Magnetic Resonance Contrast Prediction Using Deep Learning

Cagan Alkan
Department of Electrical Engineering
Stanford University
calkan@stanford.edu

John Cocjin
Department of Bioengineering
Stanford University
john.cocjin@stanford.edu

Andrew Weitz
Department of Bioengineering
Stanford University
aweitz@stanford.edu

Abstract

A key strength of magnetic resonance imaging (MRI) is its ability to measure different tissue contrasts. However, each contrast is typically collected in series, making clinical MRI a slow and expensive procedure. In this paper, we present a novel approach for learning transformations from one MRI contrast to another. We train a fully convolutional neural network that takes T1-weighted MRI images as input, and generates the corresponding T2-weighted images as output. To evaluate our results, we compare different network depths, input features, and numbers of training subjects. We find that a nine-layer 2D convolutional neural network (CNN) using T1-intensity with eight tissue masks as input generates the best output images with lowest L2 error. Furthermore, the results can continue to improve with even larger training datasets. These results confirm the feasibility of using deep learning to predict one MRI contrast from another and accelerate clinical MRI acquisition.

1. Introduction

Magnetic resonance images can represent many different tissue contrasts depending on the specific acquisition paradigm that is used. For example, two of the most common MRI contrasts are T1-relaxation and T2-relaxation. T1-weighted images are acquired using short echo and repetition times, while T2-weighted images are acquired using long echo and repetition times. Each contrast conveys specific information about the local tissue and its physical properties. Because T1- and T2-relaxation values differ across various tissue types, radiologists and scientists must use the information conveyed in both contrasts to make informed decisions. For example, in T2-weighted images, areas of the brain filled with water appear bright and tissues with high fat content appear dark. This can aid in localizing pathology since many brain lesions are associated with an increase in water content. Similarly, in T1-weighted images, tissues with high fat content appear bright and compartments filled with water appear dark. T1-weighted images are therefore useful for looking at detailed anatomical images. Examples of these two contrasts are provided in Figure 1.

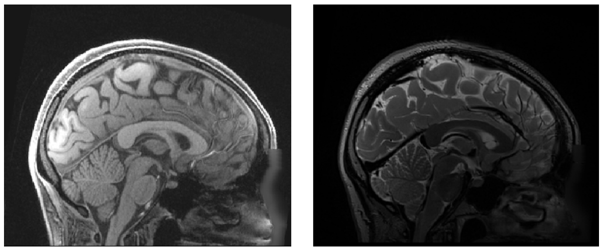
Figure 1. Example pair of T1-weighted (left) and corresponding T2-weighted (right) images. The proposed CNN predicts T2-weighted images from T1-weighted images.

It is important to note that despite the tissue-specific changes in brightness associated with T1- and T2-weighted images described above, these two contrasts are not simply inverted versions of one another. Indeed, they each correspond to distinct tissue properties, and a low T1 value will not necessarily correspond to a high T2 value (or vice versa).

Because the information provided by both T1- and T2-weighted images is important, clinical MRI protocols will often collect both. However, MRI acquisition is a slow and costly procedure. Therefore, a significant aim of modern MRI research is to accelerate the image acquisition process. Current approaches focus on both software (e.g. compressed sensing) and hardware (e.g. stronger gradient magnets). Here, we propose the use of convolutional neural networks and deep learning to predict one contrast from another. If successful, this approach would eliminate the need for one acquisition altogether and dramatically cut down the time and costs of clinical MRI.

We formulate the problem of T1-to-T2 mapping as a standard regression problem. The input to our algorithm is a two-dimensional T1-weighted image concatenated with eight additional channels that serve as binary masks to indicate different tissue types. We then use a CNN to output a predicted T2-weighted image. We choose to learn the mapping of T1-to-T2 transformations rather than vice versa, since the eight tissue mask channels can be easily computed from T1-weighted images.

2. Related Work

In the last year, deep learning and CNNs have rapidly taken over the field of medical imaging and MRI. Researchers in the field have focused on two main problems: noiseless reconstruction of undersampled data [15, 19, 17, 7, 12], and automated segmentation of different tissue types [18, 14, 8, 2]. Regarding the former, various deep learning methods have been developed to reconstruct MRI images that are undersampled in the frequency (i.e. acquisition) domain. For example, Schlemper et al. [15] used a cascade of CNNs with data consistency layers to outperform state-of-the-art compressed sensing methods for reconstructing six-fold undersampled cardiac MRI images. In another case [19], researchers developed a generalized reconstruction algorithm composed of three fully-connected layers followed by a three-layer convolutional autoencoder. The researchers show that this approach can be used to noiselessly reconstruct undersampled and/or corrupted MRI images, independent of the sampling trajectory.

Deep learning and CNNs have also been used for automated segmentation and detection of various pathologies or tissue types in MRI. For example, CNNs were used to segment brain tissue into white matter, gray matter, and cerebrospinal fluid in infant MRIs [18]. The authors used three modalities of imaging as input - T1, T2, and fractional anisotropy - and patched each image into 13x13 subimages. Their architecture included three convolutional layers and one fully connected layer with a soft-max loss function on the three different tissue classes. Using this approach, they were able to achieve results that significantly outperform previous methods of infant brain tissue segmentation.

Despite the many applications of deep learning to medical imaging, there have been very few applications for performing image transformations - the problem we address in this paper. In one application, CNNs were used to predict computed tomography (CT) images from corresponding MRI images [13]. In another, CNNs were trained to predict high resolution MRI images acquired with a 7 Tesla scanner from low resolution images acquired with a 3 Tesla scanner [3]. Both approaches used relatively straightforward architectures – three repeated convolutional layers followed by a loss function, with no pooling or fully connected layers. Both approaches also used patches to divide the original images into smaller subimages. Finally, both used the L2 loss function to learn their respective transformations. The latter network [3] used anatomical tissue masks as input to improve performance. We believe each of these features worked well for their respective problems, and adapt them here. Specifically, we use fully convolutional networks, the L2 loss function, ”small” input patches, and eight tissue masks to serve as additional input channels. These methods are described in detail in the following sections.

3. Dataset and Features

We utilized the Human Connectome Project (HCP) data release of March 2017, which consists of both raw and processed 3T MRI imaging data from 1206 healthy young adults [16]. Each subject includes T1- and T2- weighted image sets collected within the same scanning session. In a minority of subjects, more than one scanning session was conducted. For these, the image sets were preprocessed, registered, and averaged within contrast. All image sets were screened to exclude major neuroanatomical anomalies. Minor, benign variation such as a cyst is allowed, however, and is reported in ConnectomeDB, the data management platform for accessing HCP data. We do not distinguish between subjects with or without such variations. Below, we briefly describe preprocessing of the data that is relevant to the interpretation of our experiments. Further details can be found in [6]. All files specified below are available via the ConnectomeDB database.

Separately, the raw T1- and T2- weighted image sets were corrected for nonlinearities induced by B0 field image distortions. The T1-weighted images underwent anterior commissure-posterior commissue (ACPC) alignment, and the T2-weighted images were then aligned to these using rigid body transformation. These steps produced the images stored in `T1w_acpc_dc.nii.gz` and `T2w_acpc_dc.nii.gz` that we use for T1-to-T2 contrast mapping. Images in both datasets are of size $260 \times 311 \times 260$ voxels with 0.7 mm isotropic resolution.

The T1 volume was skull stripped and segmented into different tissue types using HCP-specific Freesurfer scripts [5]. The segmentation is provided in the output file `wmparc.nii.gz` and corresponds to a combination of white matter/grey matter cortical parcellation and subcortical segmentation. The brain mask derived from the skull-stripping step is provided in the file `brainmask_fs.nii.gz`. We aggregate the brain mask and segmentation into eight label sets: brain, grey matter, white matter, cerebral spinal fluid, subcortical structure, vessel, cerebellum, and unknown. Binary masks were genereated for each anatomical tissue label, and treated as a separate input channel in addition to the T1-weighted intensity values. Binary values of -1 and +1 were chosen to maintain zero-mean input.

In this work, we operate over 2D images and treat each slice as an independent sample. This results in only 260 samples per subject. Therefore, to increase the size of our dataset, we split each slice into smaller subimages, or "patches." We use a patch size of (128 $\times$ 128) and a stride of 24, giving 12,480 samples per subject – effectively increasing the dataset by a factor of 50. Prior to patching, the average overall intensity value was calculated across all training images and subtracted from the complete dataset.

As described below, we ran experiments to explore the effect of the training set size on model output. We first developed the model using only a single subject. The 12,480 image patches were split into training, validation, and test sets, using an 80-1-19% breakdown, respectively. Thus, n=9,984 samples were used for training, and n=2,371 samples were used for testing. To determine how well our results generalized across subjects, we next trained the model using all samples from either one or fifty subjects, and used all images from five different subjects as the test set. In these cases, the total number of training samples was n=12,480 and n=624,000, respectively, while the total number of test samples was n=62,400.

4. Methods

4.1. Network Architecture

The network architecture implemented here relies on three main operations: two dimensional convolution layers, spatial batch normalization [9], and the rectified linear unit (ReLU) activation function. The 2D convolution layers compute the output using the relation shown in Equation 1

$$y[b,i,j,k] = \sum_{d_i,d_j,q} X[b, s_i \cdot i + d_i, s_j \cdot j + d_j, q] \cdot f[d_i, d_j, q, k] \tag{1}$$

where X and y are the input and output tensors stored in NHWC format, respectively, f is the 4D tensor that has the same 2D filter along N and C dimensions, and s_i and s_j are the strides along H and W dimensions, respectively.

Batch normalization is a recently developed technique that makes deep neural networks more robust to poor initialization [9]. Each batch normalization layer serves to normalize the previous layers activations into a distribution with learnable mean β and variance γ. Typically these values are initialized to 0 and 1, respectively, making the previous layer's output take on a zero-mean, unit-variance distribution. A momentum variable also allows these parameters to be updated as a running average. The batch normalization operation has the effect of implicit regularization, and allows higher learning rates to be used with less concern over initialization parameters [9]. In the model implemented here, batch normalization parameters were chosen to be the same as those initially described in [9]. There are: momentum = 0.99, ϵ = 0.001, β and its moving average initialized to 0, and γ and its moving average initialized to 1. The ϵ parameter is a small float value that is added to the variance to avoid division by zero, and has negligible impact on results.

The ReLU function takes the form $f(x) = \max(0, x)$. This activation function was favored over other options, such as the sigmoid and tanh functions, for its computational simplicity and accelerated convergence properties in stochastic gradient descent [11].

In the model used here, we pass input patches of size (128 $\times$ 128 $\times$ depth) through a nine-layer CNN. The input depth can either take on a value of 1 or 9, depending on whether tissue labels are used as additional input features. The first eight convolutional layers conform to the pattern [Conv2D]-[BatchNormalization]-[ReLU]. Each Conv2D operation includes 32 (3$\times$3) filters. To collapse the hidden layer data into the desired output dimensions (i.e. to force a depth of one), the final convolutional layer has only one (3$\times$3) filter. We note that no explicit regularization techniques were implemented in our model, since overfitting did not manifest in preliminary tests. A summary of the architecture is provided in Figure 2.

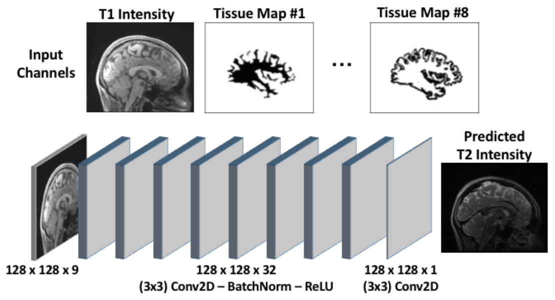

Figure 2. Proposed model architecture for T1-to-T2 mapping. Note that input depth can be either 1 or 9, depending on whether tissue labels are used as additional features.

4.2. Training

Training is implemented using the mean-square-error loss function shown in Equation 2

$$d_2(I_1, I_2) = \frac{1}{N}\sum_{n=1}^{N}(I_1[n] - I_2[n])^2 \tag{2}$$

where I_1 and I_2 are flattened images and N is the number of pixels. The Adam algorithm was used as the optimizer to minimize this loss [10]. This is a method for performing stochastic gradient descent over a high-dimensional, non-convex landscape. It computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients. The full Adam algorithm is described in Equations 3a through 3c below, where m is a moving average of the gradient, v is a moving average of the gradient squared, β_1 is the momentum parameter of the gradients, β_2 is the momentum parameter for the gradients squared, λ is the learning rate, and ϵ is a small float added to avoid division by zero. A higher momentum value means that the current gradient will contribute less to the moving average. We implemented the Adam algorithm with hyper-parameters suggested in its original formulation [10]. These are: β_1 = 0.9, β_2 = 0.999, and ϵ = 1e-09.

$$m_{t+1} = \beta_1 \cdot m_t + (1 - \beta_1) \cdot dx \tag{3a}$$

$$v_{t+1} = \beta_2 \cdot v_t + (1 - \beta_2) \cdot dx^2 \tag{3b}$$

$$x_{t+1} = x_t - \lambda \cdot m_{t+1}/(\sqrt{v_{t+1}} + \epsilon) \tag{3c}$$

In our implementation, we anneal the learning rate λ in equation 3c following a step decay schedule. We reduce λ by a half every 4 epochs, starting with an initial learning rate of 0.1. Decaying the learning rate helps improve training performance by allowing the optimizer to settle down into the narrower parts of the loss function.

As described in the experiments section below, we train our models using either data from a single subject or multiple subjects. The models that use a single subject for training are trained for 20 epochs. Per epoch training time for the models that use 50 subjects is considerably higher. Therefore, these models are trained only for 8 epochs.

We use a batch size of 32 for the models trained on a single subject. In order to take advantage of the parallelism between CPU based patch extraction and GPU based model training, we select a batch size of 128 for training the models using training data from multiple subjects.

4.3. Implementation

To implement our model, we use the Keras [4] API with a TensorFlow [1] backend. All computations are executed on Google Cloud using a CPU or NVIDIA Tesla K80 GPU. Due to the large size of the dataset, it is not possible to load the patches for all subjects into computer memory at once. In order to remedy this, Python generators are used to extract and generate patches for each batch separately. In addition, the NumPy memmap module is utilized to partially load data from disk on the fly and reduce memory usage.

5. Experiments, Results, and Discussion

5.1. Experiments

Multiple experiments were run to evaluate the effect of network depth, training set size, and the addition of input features providing voxelwise tissue-type labeling. We evaluate model performance both by visual inspection and by using the mean-square-error loss values between the output T2-weighted images and the ground truth T2-weighted images. To facilitate comparison between models, we show the same images from the test set for the across-subject experiment results.

5.2. Results and Discussion

5.2.1 Model Learning

To confirm that the proposed model could learn, we first trained and tested the model using data from one subject. This model included nine convolutional layers and nine input channels (i.e. T1-intensity plus eight tissue masks). The loss curves shown in Figure 3 exhibit a steady decline followed by a plateau, indicating that the network could learn and that network parameters are updating in a proper and stable manner. The small gap between training and validation loss also suggest that the models are properly regularized. Having confirmed that the model successfully learned, we moved on to test the role that input features, network depth, and training set size have in the accuracy of our results.

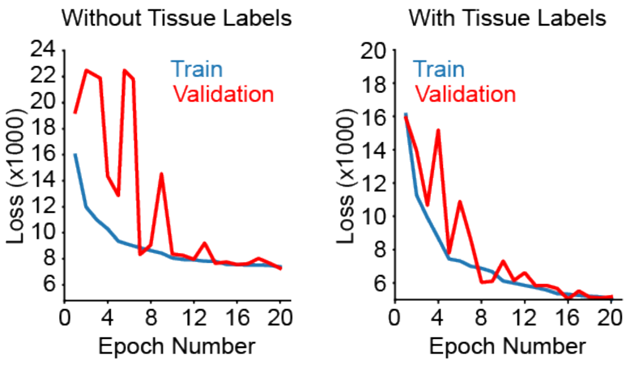

Figure 3. Learning curves for within-subject testing and training.

5.2.2 Effect of input features

We first investigated the effect of using only the T1-weighted image intensity as input versus using the T1-weighted image combined with eight additional tissue mask channels. Visually, the resulting T2-weighted images looked much more similar to the ground-truth images when the tissue labels were included. This was true in the case of both deep (Figure 4c vs. 4d) and shallow (Figure 4e vs. 4f) networks when using one subject for training and using five different subjects for testing. It was also true in the case of a deep network when using 50 subjects for training (Figure 5c vs. 5d). Furthermore, as shown in Tables 1-3, providing

	No Tissue Masks	Tissue Masks
3 Layer Network	12,795	10,664
9 Layer Network	5,177	4,091

Table 1. Within-subject loss values. Corresponding output images are not shown for simplicity.

these tissue labels as input reduced the L2 error across all model variants tested - typically by 20%. We thus conclude that the addition of binary tissue label masks significantly enhances the prediction of T2-weighted images from T1-weighted images.

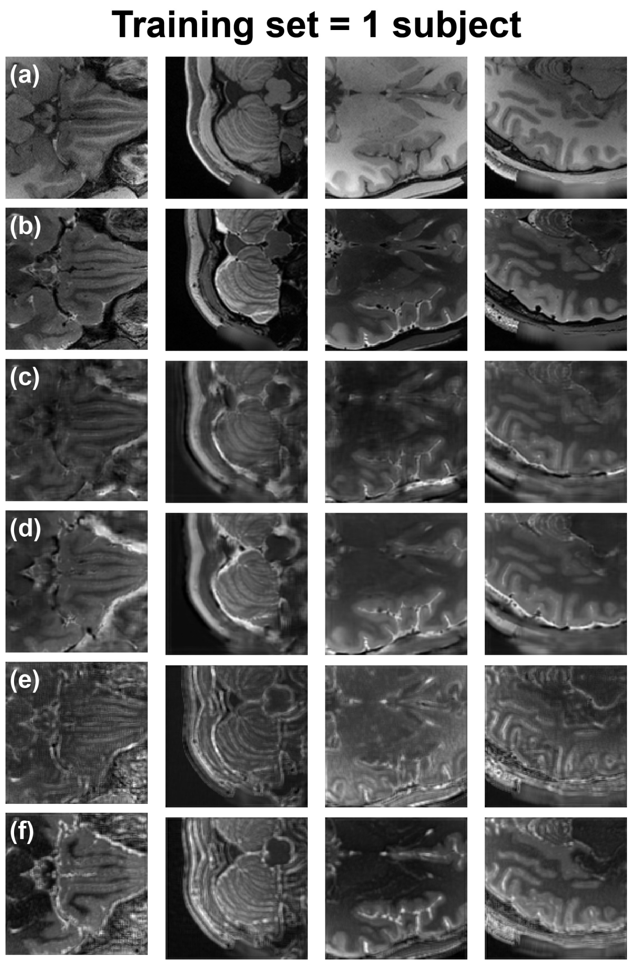

Figure 4. Across-subject T2 prediction results using n=1 subject training set for four representative patches. (a) T1-weighted image. (b) Ground-truth T2-weighted image. (c) Predicted T2 image using nine-layer network without tissue masks. (d) Predicted T2 image using nine-layer network with tissue masks. (e) Predicted T2 image using three-layer network without tissue masks. (f) Predicted T2 image using three-layer network with tissue masks.

	No Tissue Masks	Tissue Masks
3 Layer Network	20,385	16,627
9 Layer Network	12,643	9,783

Table 2. Across-subject loss values, training size = 1 subject.

	No Tissue Masks	Tissue Masks
9 Layer Network	9,093	7,728

Table 3. Across-subject loss values, training size = 50 subjects.

5.2.3 Effect of network size

We next compared our results with those of a shallower network that uses only three convolutional layers instead of nine. Visually, we found that the resulting T2-weighted images looked much more similar to the ground-truth images when a deeper network was used (Figure 4c,d vs. 4e,f). In this case, we see that adding additional convolutional layers significantly enhances the prediction performance. As shown in Tables 1-2, increasing the number of convolutional layers reduced the L2 error across all model variants tested - typically by 40-60%. We thus conclude that the addition of more convolutional layers significantly enhances the prediction of T2-weighted images from T1-weighted images.

5.2.4 Effect of training set size

Finally, we compared two scenarios to determine the improvement in performance achieved by training over larger datasets: (1) a single subject was used as the training set, while five other subjects were used as the test set, and (2) images from 50 subjects were used as the training set, while five other subjects were used as the test set. Comparing the loss values in Tables 2 and 3 indicates that the loss decreased when using the larger training set size. Furthermore, a comparison of the images shown in figures 4d and 5d shows that certain features were better preserved in the inclusion of multiple subjects, such as the apparent separation of adjacent gyri. In addition, subcortical blurring was significantly reduced on the images obtained using the model trained with a larger training set.

5.2.5 Tissue specificity

As can be seen, the trained networks could preserve tissue traits in the mapping of T1-weighted to T2-weighted images. As a remarkable example, the CNN recognized inversion. For example, while the predicted T2 images show

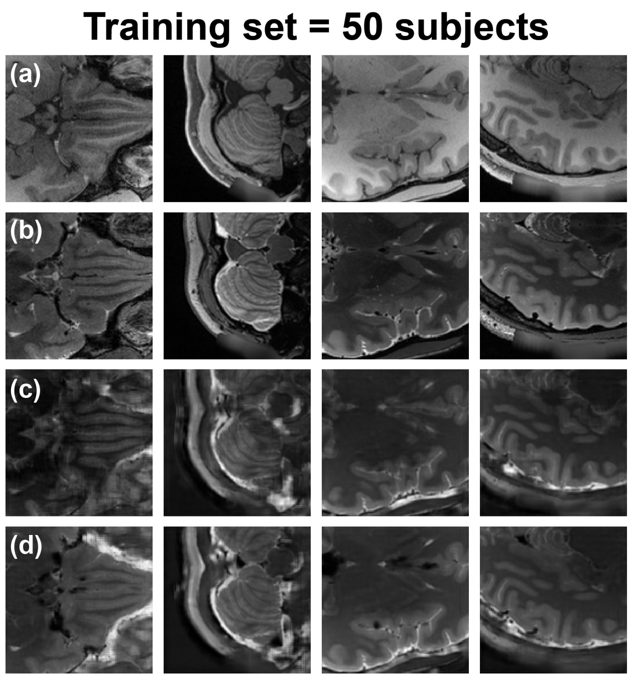

Figure 5. Across-subject T2 prediction results using n=50 subject training set for four representative patches. (a) T1-weighted image. (b) Ground-truth T2-weighted image. (c) Predicted T2 image using nine-layer network without tissue masks. (d) Predicted T2 image using nine-layer network with tissue masks.

dark white matter and bright gray matter (the opposite from the input T1 images), the skull and ventricle intensities generally do not change. All networks are able to produce these inversions. That the network without tissue labels could learn these mappings suggests that tissue specific mappings are latent in intensive patterning of the patches, that the patches provided enough view to sufficiently distinguish large scale structure, or both.

6. Conclusions and Future Work

Our experiments demonstrate the feasibility of applying deep convolutional neural networks towards the problem of predicting one MRI contrast from another. Our results indicate that deep CNNs perform better than shallow CNNs, and that including tissue type feature labels as additional inputs improves performance. Finally, we find that larger training datasets have potential in improving the performance of the network but require significant training time, even on a GPU.

A critical question moving forward with this approach is whether the algorithm can accurately predict T2-weighted images in the presence of abnormalities. Because MRI images are acquired for diagnostic purposes, this ability is necessary if it were to ever be deployed. Although the dataset used here included healthy subjects only, future work will incorporate pathological images with tissue abnormalities. If the model cannot successfully predict these abnormalities, the training set can be revised to include this type of data.

There are a number of hyperparameter options left to test. One other item to test in future work is any potential improvement by using 3D convolutions instead of 2D convolutions. 3D convolutions were initially attempted here, but found to be infeasible due to memory constraints. Running the model with higher specification computers and employing the NumPy memory map module used here are likely to make this approach more feasible. Another possibility is to sweep across patch or 1st layer kernel sizes to vary the amount of spatial information that the network is provided at the input level.

If network optimization were to reach satisfactory mapping, as validated by radiologists or imaging scientists, it would be of interest to transfer the network to other contrast problems.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] M. Avendi, A. Kheradvar, and H. Jafarkhani. A combined deep-learning and deformable-model approach to fully automatic segmentation of the left ventricle in cardiac mri. *Medical image analysis*, 30:108–119, 2016.

[3] K. Bahrami, F. Shi, I. Rekik, and D. Shen. *Convolutional Neural Network for Reconstruction of 7T-like Images from 3T MRI Using Appearance and Anatomical Features*, pages 39–47. Springer International Publishing, Cham, 2016.

[4] F. Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

[5] B. Fischl. Freesurfer. *Neuroimage*, 62(2):774–781, 2012.

[6] M. F. Glasser, S. N. Sotiropoulos, J. A. Wilson, T. S. Coalson, B. Fischl, J. L. Andersson, J. Xu, S. Jbabdi, M. Webster, J. R. Polimeni, et al. The minimal preprocessing pipelines for the human connectome project. *Neuroimage*, 80:105–124, 2013.

[7] Y. S. Han, J. Yoo, and J. C. Ye. Deep learning with domain adaptation for accelerated projection reconstruction mr. *arXiv preprint arXiv:1703.01135*, 2017.

[8] M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P.-M. Jodoin, and H. Larochelle. Brain tumor segmentation with deep neural networks. *Medical image analysis*, 35:18–31, 2017.

[9] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[10] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In

Advances in neural information processing systems, pages 1097–1105, 2012.

[12] D. Lee, J. Yoo, and J. C. Ye. Deep artifact learning for compressed sensing and parallel mri. *arXiv preprint arXiv:1703.01120*, 2017.

[13] D. Nie, X. Cao, Y. Gao, L. Wang, and D. Shen. *Estimating CT Image from MRI Data Using 3D Fully Convolutional Networks*, pages 170–178. Springer International Publishing, Cham, 2016.

[14] A. Prasoon, K. Petersen, C. Igel, F. Lauze, E. Dam, and M. Nielsen. Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network. In *International conference on medical image computing and computer-assisted intervention*, pages 246–253. Springer, 2013.

[15] J. Schlemper, J. Caballero, J. V. Hajnal, A. Price, and D. Rueckert. A deep cascade of convolutional neural networks for mr image reconstruction. *arXiv preprint arXiv:1703.00555*, 2017.

[16] D. C. Van Essen, S. M. Smith, D. M. Barch, T. E. Behrens, E. Yacoub, K. Ugurbil, W.-M. H. Consortium, et al. The wu-minn human connectome project: an overview. *Neuroimage*, 80:62–79, 2013.

[17] S. Wang, Z. Su, L. Ying, X. Peng, S. Zhu, F. Liang, D. Feng, and D. Liang. Accelerating magnetic resonance imaging via deep learning. In *Biomedical Imaging (ISBI), 2016 IEEE 13th International Symposium on*, pages 514–517. IEEE, 2016.

[18] W. Zhang, R. Li, H. Deng, L. Wang, W. Lin, S. Ji, and D. Shen. Deep convolutional neural networks for multi-modality isointense infant brain image segmentation. *NeuroImage*, 108:214–224, 2015.

[19] B. Zhu, J. Z. Liu, B. R. Rosen, and M. S. Rosen. Image reconstruction by domain transform manifold learning. *arXiv preprint arXiv:1704.08841*, 2017.