MRI Image Compression with Deep Learning

Alex Thornton
Dept. of Electrical Engineering
Columbia University
New York, NY
a.thornton@columbia.edu

***Abstract*—The purpose of this research is to create a model which can efficiently store slices of MRI scans. Efficiency is explored across two optimization domains; minimization of generational loss through recursive compression, and compressed space minimization. Some techniques and frameworks explored include k-wavelet approximation, and various auto-encoder frameworks for neural networks. Training and test data was provided by FastMRI, a joint venture between NYU and Facebook (Meta).**

***Index Terms*—Wavelet, Medical Imaging, Compression, Sparse Auto Encoder, Convolutional Neural Network**

I. Introduction

MEDICAL Resonance Imaging (MRI) scans have become incredibly useful for monitoring the health of our brains. The ability to peer inside our heads to find tumors, structural irregularities, lesions, and other ailments saves lives, and is an important tool to medical professionals for both diagnosis and treatment. This is a technology which is very important to me personally; my mother suffers from Multiple Sclerosis (MS), and undergoes MRI scans multiple times each year in consultation with her neurologist. For such an important device, MRI has only existed for a short time. The first design schematic for an MRI machine was scribbled into a napkin at a Eat'n Park Big Boy Restaurant in Pittsburgh by Paul Lauterbur in the 1970's, and would eventually lead to the awarding of the Nobel Prize in Medicine [1].

The underlying principles of MRI are very clever, and involve exploiting multiple convenient tricks in particle physics and signal processing. Firstly, we can take advantage of the fact that our bodies are largely comprised of water, whose individual particles are weakly magnetic with magnetization $\boldsymbol{M}$. Subjecting these particles to an external magnetic field $\boldsymbol{B}$, induces a torque, which forces the particles to precess. The precession is defined as a first order differential equation by the *Bloch Equation* in equation 1. The magnetic flux caused by the precession can be picked up by an RF coil, allowing a measurement signal S to be received.

$$\frac{\partial \boldsymbol{M}}{\partial t} = \gamma \boldsymbol{M} \times \boldsymbol{B} \tag{1}$$

Selecting a clever choice for the external magnetic field $\boldsymbol{B}$ allows another crucial insight of MRI, *spatial encoding* through selective excitation. Equation 2 shows how a magnetic field can be varied along the z-axis to resonate particles along a plane by introducing magnetic gradients G_x and G_y.

$$\boldsymbol{B} = (B_0 + G_x(t)x + G_y(t)y)\hat{\boldsymbol{k}} \tag{2}$$

Putting it all together, equations 3 and 4 show how an image can be reconstructed from the received signal S and the collective effects of the magnetic gradients. This can be seen as an inverse 2D spatial Fourier Transform of the signal seen at the RF coil in the MRI, using the collective effects of the external magnetic field gradients as the kernel of the transform. The process of reconstructing a particular 'slice' is shown in Figure 1 [2].

$$k_x(t) = \gamma \int_0^t G_x(\tau)\mathrm{d}\tau, \qquad k_y(t) = \gamma \int_0^t G_y(\tau)\mathrm{d}\tau \tag{3}$$

$$|\boldsymbol{M}_{xy}(x,y)| \propto \int_{k_x}\int_{k_y} S(k_x,k_y)e^{i(k_x x + k_x y)}\mathrm{d}k_x\mathrm{d}k_y \tag{4}$$

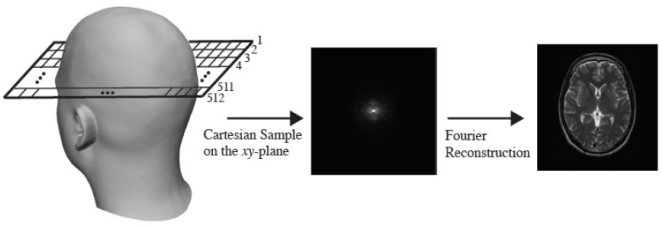

Fig. 1. Sampling a 'slice' of brain, and reconstruction [2]

The health benefits and significance of the information provided by reconstructed MRI scans cannot be understated, but there is inherently a lot of information that needs to be processed and stored. How can we optimize the way we interact with the data from MRI scans to maintain data integrity while also allowing for scalability to massive data sets?

II. Data

With permission from New York University, I have been granted access to the FastMRI database, a joint project between NYU and Facebook AI (Meta). FastMRI consists of raw k-space data collections on thousands of brain scans before any compression is applied [3]. Additionally, an open source toolbox for manipulating the data is available on GitHub.

For the purposes of this project, a small subset of the FastMRI dataset was used. The multicoil_test_dataset alone

was over 34 GB as a zipped file alone, and the training and validation files were much larger! From the multicoil_test_dataset, 27 brain scans were randomly chosen, each with 14-16 slices each, and 1 brain set aside as the 'test' brain to evaluate the models. Raw k-space data was combined over all coils to produce an image for each slice, and each slice was resized to a standard 256x256 resolution. Figure 2 shows all 16 slices for a single brain from this data sampling.

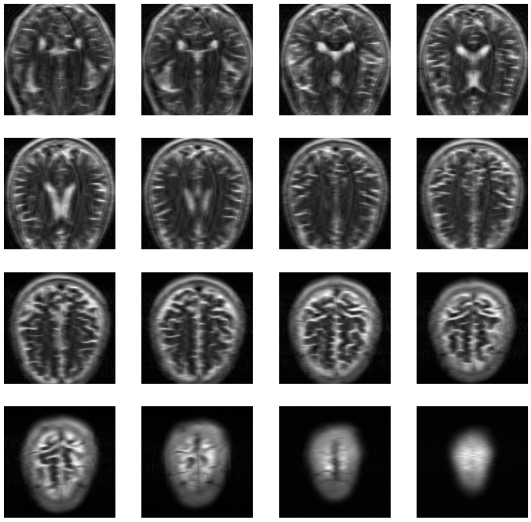

Fig. 2. Slices of single FastMRI brain scan

III. Compression Use Cases

A. Generational Loss

If you've ever taken a photocopy of a photocopy, you can see the immediate decline in image quality for subsequent scans. Thus, the term *generational loss* is sometimes informally dubbed the 'photocopier effect'. Generational loss is an inherent problem to any compression scheme that is lossy, or performed more than once [4]. But why would we care about this? After all, we're only going to compress and store and image once, right?

Consider the situation where translating between an image and its embedding is lossless. Generational loss wouldn't be an issue, and you could switch back and forth between the image and its embedding without altering the quality of the image. A subtle advantage here would be that once you decompress the image, you wouldn't need the compressed embedding anymore in memory. If we have a large quantity of images 'opened' in their decompressed state for some operation, we can delete the compressed versions and simply recover them after we're finished. In a use case where we want to perform operations on tens of thousands of images in the decompressed state at the same time, this could free up a lot of memory.

B. Minimal Embedding Size

Perhaps we don't care about generational loss, and are looking to optimize for the space taken up by our image embeddings. For an arbitrary compression projection transformation C, we can define this optimization in terms of an objective and constraint as shown in equation 5.

$$\min ||C(X)||_0 \quad \text{s.t.} \quad -\delta \leq X - C^{-1}(C(X)) \leq \delta \tag{5}$$

Since we are enforcing a hard constraint on the embedding size (L_0 norm of $C(X)$), we cannot relax to a L_1 objective minimization. Some recent research in MRI compression has applied the Alternating Directions Method of Multipliers (ADMM) technique to minimize the Lagrange dual of the above optimization, but this requires a known desired sparsity to error trade off constant β [5]. Rather than trying to optimize in the same way, a deep learning approach is preferred. We will evaluate a sparse auto-encoder (SAE) neural network with a fixed number of neurons in the latent layer to enforce the hard L_0 constraint. Many SAEs enforce sparsity at the latent layer with a KL divergence penalty for latent activations added to the overall loss function of the network. While this would expose a sparse solution, we would need to guess an appropriate sparsity to objective loss trade-off term, and eliminate latent neurons below some threshold. A more appropriate approach for this research will be to try a couple educated guesses for a fixed latent layer size, train and evaluate each network, and compare performance.

IV. Minimizing Generational Loss

A. JPEG

The JPEG standard involves only a few simple operations. As shown in Figure 3, the algorithm consists mainly of a discrete cosine transform, quantization by a preset table, and lossless data encoding with Huffman compression scheme [6]. While this model can extend without loss of generality to basically any image, it leaves a lot to be desired if the image set we want to compress have common features.

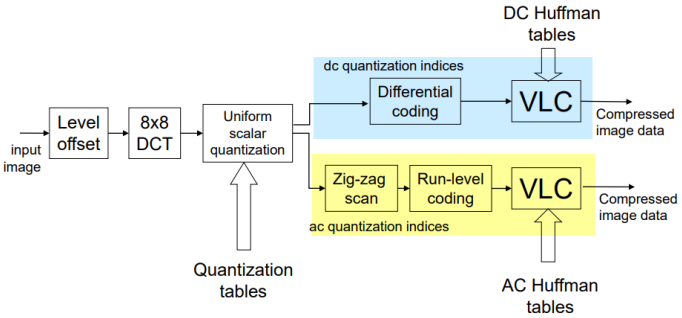

Fig. 3. JPEG encoding model [6]

The JPEG standard was not designed with consideration for iterative compression. As shown in Figure 4, an input image is reduced to pure noise after only a few iterative compressions. JPEG2000, which is based on the wavelet transform instead

of the discrete cosine transform, is also not immune to this, because it has a nonlinear quantization for every iteration [7].

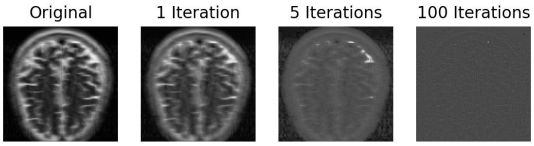

Fig. 4. Recursive JPEG Compression

B. Quantized K-Wavelet

As JPEG2000 pioneered, wavelets are an excellent compression basis technique. This is especially true for MRI scans, whose structured form is well suited for wavelets. Finding a sufficient K number of float32 quantized level-3 wavelet coefficients to keep during compression is a balancing act between reconstuction error and K, as shown in equation 6. (To explicitly call out *which* wavelet, the mother wavelet for everything here is the Debacheries class 4 wavelet). Iteratively solving this optimization via ADMM would require apriori knowledge of the desired sparsity to error tradeoff term β. Instead, for this project, I averaged the Frobenius error for the entire test set for a large number of K, and used a binary search to find the K which resulted in about 5% error. The result, 2896, was simply upper bounded to a clean 3000 value for K.

$$J(K) = K + \frac{\beta}{2} ||X - W_{inverse} P_K(WX)||_F^2 \qquad (6)$$

Setting $K = 3000$ and performing iterative compression, generational loss is all but eliminated. As shown in Figure 5, converting to k-wavelets and back many times is able to maintain the important information between iterations.

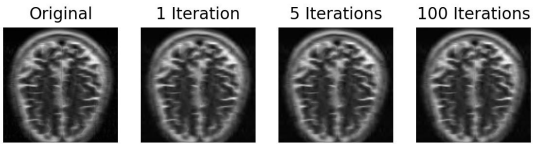

Fig. 5. Recursive Quantized K-Wavelet Compression

C. Analysis

Clearly k-wavelet approximation for the chosen K worked well, but how does it extend without loss of generality? Generational loss for JPEG is modelled in Figure 6, where subsequent iterations drift further from the original image in the image space until the result is unrecognizable.

The fundamental improvement with quantized k-wavelet compression is that image quality is only lost in the first compression, every subsequent iteration is lossless. Since we chose $K = 3000$ to approximate a 5% error in the first iteration, that

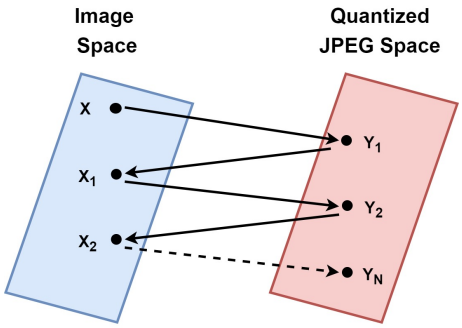

Fig. 6. JPEG Compression Space

is the most error we will ever incur. Figure 7 shows how this model contains the drift from Figure 6, and allows for lossless compression after only a single compression.

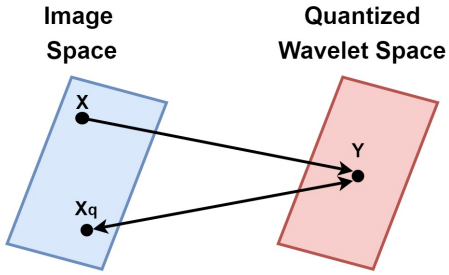

Fig. 7. K-Wavelet Compression Space

Plotting the generational loss over 100 instances (averaged over the test set) for both JPEG and quantized k-wavelet compression in Figure 8, we see immediately that the latter prevents the problem with high efficiency. The error in the first iteration was arbitrarily chosen to be 5%, but could easily be reduced by selecting a higher K, highlighting the trade off between sparsity and performance.

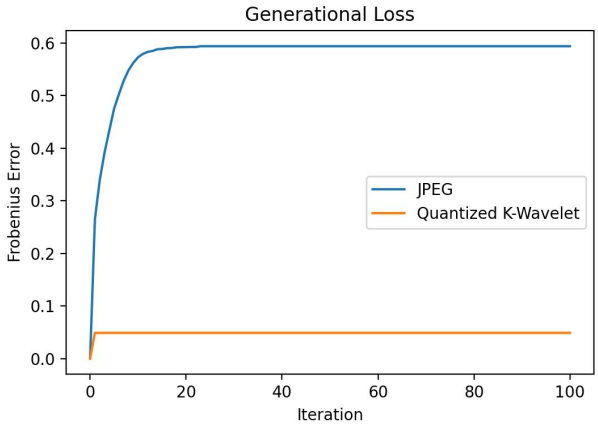

Fig. 8. Generational Loss Compared (JPEG vs Quantized K-Wavelet)

So this model solves the generational loss problem, but how much space does it take? Equation 7 shows the calculation for

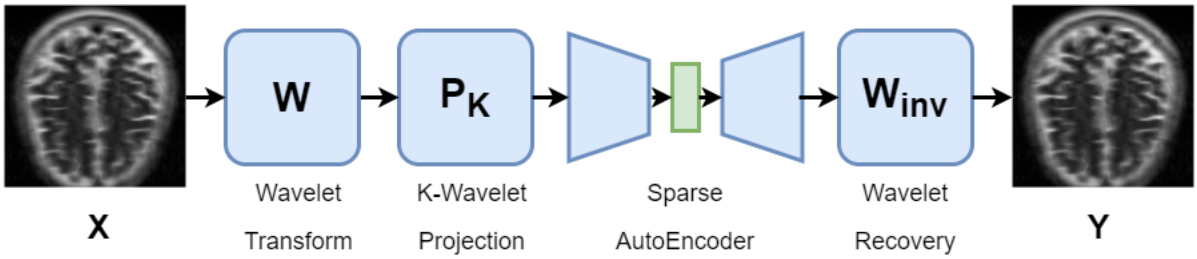

Fig. 9. Hybrid Wavelet Sparse Auto Encoder

the number of bits S required to store our compression. There are 3000 coefficients stored as a float32, but we also need the indices of the coefficients in the 256x256 map. Combined, each compression results in *14.4 KB*, which is very respectable when compared to its JPEG and JPEG2000 contemporaries. But if we're willing to part with the generational loss quality, and maybe allow a higher reconstruction error, can we compress even further?

$$S = K(32 + 2\log_2 256) \tag{7}$$

V. Compression via Deep Learning

A. Wavelets with Auto-Encoder?

Since the wavelet compression did well in the previous use case, it stands to reason that wavelets could serve as good features for a neural network to learn. Consider the proposed model in Figure 9, where an image is preprocessed by k-wavelet compression, and those features are compressed even further with a sparse auto-encoder (SAE) network, before ultimately being reconstructed with an inverse wavelet transform. Maybe by combining a deep learning framework like an SAE with some traditional signal processing with the wavelet transform, we can do even better for space compression on the bit level.

B. Shallow Feed-Forward Neural Network

A naive first approach might be to consider the simplest neural network possible, a (1, 256, 256) feed-forward linear layer passed to a latent layer dimension K, and reconstructed into by another linear (1, 256, 256) output layer. This actually didn't do too badly, it was able to recognize that the wavelets of importance were mostly in the top left corner, but reconstructing the image yielded strange tartan like textures, as seen in Figure 10.

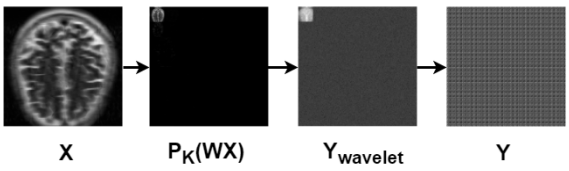

Fig. 10. Feed-Forward Neural Network Wavelet Recovery Failure

C. Convolutional Neural Network

Convolutional neural networks (CNN), which are the preeminent technique for SAEs in computer vision, provide a more robust framework than the feed-forward model. The CNN constructed for this project encodes an image into 33x33 embeddings across 16 filters, which can then be compressed further in a latent layer before decoding back into an image. The wavelets interestingly performed even worse with this architecture, even with no latent layer for a 16*33*33 embedding, as shown in Figure 11. Perhaps throwing out the wavelet features and having a CNN that learns only from the images themselves will be better.

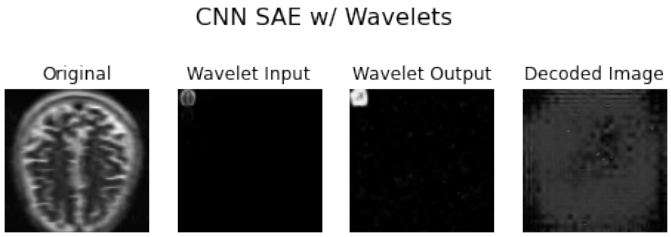

Fig. 11. CNN Wavelet Recovery Failure

D. Convolutional Neural Network, No Wavelets

Instead of pursuing the hybrid wavelet-SAE model, a return to basics provides some better results. Without the wavelet features, passing in the images themselves to the CNN would is much better at reconstructing the test set. A number of choices for the latent size K were carefully chosen at factors of the encoder output shape 16*33*33. For each of these K, a model was trained for 250 epochs, and half of the training set used as a batch.

Figures 13 and 14 shows how even with very small latent layers, the reconstructed images still bears resemblance to the original. Averaging error over the test set for each model, Figure 12 shows the degradation in Frobenius error.

An important comparison to the JPEG generational loss which topped out at about 0.6 is that while the losses for the smaller K models approach that error level, the reconstructions are much more representative of the original image features. Future work may want to investigate a better error metric that differentiates some feature recovery over pure random noise.

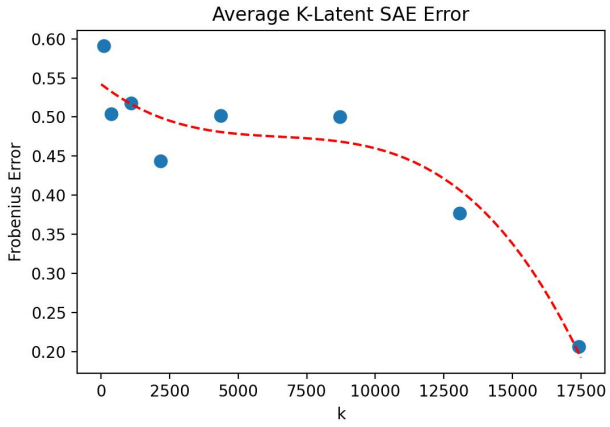

Fig. 12. CNN Recovery Error

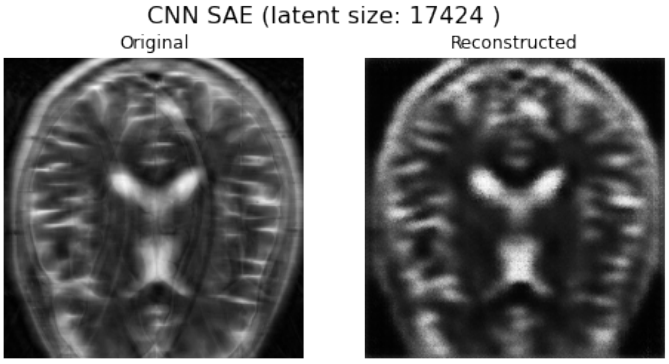

Fig. 13. CNN Recovery: K = 12*33*33

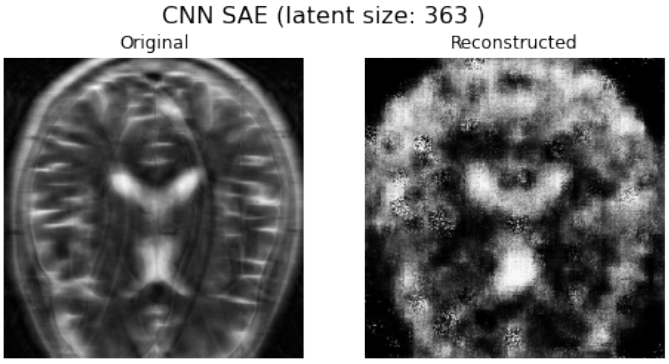

Fig. 14. CNN Recovery: K = 11*33

Evidently, the CNN SAE model allows for *very* small compressions that still somewhat capture the features of the original image. For the smaller latent layers tried, for $K = 363$ and $K = 99$, we were able to achieve compression down to 1.448 KB and 0.396 KB. This comes at a high cost in image quality, but for some applications, this might be useful. For example, if a CNN SAE can be trained to pick up on locations of tumors in brain scans, and maintain the general layout of the scan and the tumor within it, then something on the order of this size could be recovered and be meaningful.

VI. Conclusion

Both of the models discussed in this research have relevant applications and use cases. As described in the generational loss section, being able to switch between an excellent approximation of an image and its embedded format without loss could allow you to only keep one. If your situation requires having many images open at once in their uncompressed state, you could wipe the embeddings from memory and simply recover them later. This model came in with a respectable compression size as well, and the arbitrary 5% accuracy is scalable as well.

For applications requiring the smallest compression size possible, and willing to forgo the generational loss feature and a hefty toll on image quality, the CNN SAE may be useful. As discussed in the CNN SAE without wavelets section, image embeddings that capture locations of tumors or important features of brain scans could be compressed as low at less than a kilobyte using a derivative model presented in this paper, albeit with a different training data set.

For the overwhelming majority of applications, I can recommend the quantized k-wavelet model, which utilizes no deep learning frameworks at all. The benefit of generational loss immunity, coupled with the respectable compression size and recovery error make it a strong contender for any application that doesn't need extremely small compressions. I would imagine that this model would scale well to any number of applications both within medical imaging, and more generally. Well structured images with smooth features will be captured well by the wavelet transform, and this approach would likely work well for compressing those images with reasonable embedding sizes and quality loss.

References

[1] "Paul Lauterbur and Sir Peter Mansfield for MRI: In our series Focusing on Nobel Prize winners that have contributed to cardiovascular medicine, Mark Nicholls looks at the work of two scientists recognized for their discoveries concerning magnetic resonance imaging (MRI)," *European Heart Journal*, vol. 40, pp. 1898–1899, 06 2019.

[2] J. Wright and Y. Ma, *High-Dimensional Data Analysis with Low-Dimensional Models: Principles, Computation, and Applications*. Cambridge University Press, 2022.

[3] F. Knoll, J. Zbontar, A. Sriram, M. J. Muckley, M. Bruno, A. Defazio, M. Parente, K. J. Geras, J. Katsnelson, H. Chandarana, Z. Zhang, M. Drozdzalv, A. Romero, M. Rabbat, P. Vincent, J. Pinkerton, D. Wang, N. Yakubova, E. Owens, C. L. Zitnick, M. P. Recht, D. K. Sodickson, and Y. W. Lui, "fastmri: A publicly available raw k-space and dicom dataset of knee images for accelerated mr image reconstruction using machine learning," *Radiology: Artificial Intelligence*, vol. 2, no. 1, p. e190007, 2020. PMID: 32076662.

[4] J. Sneyers, "Why JPEG is like a photocopier." https://cloudinary.com/blog/why_jpeg_is_like_a_photocopier. [Online; accessed 03-May-2022].

[5] V. Corona, Y. Dar, G. Williams, and C.-B. Schönlieb, "Regularized compression of mri data: Modular optimization of joint reconstruction and coding," 2020.

[6] B. Girod, "JPEG Standard." http://web.stanford.edu/class/ee398a/handouts/lectures/08-JPEG.pdf. [Online; accessed 01-May-2022].

[7] B. Girod, "JPEG2000 Standard." http://web.stanford.edu/class/ee398a/handouts/lectures/10-JPEG2000.pdf. [Online; accessed 07-May-2022].

Thank You!

Thank you Dr. Wright for your advice over the course of this project. I have really valued getting to pick your brain about using brains (neural nets) to compress images of brains! I hope our paths will cross again.

Thank you Mariam for the support in office hours helping through homework sets. I feel like I got a lot more out of this class because I was able to ask you questions!

One More Thing

This project marks the completion of my M.Sc. at Columbia University. I'd like to thank my Mom, Dad, sister Carolyn, brother Matt, and Murphy (pictured below).

Roar Lion Roar!

-Alex