Detection and removal of biases from brain MR images using adversarial architectures

Sahitya Mantravadi
Stanford University
Institute for Computational and Mathematical Engineering
smantra@stanford.edu

Chris Gorgolewski*
Stanford University
Center for Reproducible Neuroscience
chrisgor@stanford.edu

Abstract

To practically combine MRI datasets from different sites, we must be able to remove biases in each image that point to the site at which that MR image was taken. The biases in this case are any markers or difference between MR images from different sites.

To achieve this, we outline the architecture that will preserve information in each image and make accurate predictions while mitigating the presence of confounds. Using the ABIDE dataset, we aim to develop an MRI debiasing architecture that will minimize the accuracy of site prediction and maximize the accuracy of quality prediction.

We then detail the methods and architectures tested in the development of the debiasing network. We discuss pitfalls and issues encountered. Lastly, we discuss related work and the extensibility of this architecture for other use cases.

1. Introduction

Neuroscience researchers can now leverage advances in the accuracy and explainability of convolutional neural networks to draw conclusions from MRI and fMRI data. However, reproducible neuroscience is difficult to achieve without large datasets. Hand-collected and hand-labelled MRI data can be difficult to aggregate across medical sites, especially when scanners have disparate settings and MR images have different sizes and resolutions. If not properly dealt with, differences brought about by site and scanner variation may be falsely attributed to biological differences. In order to mitigate these site effects, the ultimate goal is to remove site effects from the data, effectively "de-biasing" each image.

We develop and apply adversarial methods to a set of brain MRI data. For each image, we have a quality score label given to the image by a rater. Intuitively, preserving this quality score and retaining the ability to predict this quality score while de-biasing data allows us to preserve the quality of and important information from each image while removing site bias. To do this, we evaluate several adversarial network architectures in the context of removal of such biases from MR images. On a high level, we create an adversarial network architecture that has three components as shown in Figure 1 (building on the structure of a conditional generative adversarial network [5] [3]):

1. a discriminator to predict the site of an input image
2. a discriminator to predict the quality of an input image
3. a de-biasing component (here, the generator) that takes the input image and outputs a modified image. This component minimizes the accuracy of (1) and maximizes the accuracy of (2)

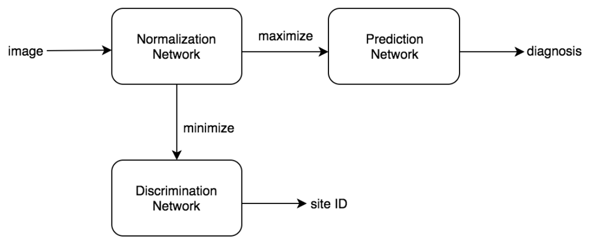

Figure 1. The three components of the adversarial architecture.

We will describe the architecture of each component as well as specific design choices that were made.

2. Related Work

Our work builds off the work of Goodfellow et. al [3] in generative adversarial frameworks. This a growing area of research and allows us to solve competing objective problems in the hopes of reaching a stable solution.

In addition, we build on the work of Mirza et. al [5] in conditional generative adversarial networks. They provide a framework to feed in the data they wish to condition on to both the generator and discriminator. We utilize part of this framework by conditioning both discriminators and the generator on the original brain MR images.

Instead of creating images to fool a discriminator, we wish to modify an existing image to meet a certain objective.

Zhang et. al [7] present a framework to mitigate the use of biases in prediction. They describe a two-player model in which the objective is to maximize the predictor's ability to predict some classification while minimizing the adversary's ability to predict the bias. Our method may provide more insight into the bias detection and removal process because we can examine the output of a generator. Our method also has potential for transfer learning, as once the generator is trained, it can be used independently to de-bias data.

3. Data

We apply methods to the Autism Brain Imaging Data Exchange (ABIDE I) dataset, which aggregates smaller sets of data from 17 different sites, as described in detail and utilized in [2].

Each MR image is represented as a three-dimensional matrix, which can be visualized with the medical research image viewing software Mango. A typical brain MR image is shown in Figure 2. We can see that the three dimensional MR image captures three views of the brain: axial, coronal, and sagittal.

Because this dataset consists of 17 sets of images from different sites, images from a particular site can have different sizes and resolutions than images from other sites. The MRI have all been standardized to be of size $106 \times 128 \times 110$. The output images from the debiasing component of the architecture are of this standard size as well. No data augmentation or other preprocessing steps were taken.

In total, the dataset contains 1103 MRI. This was randomly split into 803 training examples, 200 validation examples, and 100 test examples. Due to RAM constraints, we were limited to a batch size of 4 to train. This means each epoch over the training data is about 200 training steps.

3.1. Labels

Because the architecture encapsulates two discriminators, each data point is associated with two labels: a quality

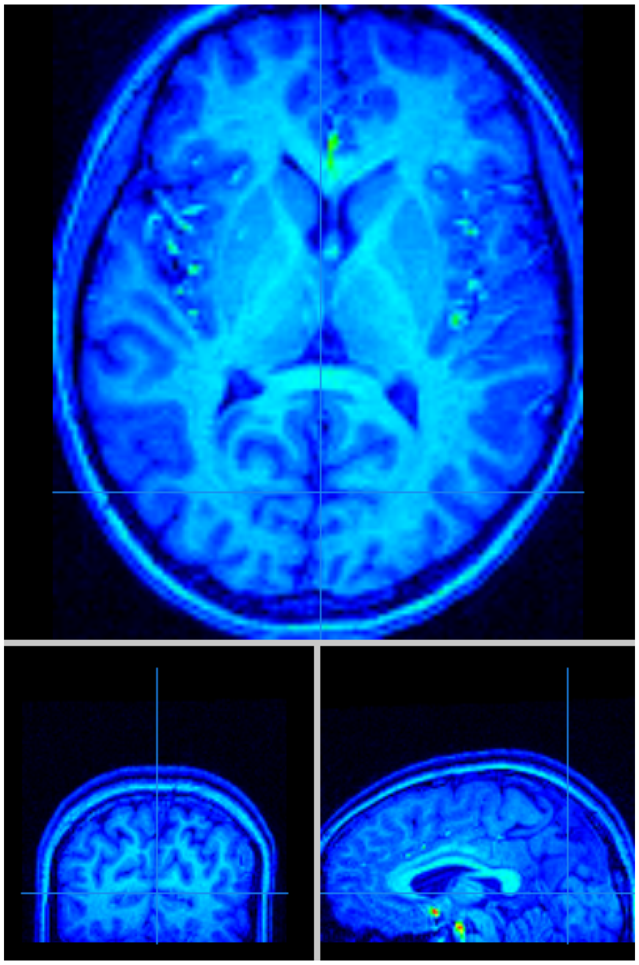
Figure 2. A typical brain MR image. Top image is axial view. Left image is coronal view. Right image is sagittal view.

score label and a site label.

Each site label is simply which site the brain MRI is from, one of the following 17 sites (corresponding class as an integer):

- California Institute of Technology (0)
- Carnegie Mellon University (1)
- Kennedy Krieger Institute (2)
- Ludwig Maximilians University Munich (4)
- NYU Langone Medical Center (5)
- Olin, Institute of Living at Hartford Hospital (7)
- Oregon Health and Science University (6)
- San Diego State University (10)
- Social Brain Lab, BCN NIC UMC Groningen and Netherlands Institute for Neurosciences (9)
- Stanford University (11)
- Trinity Centre for Health Sciences (12)
- University of California, Los Angeles (13)
- University of Leuven (3)
- University of Michigan (14)

Site	+ Quality	- Quality	Total MRIs
0	35	3	38
1	23	4	27
2	44	11	55
3	48	16	64
4	45	12	57
5	158	26	184
6	27	1	28
7	19	17	36
8	46	11	57
9	22	8	30
10	13	23	36
11	0	40	40
12	40	9	49
13	72	28	100
14	35	110	145
15	96	5	101
16	19	37	56
-	742	361	1103

Table 1. Number of examples per site label class and quality label class

- University of Pittsburgh School of Medicine (8)
- University of Utah School of Medicine (15)
- Yale Child Study Center (16).

Thus, the site label is for one of 17 classes.

Each image has at least one but up to 3 quality scores (-1, 0, or +1), given by a human rating the quality of the MRI. To build the quality score label, we averaged the human ratings and assigned a quality score of 1 if the average rating was greater than 0; the MRI was assigned a quality score of 0 is the average rating was less than or equal to 0. Thus, the quality score label is for one of 2 classes. Because there were three human raters for quality, and these labels did not always agree, the quality labels are somewhat noisy. We will present a benchmark for quality accuracy and site accuracy in a later section.

3.2. Class Imbalance

In the ABIDE II dataset, both the site and quality score labels are victims of class imbalance, which can heavily impact the performance of any algorithm [1]. Table 1 shows the number of examples in each class for each label.

We can see here that about 70% of the data has a 'good' quality label of 1, so only 30% has a 'bad' quality label of 0. It is also important to note that several sites (5, 14, 15, 13) have significantly more examples than other sites. In addition, some of the sites (11, 14) are highly correlated with poor quality MR images. while other sites (0, 1, 6, 15) are highly correlated with good quality MR images.

In our data pipeline, each training batch is rejection resampled to contain 50% good quality and 50% bad quality MR images. Rejection resampling for both labels at the same time proved inefficient, but a better solution to this dual class imbalance should be developed. In our results, we note that balancing just the quality label for each batch mitigates some of the class imbalance for the site labels as well.

4. Model and Methods

4.1. Layers

Because we have three-dimensional images, we utilize three-dimensional convolutions in each architecture. Each kernel for the three dimensional convolutional layer spans 4 pixels. In the discriminators, we utilize strides of 1. In the generator, we utilize strides of 2 for each convolutional layer to allow downsampling of the image.

The generator also has transpose convolutional layers, with a kernel size of 4 and stride of 2. This allows for image upsampling.

Because the ABIDE II dataset is not very large compared to many typical datasets used in computer vision tasks, we apply two types of regularization to prevent overfitting: dropout and batch normalization. Both dropout and batch normalization are applied in the discriminators and the generator. Dropout randomly drops neurons in the layers to which it is applied with a pre-specified probability. We used a dropout probability of 0.3. Dropout was applied only when training. In addition to acting as a slight regularizer, batch normalization allows us to standardize training examples over each batch and standardize testing data with a running average taken from batches during training.

We utilize a few types of activation functions. With a convolutional layer, the Leaky ReLU was used. For each element x of a tensor, the resulting activation after passing through the Leaky ReLU is $\max(\alpha \cdot x, x)$ with $\alpha = 0.1$. Leaky ReLU is advantageous because it avoids saturated or killed gradients. With the transposed convolutional layers, we utilize the hyperbolic tangent, as this is standard for most transposed convolutional layers [4]. Thus, for each element x of a tensor, the resulting activation after passing through the hyperbolic tangent layer is $\tanh(x)$.

The last 2 layers of the quality and site discriminators are affine, or fully connected, layers. These allow us to predict for a certain number of classes, here 2 for the quality discriminator and 17 for the site discriminator. The last affine layer outputs logits for each class, so we can calculate softmax loss or cross entropy loss or take the $\arg\max$ to find a prediction.

4.2. Discriminator Architectures

Although we experiment with the architecture of the generator later on, we keep the discriminator architectures the same. The output of the generator, which is of the same size and shape as the original MR images, is the input to each of the discriminators, whose architecture is detailed below. Both discriminators have exactly the same architecture, save for the last layer, which is a fully connected layer with the number of units equal to the number of classes for the label (2 for quality, 17 for site).

1. 3D convolution with 32 kernels of size 4 and stride 1
2. Max pooling with window size of 2 and stride of 2
3. Leaky ReLU with $\alpha = 0.1$
4. 3D convolution with 16 kernels of size 4 and stride 1
5. Max pooling with window size of 2 and stride of 2 2
6. Batch normalization with momentum of 0.1
7. Leaky ReLU with $\alpha = 0.1$
8. Dropout with keep probability of 0.7
9. 3D convolution with 8 kernels of size 4 and stride 1
10. Max pooling with window size of 2 and stride of 2
11. Batch normalization with momentum of 0.1
12. Leaky ReLU with $\alpha = 0.1$
13. Dropout with keep probability of 0.7
14. 3D convolution with 1 kernel of size 4 and stride 1
15. Max pooling with window size of 2 and stride of 2
16. Fully connected layer with 100 units
17. Leaky ReLU with $\alpha = 0.1$
18. Dropout with keep probability of 0.7
19. Fully connected layer with 17 units (2 units) for site discriminator (quality discriminator)

4.3. Loss

To calculate loss for each network, we examine the logits produced by each of the discriminator networks. We compare the logits to the true labels by using the cross entropy loss.

Site Loss: For examples $i = 1, \ldots, n$, and classes $j = 0, .., 16$, we are given true site labels y_i^s for each MR image x_i. Each label is one-hot encoded, which means $y_{ij}^s = 1$ if x_i belongs to class j and $y_{ij}^s = 0$ otherwise. In addition, the site discriminator produces probability p_{ij}^s of x_i belonging to class j. Then, we can define the cross entropy loss for site as follows:

$$L^s = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=0}^{16} y_{ij}^s \log(p_{ij}^s) \qquad (1)$$

Quality Loss: For examples $i = 1, \ldots, n$, and classes $k = 0, 1$, we are given true site labels y_i^q for each MR image x_i. In addition, the quality discriminator produces probability p_{ik}^q of x_i belonging to class k. Then, we can see that the cross entropy loss simplifies because quality prediction is a binary classification problem. Instead of using a one-hot encoding, we can simply use y_i^q and p_i^q as follows:

$$L^q = -\frac{1}{n}\sum_{i=1}^{n} [y_i^q \log(p_i^q) + (1 - y_i^q)\log(1 - p_i^q)] \qquad (2)$$

De-biasing Generator Loss: The generator loss is where we can tie together our two objectives: maximize quality prediction and minimize site prediction. The generator modifies input images, and the output is fed into the site discriminator and quality discriminator in parallel. Thus, we update the discriminator based on the site loss and the quality loss. To maximize quality prediction, we minimize the associated loss L^q. To minimize site prediction, we maximize the associated loss L^s, equivalent to minimizing $-L^s$. Since we aim to minimize L^G, we simply add the two functions to minimize, resulting in $L^q - L^s$. We add W_q, W_s as weights to allow for tuning of the loss function. Thus, the loss of the de-biasing generator is the weighted difference between the L^s and L^q:

$$L^G = W_q \cdot L^q - W_s \cdot L^s \qquad (3)$$

4.4. Training

When training, we update the model in 3 steps. First we feed a batch of data through the network. Then we update the de-biasing generator based on the site and quality predictions. Next, we update the quality discriminator. Lastly, we update the site discriminator.

For each of the three components (de-biasing generator, quality discriminator, and site discriminator), the update step is made using Adam with a learning rate of 0.0001 and momentum of 0.5. Adversarial networks are notoriously finicky with respect to convergence. Salimans et. al /citeGANtraining also recommend testing lower learning rates. In addition, we use a small batch size due to memory constraints, so we employ a low learning rate to avoid overshooting at each training step. Having tested learning rates in the range 10^{-6} to 10^{-1}, the selected learning rate 10^{-4} had the best convergence properties.

5. Experiments and Results

5.1. Software and Hardware

We implemented the architecture using Tensorflow in Python. We employed the use of two MRI-specific Python packages: NiBabel to read the MR image files (.nii/.nii.gz) and NiLearn to resample the MR images.

Visualizations of the original MR images and outputs of the de-biasing generator were produced with the use of Mango (Multi-image Analysis GUI), a medical research image viewing software.

I based my implementation on the pix2pix [4] implementation as well as on CS 231n assignment 3.

In addition to using 8 vCPUs for data input and output, much of the computation was done on two NVIDIA Tesla K80 GPUs, each with 12GB of memory.

5.2. Benchmarks and Metrics

A natural control for our framework is the same setup as Figure 1, but with the generator replaced by the identity function. Thus, the original image is fed into the quality discriminator and the site discriminator without modification. This is mathematically equivalent to training two separate convolutional neural networks, one to predict site and another to predict quality.

Utilizing the discriminator architecture detailed in section 4.2, we achieved 85% accuracy for quality prediction (due to noisy labels) and 96% accuracy for site prediction. We note that the 85% accuracy for quality prediction achieved with no image modification represents a ceiling on quality accuracy we can expect from an adversarial approach. With a reduction in site information in each debiased MR image, we can expect some marginal decrease in quality accuracy.

5.3. Three Conditionally Generative Architectures

The first architecture approached for the de-biasing generator was that of pure convolution. This network's architecture was as follows:

1. 3D convolution with 16 kernels of size 4 and stride 1
2. Batch normalization with momentum of 0.1
3. Leaky ReLU with $\alpha = 0.1$
4. Dropout with keep probability of 0.7
5. 3D convolution with 8 kernels of size 4 and stride 1
6. Batch normalization with momentum of 0.1
7. Leaky ReLU with $\alpha = 0.1$
8. Dropout with keep probability of 0.7
9. 3D convolution with 4 kernels of size 4 and stride 1
10. Batch normalization with momentum of 0.1
11. Leaky ReLU with $\alpha = 0.1$
12. Dropout with keep probability of 0.7
13. 3D convolution with 1 kernel of size 4 and stride 1

The images produced by this architecture are shown in Figure 3. They tended to be overly blurry. In addition, quality loss increased after several epochs (see Figure 4), as the convolutions blurred out many of the minute details used in quality detection. In the next attempt, we want to search for a method that retains a higher level of resolution.

The second generative architecture approached was that of an encoder/decoder. This network's architecture was as follows:

1. 3D convolution with 16 kernels of size 4 and stride 2
2. Leaky ReLU with $\alpha = 0.1$
3. 3D convolution with 32 kernels of size 4 and stride 2
4. Batch normalization with momentum of 0.1
5. Leaky ReLU with $\alpha = 0.1$
6. Dropout with keep probability of 0.7
7. 3D transposed convolution with 32 kernels of size 4 and stride 2
8. Batch normalization with momentum of 0.1

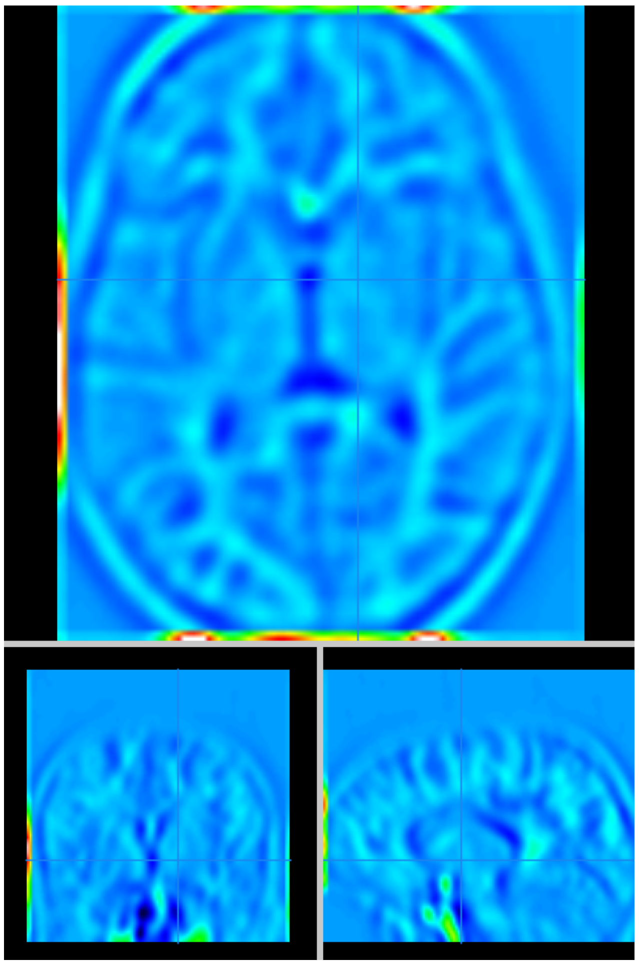
Figure 3. Brain MR image output from first de-biasing attempt using several purely convolutional layers.

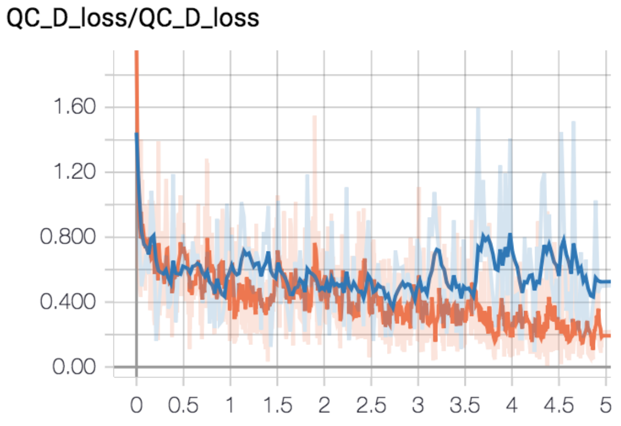

Figure 4. L^q with respect to time for the first de-biasing attempt using several purely convolutional layers. Blue line is testing. Orange line is training

9. tanh
10. 3D transposed convolution with 1 kernels of size 4 and stride 2

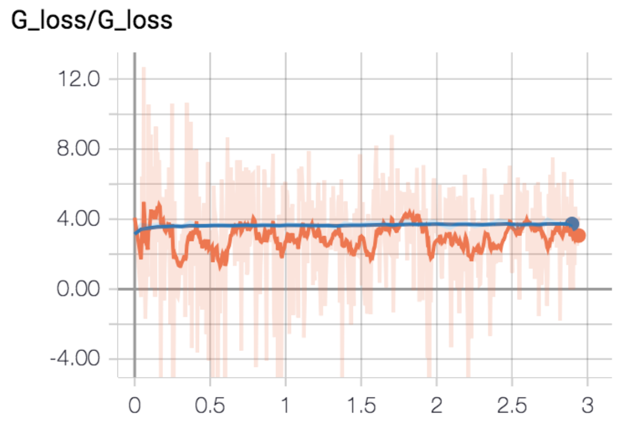

Figure 5. L^G with respect to time for the second de-biasing attempt using convolutional then transposed convolutional layers. Blue line is testing. Orange line is training.

This architecture did not have very smooth convergence for either loss. Variance was high for several epochs. After adding smoothing, the generator loss looked stagnant, as shown in Figure 5. The images produced by this architecture are shown in Figure 6. They had a very distinct checkerboard pattern. We found that this phenomenon had been seen before [6]. The advice given by the authors was to use a different method of upsampling or to supplement upsampling with another operation that prevents information loss.

Isola et. al [4] maintained that the checkerboard issue was mitigated with the use of skip connections, which preserved the scale of resolution and granularity of images before each iteration of downsampling. Utilizing skip connections within the previous architecture yielded the following architecture:

1. 3D convolution with 16 kernels of size 4 and stride 2
2. Leaky ReLU with $\alpha = 0.1$
3. 3D convolution with 32 kernels of size 4 and stride 2
4. Batch normalization with momentum of 0.1
5. Leaky ReLU with $\alpha = 0.1$
6. Dropout with keep probability of 0.7
7. 3D transposed convolution with 16 kernels of size 4 and stride 2
8. Batch normalization with momentum of 0.1
9. tanh
10. Resizing/padding to be of the same size as the output of Layer 2
11. Concatenation with Layer 2
12. 3D transpose convolution with 4 kernels of size 4 and stride 2

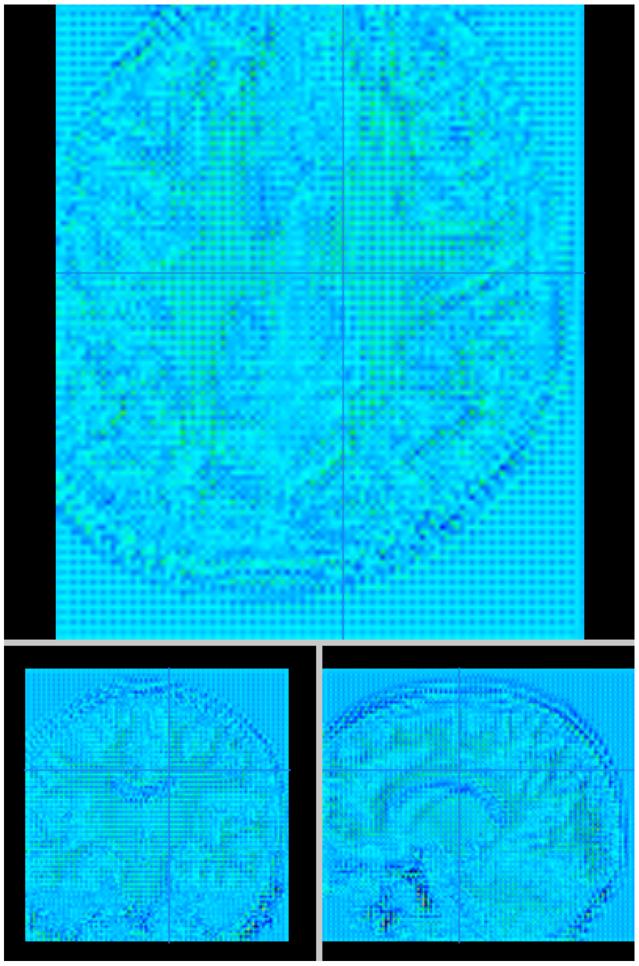

Figure 6. Brain MR image output from second de-biasing attempt using convolutional then transposed convolutional layers.

13. Batch normalization with momentum of 0.1
14. Leaky ReLU with $\alpha = 0.1$
15. Dropout with keep probability of 0.7
16. 3D convolution with 1 kernel of size 4 and stride 1

De-biased images resulting from this architecture can be seen in Figure 7. This image is more granular than the results from our first attempt but does not contain the checkerboard artifacts from our second attempt.

Over the course of training, quality loss decreased steadily (Figure 8). Quality prediction on these de-noised images was 76% on the test set. A confusion matrix for these quality predictions is shown in Table 2. Site prediction on these de-noised images was 12% on the test set.

6. Conclusion and Future Work

We have surveyed the use of several generative architectures to remove site effects from data while preserving quality information, with fairly successful results. This method-

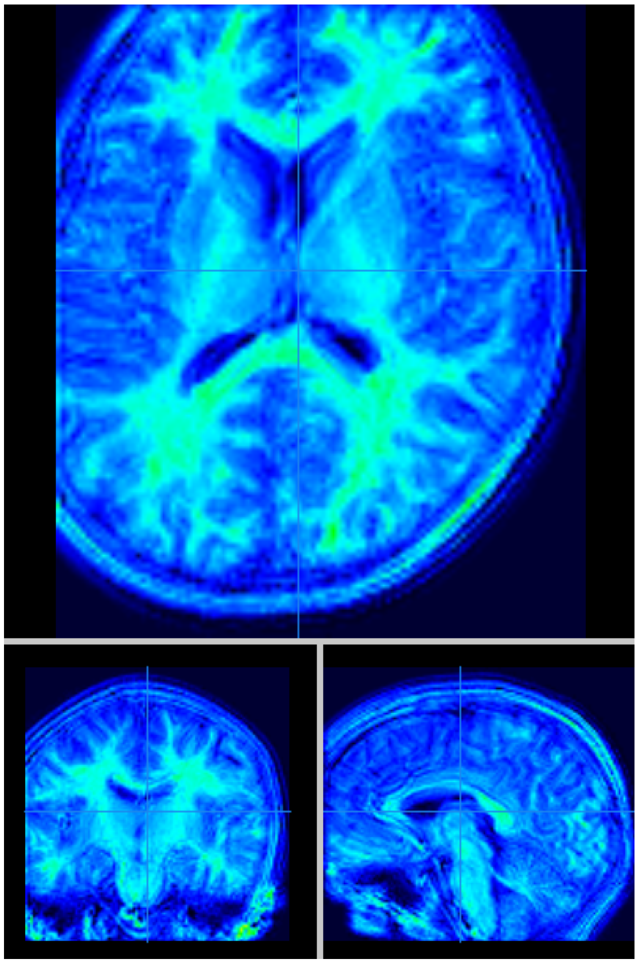

Figure 7. Brain MR image output from third de-biasing attempt using convolutional then transposed convolutional layers with skip connections.

	+ Quality Predicted	- Quality Predicted	Total
+ Quality True	47	19	66
- Quality True	5	29	34
Total	52	48	100

Table 2. Confusion matrix for predictions on test set

ology can be applied to many different applications, not even purely image-related applications.

With more time, I would further tune each network and experiment with more complex and deeper CNN architectures. The largest bottleneck here was compute time. With more compute power, I would be able to use larger batch sizes, so training would go more smoothly and training loss would have less variance than during my experiments.

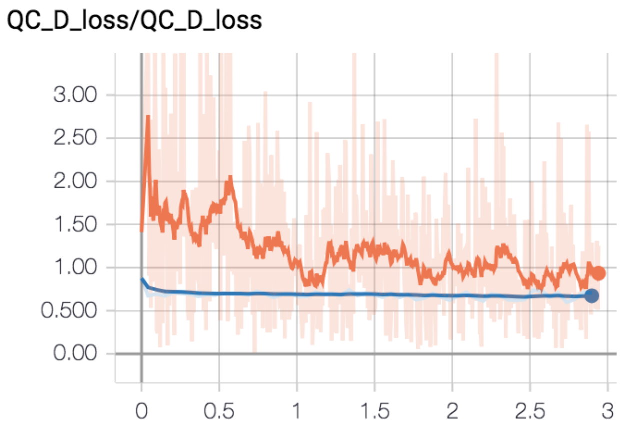

Figure 8. L^q with respect to time for the third de-biasing attempt using convolutional then transposed convolutional layers with skip connections. Blue line is testing. Orange line is training.

References

[1] M. Buda, A. Maki, and M. A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *CoRR*, abs/1710.05381, 2017.

[2] O. Esteban, D. Birman, M. Schaer, O. Koyejo, R. Poldrack, and K. Gorgolewski. Mriqc: Advancing the automatic prediction of image quality in mri from unseen sites. 2017. PLOS ONE 12(9):e0184661.10.1371/journal.pone.0184661.

[3] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial networks. *CoRR*, abs/1406.2661, 2014.

[4] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016.

[5] M. Mirza and S. Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014.

[6] A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016.

[7] B. Zhang, B. Lemoine, and M. Mitchell. Mitigating unwanted biases with adversarial learning. 2018. CoRR, abs/1801.07593.