Greedy Learning for Large-Scale Neural MRI Reconstruction

Batu Ozturkler[1], **Arda Sahiner**[1], **Tolga Ergen**[1], **Arjun D. Desai**[1], **Shreyas Vasanawala**[2], **John M. Pauly**[1], **Morteza Mardani**[1], **Mert Pilanci**[1]
Department of Electrical Engineering[1] and Radiology[2]
Stanford University
`{ozt, sahiner, ergen, arjundd, vasanawala, pauly, morteza, pilanci}@stanford.edu`

Abstract

Model-based deep learning approaches have recently shown state-of-the-art performance for accelerated MRI reconstruction. These methods unroll iterative proximal gradient descent by alternating between data-consistency and a neural-network based proximal operation. However, they demand several unrolled iterations with sufficiently expressive proximals for high resolution and multi-dimensional imaging (e.g., 3D MRI). This impedes traditional training via backpropagation due to prohibitively intensive memory and compute needed to calculate gradients and store intermediate activations per layer. To address this challenge, we advocate an alternative training method by greedily relaxing the objective. We split the end-to-end network into decoupled network modules, and optimize each network module separately, thereby avoiding the need to compute costly end-to-end gradients. We empirically demonstrate that the proposed greedy learning method requires 6x less memory with no additional computations, while generalizing slightly better than backpropagation.

1 Introduction

Magnetic resonance imaging (MRI) is a widely used medical imaging modality that provides high resolution information of the soft tissue anatomy, but is limited by long scan times. Deep learning methods have gained popularity for accelerated MRI, improving image quality of reconstructions from undersampled measurements over parallel imaging and compressive sensing (CS) techniques [1, 2]. A typical approach is model-based deep learning, where an iterative algorithm is unrolled for a fixed number of iterations. The iterative algorithm alternates between enforcing data-consistency with the measurement model and a neural proximal operation [3, 4]. Model-based networks are trained with backpropagation where the parameters of the end-to-end network are jointly optimized. The graphics processing unit (GPU) memory requirement of training model-based networks with backpropagation increases linearly with the measurement count and the number of layers in the neural proximal operation, limiting the number of unrolled iterations of the model-based network which in turn limits expressivity [5]. Therefore, improving the memory footprint of training could enable the use of such networks in high-dimensional large-scale imaging applications such as Dynamic Contrast Enhanced (DCE) imaging, or blood flow imaging (4D-flow).

Related Works. Several works have aimed to tackle the memory bottleneck of training model-based deep networks. One line of work uses invertible networks where intermediate activations are calculated by reversing each layer, hence removing the need to store intermediate activations using backpropagation [6, 7, 8]. [9] applies stochastic approximations to the data-consistency layers of model-based networks to reduce memory complexity with increasing number of measurements. It

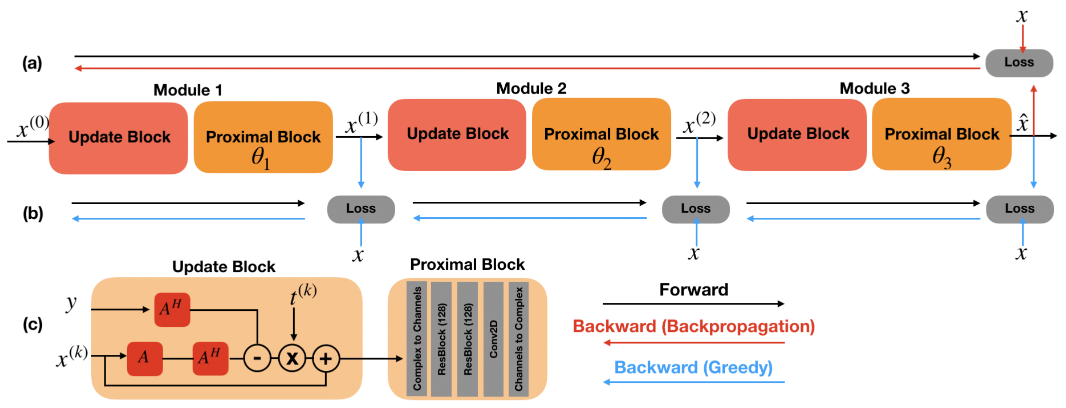

Figure 1: Example network with $N = M = 3$ where N is the number of unrolled iterations, M is the number of modules. (a) Standard end-to-end training with backpropagation. Forward pass is computed through all modules, and gradient updates are performed for all modules in the same backward pass. (b) Proposed greedy learning algorithm. At the end of each proximal block, loss is computed, and a local gradient update is performed on the current module. (c) Update block consists of data-consistency with measurements, and the proximal block consists of residual blocks.

must be noted that these approaches make modifications to the network architecture whereas our method proposes a greedy optimization method that is compatible with any model-based network. Thus, our method can be synergistically combined with any of these approaches to further reduce memory.

Recent works in computer vision propose alternatives to backpropagation for convolutional neural networks (CNNs) in the context of image classification [10, 11, 12]. One approach is layer-wise training, where single hidden-layer subnetworks are trained sequentially until convergence while the previous part of the network remains frozen [11]. Another alternative is to use a greedy learning approach, where a network is split into several smaller network modules, and gradient updates for each module are decoupled, which was shown to achieve a similar generalization performance as backpropagation [12]. Although memory efficiency is not the primary focus of those works, greedy learning can improve the memory footprint of training by performing local gradient updates as opposed to end-to-end gradient updates in backpropagation.

In this work, we propose a greedy training objective for learning model-based networks. We split the end-to-end network into decoupled network modules, and perform gradient updates on each module independently. Since each module has a significantly lower memory footprint, the memory requirement of the overall network decreases as well. Hence, our approach reduces memory when the number of measurements or number of layers is large.

Contributions. Our contributions can be summarized as follows:

- We propose greedy learning for MRI reconstruction, which requires 6x less GPU memory during training while the compute time remains the same.
- We demonstrate that greedy learning can improve generalization performance compared to backpropagation for MRI reconstruction with model-based networks.

2 Background

The forward model of accelerated MRI with parallel imaging and compressive sensing (CS) [1, 2] can be modeled as

$$y = UFSx + \epsilon \tag{1}$$

where y is the observed measurements in Fourier space (k-space), x is the real image we would like to reconstruct, S are coil sensitivity maps associated with the receiver coils used in parallel imaging, F is the Fourier transform matrix, U is the undersampling mask, and ϵ is additive noise. Then, the

Algorithm 1 Greedy Learning of Model-Based Networks

```
Input: Samples or mini-batches $\{(x_k^{(0)}, x_k)\}_{k \leq N_b}$
Initialize: Parameters $\{\theta_m\}_{m \leq M}$
  for $k \in 1, ..., N_b$ do
    for $m \in 1, ..., M$ do
      $x_k^{(m)} \leftarrow f_{\theta_m}(\mathbf{DC}(x_k^{(m-1)}))$
      Compute $\nabla_{\theta_m} \mathcal{L}(x_k^{(m)}, x_k)$
      $\theta_m \leftarrow$ Update Parameters$(\theta_m)$.
    end for
  end for
```

MRI reconstruction problem can be formulated as

$$\hat{x} = \arg\min_x \frac{1}{2}\|Ax - y\|_2^2 + \lambda R(x) \tag{2}$$

where $A = UFS$ denotes the forward model, R is a regularization function and λ is the regularization strength. This optimization problem can be solved in an alternating manner using proximal gradient descent [3]

$$x^{(i+)} = \mathbf{DC}(x^{(i)}) = x^{(i-1)} - 2tA^H(Ax^{(i)} - y) \tag{3}$$

$$x^{(i+1)} = \mathbf{Prox}_R(x^{(i+)}) \tag{4}$$

where i is the iteration number, t is the step size, A^H is the Hermitian transpose of the forward model operator A. $\mathbf{DC}$ denotes data-consistency, and $\mathbf{Prox}_R$ is the proximal operator. In model-based networks, the proximal operator is learned with a CNN with trainable parameters, and the iterative optimization algorithm described above is unrolled for a fixed number of iterations. Then, (4) can be rewritten as

$$x^{(i+1)} = f_{\theta_i}(x^{(i+)}) \tag{5}$$

where f_{θ_i} is the network, and θ_i its parameters at i^{th} iteration. Such a models parameters can be jointly optimized using backpropagation

$$\min_{\theta_1,..,\theta_N} \sum_k \mathcal{L}(f_{\theta_1,..,\theta_N}(y_k, A_k), x_k) \tag{6}$$

where $f_{\theta_1,..,\theta_N}$ is the end-to-end network, y_k is the k^{th} undersampled image, x_k is the k^{th} fully-sampled reference, $\mathcal{L}$ is the loss function, and N is the number of unrolled iterations. In our framework, we use the unrolled neural network described above as the baseline model for comparison.

3 Greedy Learning for MRI Reconstruction

Here, we describe our method for optimizing model-based deep networks. We adapt our method from the greedy learning aproach in [12] for training CNNs, where we extend this to model-based networks. We consider a model-based network with N number of unrolled iterations, where the network is split into M modules, such that $\frac{N}{M}$ is the number of data-consistency steps and proximal steps in each module. For simplicity we outline the case where $N = M$. In particular, let $x^{(0)} = y$ be the input, x the fully sampled reference, $x^{(m)} = f_{\theta_m}(\mathbf{DC}(x^{(m-1)}))$ the output of the m^{th} network module, and N_b is the number of mini-batches. For the m^{th} network module, let the cost function be $\mathcal{L}(f_{\theta_m}(\mathbf{DC}(x^{(m-1)})), x)$, where we select $\mathcal{L}$ to be L_1 loss. Then, the parameters of module m can be optimized using the greedy training objective:

$$\min_{\theta_m} \mathcal{L}(f_{\theta_m}(\mathbf{DC}(x^{(m-1)})), x) \tag{7}$$

where $x^{(m-1)} = f_{\theta^*_{m-1}}(\mathbf{DC}(x^{(m-2)}))$ for $m > 1$, and θ^*_{m-1} are the optimal weights for the previous module. Our method to optimize the greedy training objective in Eq. 7 for M modules is outlined in Alg 1. When a mini-batch is sampled during training, the first module computes the forward pass

R	Method	SSIM	nRMSE	PSNR (dB)	Memory (MB)
12x	Backpropagation	0.896 (0.006)	0.127 (0.007)	40.06 (0.33)	10016
	Greedy ($M = 8$)	**0.919 (0.001)**	**0.124 (0.008)**	**40.24 (0.36)**	**1679**
	Greedy ($M = 4$)	0.903 (0.004)	0.126 (0.007)	40.12 (0.33)	2603
16x	Backpropagation	0.887 (0.007)	0.134 (0.008)	39.61 (0.33)	10016
	Greedy ($M = 8$)	**0.912 (0.001)**	**0.131 (0.008)**	**39.79 (0.36)**	**1679**
	Greedy ($M = 4$)	0.888 (0.007)	0.134 (0.008)	39.60 (0.32)	2603

Table 1: Comparison of performance and maximum GPU memory during training for backpropagation and greedy learning with $M = 4$, and $M = 8$. Metrics are reported as *mean (standard deviation).*

on the mini-batch of images, performs a gradient update based on its own local loss, and passes the result of the forward pass to the next module. This procedure is repeated until the final module which produces the final prediction of the network, after which a new mini-batch is sampled. Therefore, individual updates of each set of parameters are performed independently across different modules, which drastically reduces the memory requirement. In our approach, we update each module using backpropagation. The model-based network trained with the proposed greedy approach is illustrated in Fig. 1.

4 Results

We evaluated the effectiveness of our approach for two acceleration factors, $R = \{12, 16\}$. We considered greedy learning with two different network splits $M = \{4, 8\}$. We reported structural similarity (SSIM), normalized root-mean-square-error (nRMSE), and peak signal-to-noise ratio (PSNR) calculated on magnitude images. Additional experimental details are provided in Appendix A.

Table 1 shows the performance and maximum GPU memory during training comparison for greedy learning and backpropagation. Greedy learning with $M = 8$ performed best across all image metrics compared with backpropagation, and greedy learning with $M = 4$. Representative knee images from the test set are illustrated in Fig. 2 in Appendix B. Greedy learning successfully recovered relevant structural information, preserving details at least as well as backpropagation.

We benchmarked the training speed and the maximum memory required during training on a 12GB NVIDIA Titan Xp graphics card. The maximum GPU memory used by greedy learning with $M = 8$ was 6x less than maximum GPU memory used in end-to-end backpropagation. On the same GPU, training time per-iteration was same for backpropagation and greedy learning (Fig. 3). Details on the training time comparison for backpropagation and greedy learning is provided in Appendix C.

5 Discussion and Conclusion

In this work, we presented an alternative to backpropagation for training model-based networks via greedy learning. By performing gradient updates on several network modules, greedy learning allows a large reduction in the memory-footprint of training. Our experiments on a knee MRI dataset show the effectiveness of our approach in reducing memory during training while preserving, and potentially improving, generalization performance.

A stated limitation of greedy learning is the decoupling of gradient updates, which prevents information flow between learned features among network modules that are optimized independently, resulting in less generalizable features. In our experiments for this application, however, we found that greedy learning with $M = 8$ generalizes better compared to $M = 4$ and backpropagation, in addition to great improvements to memory consumption. These memory improvements will undoubtedly be important for enabling larger model-based networks with higher expressivity for reconstructing images from 3D MRI, DCE imaging, or 4D-flow. The demonstrated benefits of greedy learning thus provide a vast potential for enhanced clinical outcomes, as can be investigated in future work.

Acknowledgments and Disclosure of Funding

This work was supported by NIH Grants R01 EB009690, R01 EB026136 and NSF Grant DGE-1656518.

References

[1] Klaas P. Pruessmann, Markus Weiger, Markus B. Scheidegger, and Peter Boesiger. Sense: Sensitivity encoding for fast mri. *Magnetic Resonance in Medicine*, 42(5):952–962, 1999.

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

[3] Christopher M Sandino, Joseph Y Cheng, Feiyu Chen, Morteza Mardani, John M Pauly, and Shreyas S Vasanawala. Compressed sensing: From research to clinical practice with deep neural networks: Shortening scan times for magnetic resonance imaging. *IEEE Signal Processing Magazine*, 37(1):117–127, 2020.

[4] Hemant K. Aggarwal, Merry P. Mani, and Mathews Jacob. Modl: Model-based deep learning architecture for inverse problems. *IEEE Transactions on Medical Imaging*, 38(2):394–405, Feb 2019.

[5] Morteza Mardani, Qingyun Sun, Shreyas Vasawanala, Vardan Papyan, Hatef Monajemi, John M. Pauly, and David L. Donoho. Neural proximal gradient descent for compressive imaging. *in Proc. Neural Information Processing Systems (NeurIPS)*, 2018.

[6] Patrick Putzky and Max Welling. Invert to learn to invert, 2019.

[7] Michael Kellman, Kevin Zhang, Eric Markley, Jon Tamir, Emrah Bostan, Michael Lustig, and Laura Waller. Memory-efficient learning for large-scale computational imaging. *IEEE Transactions on Computational Imaging*, 6:1403–1414, 2020.

[8] Ke Wang, Michael Kellman, Christopher M. Sandino, Kevin Zhang, Shreyas S. Vasanawala, Jonathan I. Tamir, Stella X. Yu, and Michael Lustig. Memory-efficient learning for high-dimensional mri reconstruction, 2021.

[9] Jiaming Liu, Yu Sun, Weijie Gan, Xiaojian Xu, Brendt Wohlberg, and Ulugbek S. Kamilov. Sgd-net: Efficient model-based deep learning with theoretical guarantees, 2021.

[10] Zhouyuan Huo, Bin Gu, Qian Yang, and Heng Huang. Decoupled parallel backpropagation with convergence guarantee, 2018.

[11] Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. Greedy layerwise learning can scale to imagenet, 2019.

[12] Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. Decoupled greedy learning of cnns, 2020.

[13] F Ong, S Amin, S Vasanawala, and M Lustig. Mridata. org: An open archive for sharing mri raw data. In *Proc. Intl. Soc. Mag. Reson. Med*, volume 26, 2018.

[14] F Ong and M Lustig. Sigpy: a python package for high performance iterative reconstruction. In *Proceedings of the ISMRM 27th Annual Meeting, Montreal, Quebec, Canada*, volume 4819, 2019.

[15] Leslie Ying and Jinhua Sheng. Joint image reconstruction and sensitivity estimation in sense (jsense). *Magnetic Resonance in Medicine*, 57(6):1196–1202, 2007.

[16] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

[17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

A Experimental Details

A.1 Dataset

Our experiments were performed with the fully-sampled 3D fast-spin echo (FSE) multi-coil knee MRI dataset publicly available in mridata.org [13]. Each 3D volume had a matrix size of $320 \times 320 \times 256$, where training is performed 2D by treating each axial slice with size 320×256 as separate examples. The dataset consists of unique scans from 19 subjects, where 14 subjects (4480 slices) were used for training, 2 subjects (640 slices) were used for validation, and 3 subjects (960 slices) were used for testing. Sensitivity map estimation was performed in Sigpy [14] using JSENSE [15] with kernel width of 8 for each volume. Fully-sampled references were retrospectively undersampled using a 2D Poisson Disc undersampling mask.

A.2 Training Details

Experiments were conducted using Pytorch [16]. A modified version of the network in [3] was used as the baseline network architecture. Our network had $N = 8$ unrolled blocks, where each proximal block consisted of 2 residual blocks with 128 channels. Each residual block consisted of 2 ReLUs followed by convolutional layers with no normalization. A batch size of 4 was used in all experiments, which was the largest possible batch size to fit the baseline network to a 12 GB GPU. The baseline network, as well as each network module in greedy learning were trained with the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ [17]. All networks were trained for 80000 steps. At inference time, the model checkpoint that achieved the lowest validation nRMSE was selected for each method.

B Sample Reconstructions

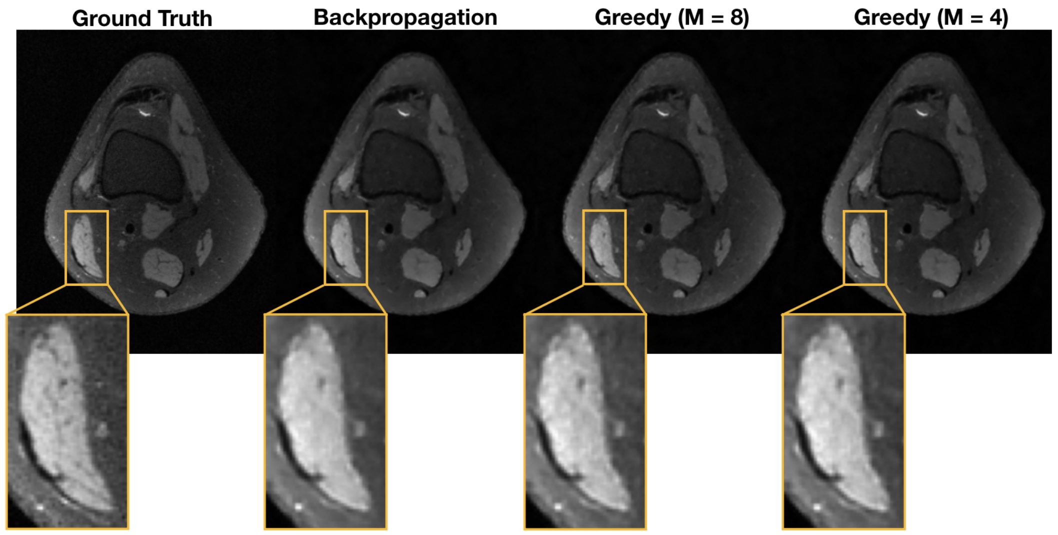

Figure 2: Reconstruction of a representative test knee slice with backpropagation, greedy learning with $M = 8$, and $M = 4$.

C Training Time

In this section, we report the computation time per-iteration across different methods. We compare backpropagation with greedy learning on the same GPU.

In addition, since individual updates of modules are independent, optimization of each module can be parallelized. In parallel greedy learning, the network modules are split across multiple devices. When a mini-batch is sampled, the forward pass is computed sequentially through all modules. After the forward pass is computed, each backward pass can be asynchronously executed among all modules in parallel. Since backward time is the primary bottleneck, asynchronous execution of the backward operation can greatly decrease overall computation time.

Computation time comparison for backpropagation on 1 GPU, greedy learning on 1 GPU, and parallel greedy learning on 2 GPUs are illustrated in Fig. 3. Since no additional computations are needed for greedy learning, the training time is same as backpropagation using the same GPU. Split across two GPUs, parallel greedy learning results in a speed up in training compared to backpropagation and greedy learning. When multiple GPUs are available, parallel greedy learning can enable layer-wise parallelism to reduce overall training time without compromising reconstruction performance.

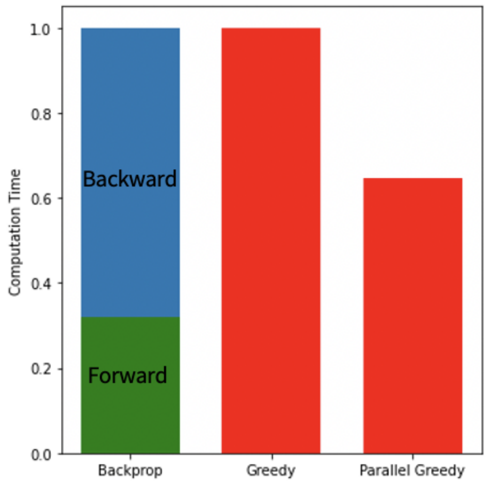

Figure 3: Comparison of computation time per iteration for backpropagation on 1 GPU, greedy learning on 1 GPU, and parallel greedy learning on 2 GPUs. Computation time for backpropagation was normalized to 1 to show relative speed across methods. On a single GPU, backpropagation and greedy learning have similar computation time. When independent modules in parallel greedy learning are split across 2 GPUs, backward time can be reduced to speed up training.