \times l + 1) \times \frac{k}{2}$ which is the same for a usual convolutional layer having the same numbers of input and output feature maps.

  (ii) A more general way to support convolution operations on complex-valued data is to stack the real and imaginary parts of all input complex-valued channels along the channel axis, get a real-valued volume with the double number of channels compared to the original complex-valued volume, and then apply standard convolution on this transformed volume. However, if we implemented the convolutional layers of the MDU-Net in this way, one such layer would have $(n \times m \times 2l + 1) \times k$, which is approximately the double number of parameters of a similar layer with our current implementation.

- **Complex Instance Normalization and complex LeakyReLU**

  To perform Instance Normalization and LeakyReLU on complex-valued input data, we perform the real versions of these operations separately on the real and imaginary part.

2.3 Better input data initialization with GRAPPA

In this section, we describe how we use a classic image reconstruction method prior to passing the under-sampled images through the network. We use the *Generalized Autocalibrating Partially Parallel Acquisitions* (GRAPPA) algorithm.

2.3.1 GRAPPA

GRAPPA is an auto-calibrating coil-by-coil reconstruction method. It poses the parallel imaging reconstruction as an interpolation problem in k-space. In the GRAPPA algorithm, *unacquired* k-space values are synthesized by a linear combination of acquired neighbouring k-space data from all coils.

2.3.2 Implementation

We used the implementation provided by the package [McK]. The actual function implementing the algorithm takes in two arguments: the 2D multi-coil k-space data to reconstruct from, where the missing (undersampled) entries have exact zeros in them and calibration data which corresponds to fully-sampled k-space data. To get this calibration data, we use the central region of k-space corresponding to low frequencies, called the *Auto-Calibration Signal (ACS)*, which is typically fully sampled. To accurately reconstruct the missing values, the ACS must be sufficiently large, which limits the quality of the reconstructed images using this algorithm.
Figure 2.6 shows the absolute value of the inverse DFT of under-sampled complex-valued k-space signals from a brain multi-coil array without and with applying the GRAPPA algorithm on the under-sampled k-space data, and Figure 2.7 shows the root-sum-of-squared combination of these multi-coil images.

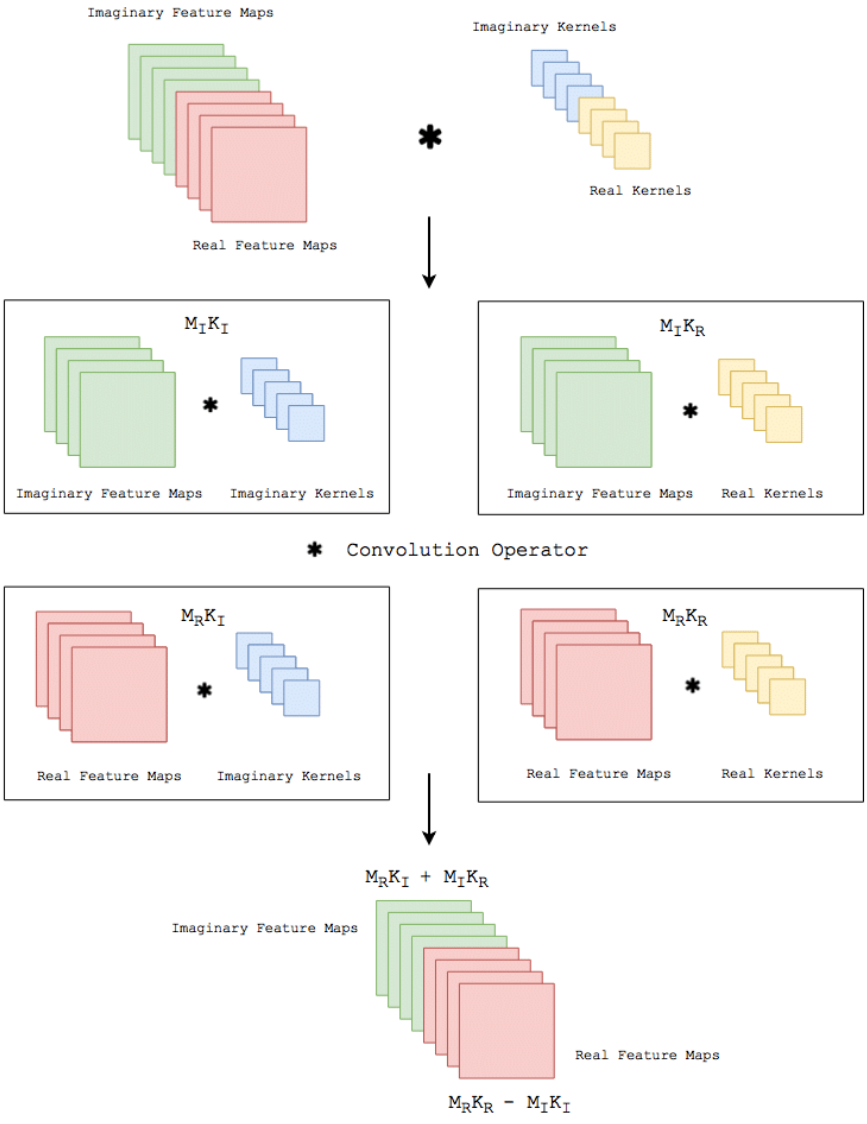
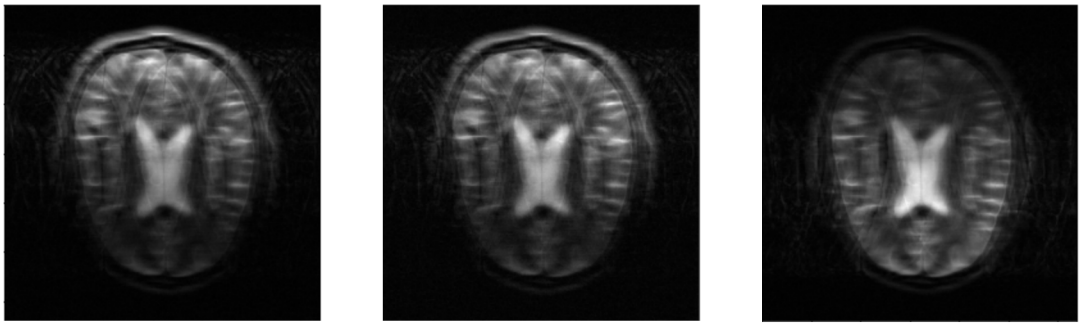
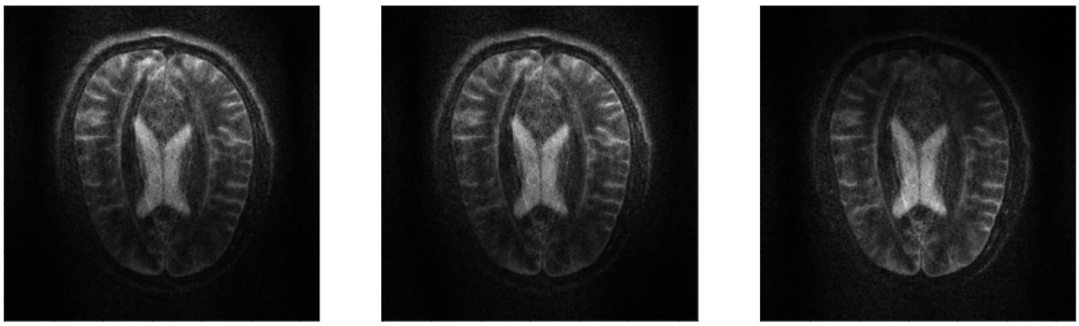


Figure 2.5: Schematic representation of the complex convolution operation using only real operations. This figure is taken from this repository [Che].

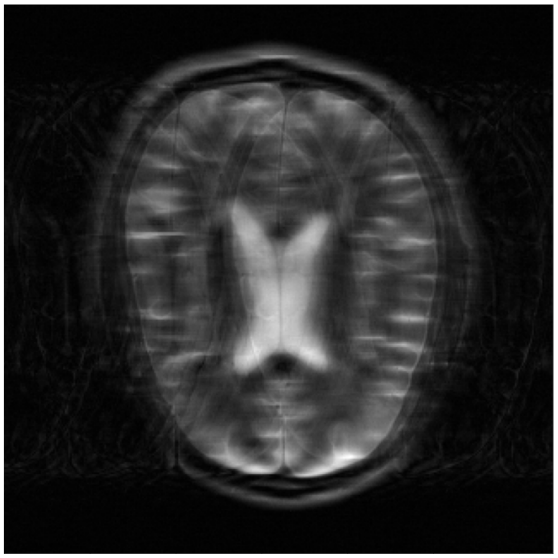


(a) Absolute value of coil images corresponding to under-sampled k-space data.

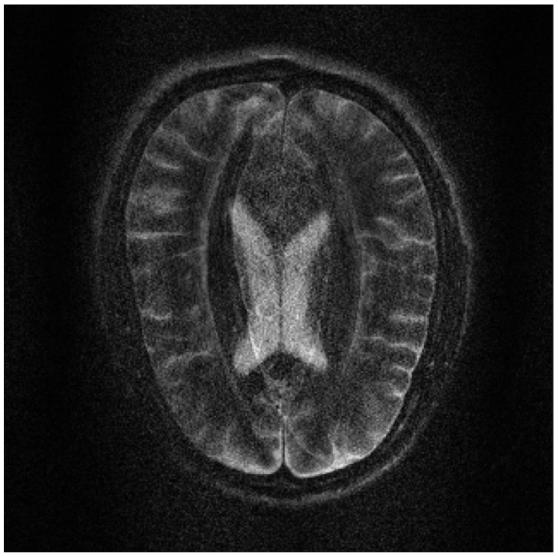


(b) Absolute value of coil images corresponding to GRAPPA-preprocessed under-sampled k-space data.

Figure 2.6: Multi-coil images corresponding to under-sampled k-space data without and with applying GRAPPA algorithm. This figure shows that the preprocessed images are less blurry than the under-sampled ones.



(a) Absolute value of an rss-combined image corresponding to masked k-space data.



(b) Absolute value of an rss-combined image corresponding to GRAPPA preprocessed k-space data.

Figure 2.7: Rss-combined multi-coil images corresponding to under-sampled k-space data without and with applying GRAPPA algorithm. The preprocessed image looks much less blurry than the non-preprocessed one.

Chapter 3

Evaluation

3.1 Experimental Results without applying GRAPPA

We performed our first experiments on approximately 50% of the knee images without fat suppression (PD) and on approximately 25% of the brain images with sequence AXT2. All models are trained using the RMSProp algorithm with a learning rate of 0.001 for 10 epochs. We train the following models on both datasets:

- Baseline model: The U-Net model with the root-sum-of-squares combined multi-coil image $\tilde{m}_{rss}$ as input, as given by equation (1.4).
- Baseline model + data standardization: The U-Net model with the absolute value of the combined image $|\tilde{m}_{comb}|$ given by equation (2.1) as input.
- MDU-Net: The MDU-Net with the combined image $\tilde{m}_{comb}$ computed with equation (2.1) as input.

The results of these experiments are reported in Table 3.1. For reference, we also add the results of the baseline model training reported in the paper [ZKS+18], where it was trained on 100% of the dataset for 50 epochs and the learning rate was multiplied by 0.1 after the first 40 epochs.
Note that the results reported in this section are not comparable to those reported in the following section, because different subsets are used for training and validation. These results are discussed in Chapter 4.

Model		NMSE	SSIM	PSNR
Baseline model (from the paper [ZKS+18])	Knee	0.0066	0.9192	36.7
	Brain	0.0255	0.9112	32.51
Baseline model	Knee	0.0076	0.9063	35.34
	Brain	0.0155	0.9000	32.29
Baseline model + data standardization	Knee	0.0092	0.8902	34.44
	Brain	0.0218	0.8615	31.02
MDU-Net	Knee	0.0188	0.8358	30.74
	Brain	0.0283	0.8274	28.16

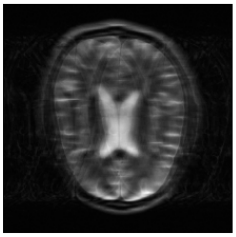
Table 3.1: Metrics on validation data of the three models.

3.2 Experimental Results on GRAPPA preprocessed data

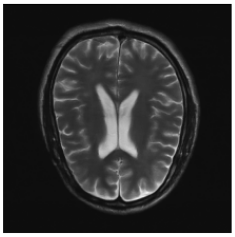
Since one run of the GRAPPA algorithm with the implementation we use is computationally very expensive, we slightly change the training pipeline; instead of sampling new random masks and computing new undersampled k-space data on each step, we take a subset of the training and validation brain dataset, sample random masks **once**, apply them on the k-space data, apply the GRAPPA algorithm on each of the undersampled k-space slices as discussed in Section 2.3, and save the resulting GRAPPA-preprocessed k-space data as well as the undersampled (not preprocessed) k-space data, in order to use them in every epoch of the training. We used 267 training volumes and 73 validation volumes from the brain dataset with sequence AXT2, which contain **4228** and **1162** image slices, respectively. We used the Adam algorithm for training. We used an initial learning rate of 0.001, which was multiplied by 0.1 after 40 epochs, after which the models were trained for an additional 10 epochs. Table 3.2. shows the performance of the different models on the validation set after the end of training. Figure 3.1 shows the inputs to the trained models, their respective outputs and the ground truth image corresponding to one slice from the validation data. Figure 3.2 shows the validation curves of the U-Net model with and without the GRAPPA preprocessing step. These results are also discussed in Chapter 4.

Model	L1-LOSS	NMSE	SSIM	PSNR
Baseline model(U-NET)	0.1186	0.0107	0.9205	33.8951
Baseline model + GRAPPA	**0.1052**	**0.0101**	**0.9222**	**34.9951**
Baseline model + GRAPPA + Data standardization	0.1326	0.0128	0.9069	34.1884

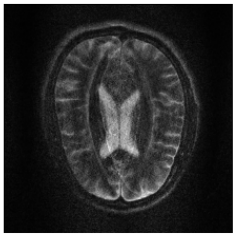
Table 3.2: Metrics on validation data



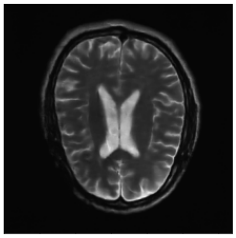
(a) Input to baseline model



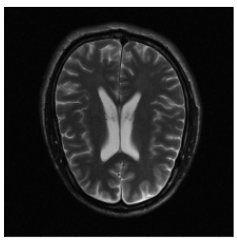
(b) Output of baseline model



(c) GRAPPA-preprocessed input

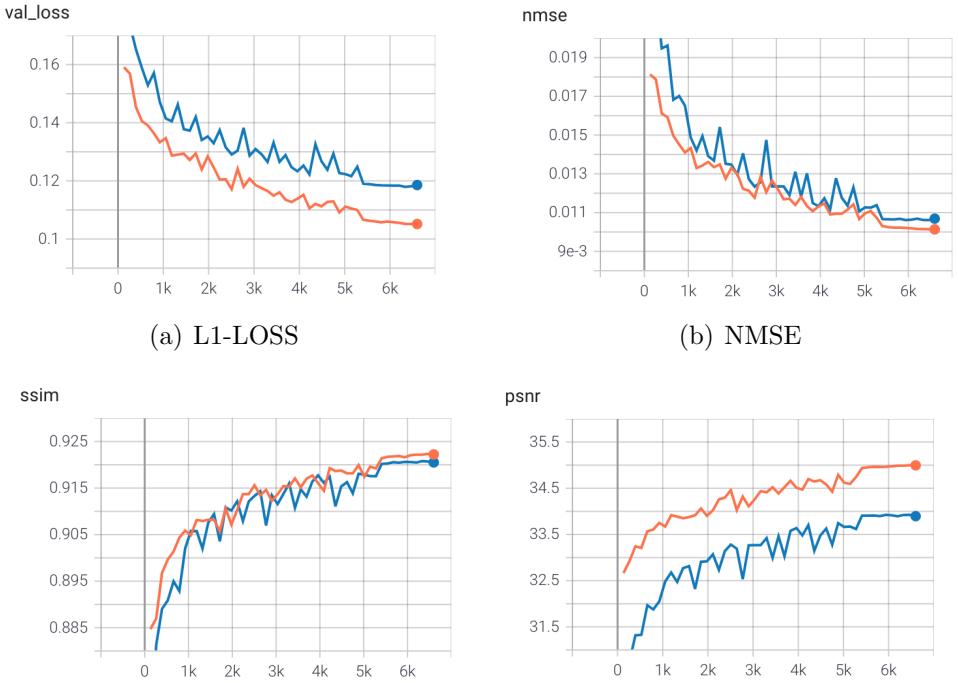


(d) output of the model trained on GRAPPA-preprocessed data



(e) ground truth

Figure 3.1: Inputs and outputs of the trained models and ground truth.



(a) L1-LOSS (b) NMSE

(c) SSIM (d) PSNR

Figure 3.2: Metrics on validation data. The blue curves correspond to the baseline model and the orange curves to the model trained on GRAPPA-preprocessed data. The models are trained for 50 epochs using Adam, with a stepsize decay for the last 10 epochs. These curves show that GRAPPA helps improve the model performance across all metrics and during the whole training period. Besides, the values obtained with GRAPPA are slightly less chattering than the other ones.

Chapter 4

Discussion

The results reported in Chapter 3 show that the implemented methods influence the performance on the MRI reconstruction task in different ways.
On one hand, the data standardization steps introduced in Section 2.1.1 slightly decreased the performance, compared to combining the multi-coil images with the widely used root-sum-of-squares method. One possible reason for this observation could be that we did not find a way to effectively incorporate the residual images, defined by equation (2.2), into our models. Another reason could be that this technique is indeed not as effective as the root-sum-of-squares method, but combining it with many other techniques led to the successful submission of the AIRS medical team.
On the other hand, the multi-domain model performed relatively a lot worse than the baseline U-Net model. This is actually surprising, because we would expect that giving the model access to both the k-space and image domain should improve the model's expressiveness and thus its performance. The most likely explanation for this behaviour is that we are not aware of all the implementation details used by the AIRS medical team. Since we are not able to test every possible implementation, we had to make many design choices which could differ from those used in the original implementation.
Finally, using the GRAPPA algorithm on the undersampled k-space corresponding to the input images as a preprocessing step significantly improved the overall performance of the model. The GRAPPA algorithm fills the missing values in the undersampled k-space based on some estimations, while keeping the acquired data unchanged. Intuitively, this process can only improve the quality of the reconstruction with the neural network model. This intuition was validated empirically by the results of our experiments.

Chapter 5

Conclusion

In this work, we studied the effects of the main methods used by the AIRS medical team in their submission in the 2020 fastMRI challenge. While we were able to validate the effectiveness of the data preprocessing step with GRAPPA, our results regarding the data standardization technique and multi-domain learning did not match the results of the team's submission. Future work could address this issue, by trying to find effective ways to incorporate the residual images into the training process and/or improving the multi-domain model to make it more effective and more stable.

List of Figures

List of Tables

Bibliography

[Che] James Youngchae Chee. https://github.com/litcoderr/ComplexCNN.

[CRT06] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on information theory*, 52(2):489–509, 2006.

[LDP07] Michael Lustig, David Donoho, and John M Pauly. Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

[McK] Nicholas McKibben. pygrappa. https://pygrappa.readthedocs.io/en/latest/#.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[TBZ+18] Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, JoÃ£o Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. Deep complex networks, 2018. arXiv:1705.09792.

[ULM+14] Martin Uecker, Peng Lai, Mark J Murphy, Patrick Virtue, Michael Elad, John M Pauly, Shreyas S Vasanawala, and Michael Lustig. Espirit an eigenvalue approach to autocalibrating parallel mri: where sense meets grappa. *Magnetic resonance in medicine*, 71(3):990–1001, 2014.

[UVO+13] Martin Uecker, Patrick Virtue, Frank Ong, Mark J Murphy, Marcus T Alley, Shreyas S Vasanawala, and Michael Lustig. Software toolbox and programming library for compressed sensing and parallel imaging. In *ISMRM Workshop on Data Sampling and Image Reconstruction*, page 41. Citeseer, 2013.

[ZKS+18] Jure Zbontar, Florian Knoll, Anuroop Sriram, Tullie Murrell, Zhengnan Huang, Matthew J Muckley, Aaron Defazio, Ruben Stern, Patricia Johnson, Mary Bruno, et al. fastmri: An open dataset and benchmarks for accelerated mri. *arXiv preprint arXiv:1811.08839*, 2018.