Explorations in Deep-Learning-Based MRI Reconstruction

Ian Liu, Ben Mackenzie, Lakshmi Sathidevi
Georgia Institute of Technology
iliu31@gatech.edu, bmackenzie3@gatech.edu, lsathidevi3@gatech.edu

Abstract

With the pace of innovation growing in the field of deep learning, improving on successful approaches depends on rapid experimentation and exploration. In that spirit, we look to improve the fastMRI baseline U-Net model through modifications of all three of this model's core components: architecture, loss function, and optimization method. A transformer inspired modification is attempted for the U-Net, the loss metrics NMSE, MSE, PSNR and SSIM are put to the test quantitatively to propose a new combined loss metric with emphasis on structural reconstruction, and the impact of the optimizer on training is assessed. Experimental results do not show revolutionary improvements, but they do point to promising avenues of future research.

1. Introduction/Background/Motivation

A key area of magnetic resonance imaging (MRI) research has been acceleration of image acquisition. While earlier approaches such as parallel imaging (PI) [5, 11, 14] and compressed sensing (CS) [8] have now become the standard in clinical use, solutions based on deep learning (DL) have the potential to significantly increase acceleration with minimal sacrifice in reconstruction quality.

Many of the recent DL-based MRI reconstruction methods, such as the U-Net [12] and VarNet [15], employ convolutional neural network (CNN) architectures, owing to their dominance in computer vision tasks. However, there has lately been increased interest in the use of transformer architectures for vision tasks, culminating in state-of-the-art results [4, 6]. This trend has now made its way into medical imaging research via transformer/U-Net hybrid models for medical image segmentation [3, 10].

We would like to expand on these findings to assess whether transformer-based architectures can improve the performance of the baseline U-Net architecture on the fastMRI dataset [17]. The motivation for this is to explore the impact of self-attention mechanisms within the context of MRI reconstruction. We hope that by adding long-range spatial context to U-Nets, we can achieve improvements that allow for higher quality reconstructions.

We also seek to address a problem highlighted in the 2020 fastMRI Challenge [9], in which radiologists were used for a qualitative evaluation of the scans. Researchers noticed that scans that scored well during quantitative evaluation were sometimes scored lower by radiologists because they depicted features that either mimicked normal structures that were not there or appeared abnormal even though no abnormalities were seen in reference scans. A potential solution that we explore here involves scoring scans differently in such a way that images are reconstructed more accurately, with fewer "false" features. If we are successful, we could narrow the gap between quantitative and qualitative metrics in reconstruction quality, increasing the reliability of quantitative measures.

Finally, we conduct a preliminary exploration of the optimizer AdaBound as an alternative to RMSProp, the optimizer used in the baseline fastMRI U-Net model. AdaBound developed out of the desire to create an optimizer as fast as adaptive optimization methods like RMSProp and Adam, but with the generalization power of Schotastic Gradient Descent [7]. The idea is to see if AdaBound performs similarly to RMSProp in early-stage training in order to motivate further exploration into AdaBound's potential to improve on RMSProp in later-stage training.

The fastMRI single-coil knee dataset was used for each exploration. Training was done on a fixed subset of the training set due to resource limitations, but the entire validation and test sets were used.

2. Approach

2.1. Improving the U-Net Architecture

In the architecture exploration, we first looked at how elements of transformers were being used to augment U-Nets. In Chen et al. [3], a number of transformer blocks were added to the output of the U-Net encoder, while Petit et al. [10] added multi-head cross-attention to the skip connections in addition to the multi-head self-attention after the encoder. We tested out the official implementations of both these architectures, but due to difficulties getting them to run, we used an unofficial PyTorch implementation [1] based on Chen et al.'s TransUNet.

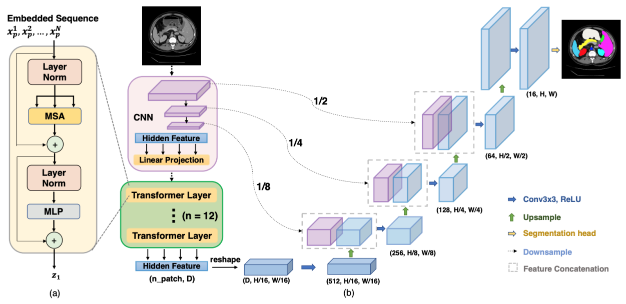


Figure 1. The TransUNet architecture from Chen et al. 2021.

The U-Net in the fastMRI baseline model was substituted for this TransUNet architecture, depicted in Fig. 1, which involved modifying the PyTorch Lightning modules and training scripts to accept a new model. By adding the transformer layers to the baseline U-Net, we hoped to demonstrate that the performance gains seen in the image segmentation studies could translate to MRI reconstruction, especially since the U-Net was originally proposed for image segmentation. To the best of our knowledge, ours is one of the first studies to apply transformer-augmented U-Nets to MRI reconstruction.

Two concerns we had prior to conducting the experiments were the fidelity of the unofficial implementation and the need to scale down our training due to constraints in compute resources. These concerns were indeed valid and are addressed in later sections.

2.2. Optimization: RMSProp vs AdaBound

In the optimizer exploration, we first trained the baseline fastMRI U-Net model [12] on a subset of the fastMRI single-coil dataset using RMSProp. One problem we anticipated and encountered was a lack of computing power, which resulted in long training times. In addition to training on a subset of the training set, we also trained the model for 5 epochs vs the default 50 in the fastMRI baseline U-Net model. We thought this would work because only a rough comparison of AdaBound and RMSProp in the early stages of training would be needed to validate future exploration into AdaBound as a way to improve on RMSProp in later-stage training.

2.3. Quantitative Evaluation of Loss Metrics

The aim of the exploration into loss metrics is to come up with something better than the commonly used L1 metric for training MRI reconstruction models. The baseline Unet model is used for training [12]. The only real assessment of the performance of metrics would be to visually (by sight) analyse them and give them scores, and this is attempted in a quantitative manner.

The flow followed for the quantitative assessment of performance of each loss metric is to first create a purely visual (by sight) evaluation of difference in image quality and then compare with that the same difference evaluated according to the loss metrics. The fundamental consideration behind this visual evaluation is that clarity, high level and low level similarity in structure are the main factors that contribute to clinical quality of MR images. Different pairs of MR images are ranked using each of the loss metrics and these rankings are compared with a visual ranking (given by manual observation). A score is then assigned for each metric so that then the metrics can be weighted using the scores.

Through experimentation it is observed that training with MSE or NMSE does not induce clarity of image. SSIM seems to slightly capture the clarity aspect by emphasizing the structural aspect but still even SSIM doesn't seem to really induce clarity. In this context, a new type of loss called Edge Loss [13] is attempted. The idea is to detect the edges of the space resolved ground truth MR image and to combine this with the training loss in such a way that, better learning of edges get enforced.

The clarity of images is something that still has room to improve significantly. So a further attempt is made to try different deblurring filters to find a suitable one that can enhance the subsampled space resolved MR image before it's fed to the U-Net.

3. Experiments and Results

3.1. TransUNet Performance

Both the baseline U-Net and the TransUNet were trained on two sets of data: a small portion of the single-coil knee training set (40 scans or around 4% of the data) and 70% (139 scans) of the single-coil knee validation set. Validation was performed on 12 scans (6%) of the validation set for the first training set and the remaining 30% (60 scans) of the validation set for the second training set. The reason for using part of the validation set as the training set in the second experiment was due to technical issues preventing transfer of the training set data to the cloud instance used for training.

In both these experiments, the same model hyperparameters were used. For the U-Net, five double convolution blocks with filter sizes doubling from 32 to 512 and max pooling layers in between made up the encoder. The decoder reversed this process with four transpose convolution blocks that produced an output with the same dimensions as the input. There were 7.76M total parameters in the model. By default, the RMSProp optimizer was used with a learning rate of 0.001. The model was trained with a batch size of one for 10 epochs.

The TransUNet had four convolution blocks in the encoder with double the filters (128 to 1024) as well as eight transformer blocks after the encoder. Each transformer block used four attention heads for MHSA and a linear layer of size 512. The decoder used four upsampling convolution blocks to produce the correct size output. The resulting model had 66.8M parameters total. The same optimizer, learning rate, batch size, and number of epochs were used.

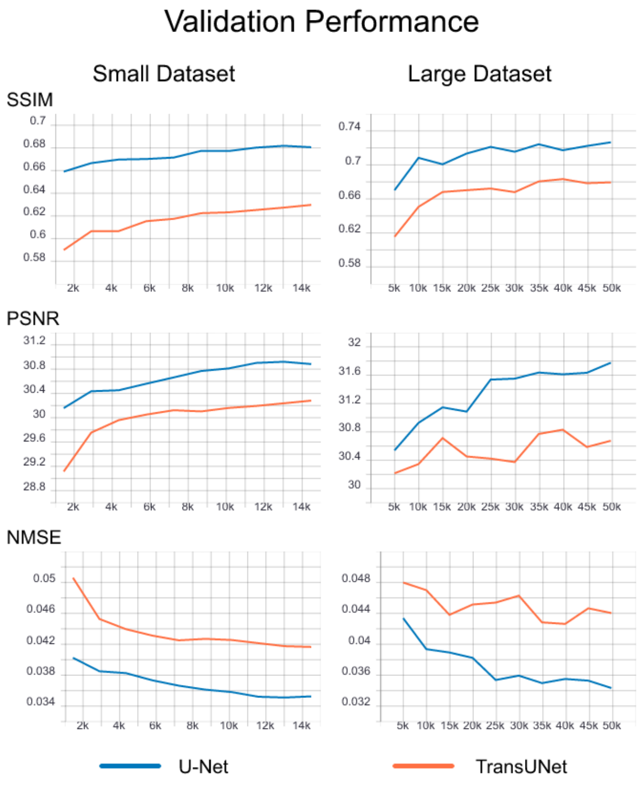


Figure 2. Validation performance of TransUNet compared to the baseline U-Net on small and large datasets.

The validation curves of both experiments are shown in Fig. 2, with SSIM, PSNR, and NMSE curves for the U-Net compared to TransUNet. We can see that TransUNet performed worse than U-Net on all metrics in both the small and large dataset experiments. In addition, TransUNet took nearly twice as long to train as U-Net: 31min vs 16min (small dataset) and 2hr8min vs 1hr13min (large dataset).

While these results were disappointing, there are several factors that could explain them and offer guidance on further exploration. As mentioned previously, we had concerns on whether the unofficial implementation of the TransUNet was completely faithful to the architecture proposed in the paper. Ideally, we would take the time to get the official implementation working or reproduce the architecture ourselves to be certain. In addition, computing resource limitations prevented us from training on the full datasets. Previous studies have shown that vision transformers require massive amounts of training data to achieve SOTA performance [4], so it is possible that TransUNet could have higher asymptotic performance given enough training data. This is further supported by the fact that the baseline U-Net was still 0.9 away from the 0.81 SSIM achieved in the original fastMRI paper [17].

Beyond these issues, there is still the question of how best to tailor transformer-based models to multi-coil datasets, which better represent the actual clinical data that exists. In fact, most clinical scans today are taken using PI and CS, which means that the majority of available training data will be subsampled multi-coil data. Given this, it would be worthwhile to explore the impact of attention mechanisms on multi-coil baselines like the VarNet [15], which operate on the raw k-space data. Furthermore, we originally wanted to explore self-supervised learning (SSL) techniques such as SSDU [16] but we were set back by technical challenges. Future studies in transformer-based techniques would lend themselves well to SSL paradigms given the auto-regressive nature in which transformers were originally trained as well as recent findings in self-supervised vision transformers [2].

3.2. Loss Metrics - A Quantitative Assessment

From the fastMRI dataset from FAIR [17], two sets with 11 slices of single-coil knee MRI data each, is taken (only the main slice in each volume is considered) - each slice is spilt into a pair, each pair containing the original fully sampled image and the corresponding subsampled image obtained by randomly masking it with an acceleration of 4 and center fraction of 0.08 (default masking provided in the fastMRI repository). These two sets can be observed in the Appendix in Fig. 11 and Fig. 12 respectively. The distance/similarity between each pair according to each loss metric - MSE, NMSE, PSNR and SSIM are printed next to each pair in the figures.

For comparing the loss metrics with visual assessment of quality, each pair of fullysampled and subsampled image is assessed to come up with a visual ranking (by sight) of quality difference - the more similar the perceived quality within a pair, the higher the ranking of that pair. Then the ranking for the pairs is taken with respect to each of the loss metrics - MSE, NMSE, PSNR and SSIM. Now the among the five different rankings we have obtained, the visual ranking is taken as the ground truth ranking. The rankings according to the loss metrics are then compared with the visual ranking by a custom algorithm (Algorithm 1) that assigns a score to each loss metric according to how similar the rankings they created are in comparison with the visual ranking - the greater the score, the more the similarity. This gives us a score value for each loss metric which helps us quantitatively compare the effectiveness of each metric in evaluating difference in quality of MR images. The exact scoring scheme for this method is developed over multiple iterations of trials to see what kind of a scoring would be

	SSIM	PSNR	MSE	NMSE
Set 1	7	3	3	9
Set 2	8	8	10	6
Set 1 + Set 2	-6	-12	1	0
Combined Score	12	5	13.5	15

Table 1. Ranking scores obtained by each loss metric

apt to allow room for the uncertainty involved in evaluating something like clinical quality and also at the same time provide enough penalty in the right way to distinguish the well performing and poorly performing metrics.

The whole process of ranking based on visual evaluation, evaluation by loss metrics and then scoring the loss metrics based on the similarity of the ranking they create with respect to the visual ranking, is done multiple times to improve the credibility of such a result. First it is done on Set 1, then on Set 2 and finally for further avoiding variations that could occur due to the small number of samples involved in the assessment, it is done on a third set which is nothing but a combination of all the pairs in Set 1 and Set 2. The scores obtained for each loss metric across each Set is summarised in Table 2. One interesting observation on the scores is that the highest scoring metric is also the one that is highly regarded for MRI reconstruction - NMSE [12].

For reproducibility, the rankings for the combined set with pairs from Set 1 and Set 2 are shown in Table 1 in the Appendix. The elements are in the form <pair number>_<set number>, for example element '10_2' means 10th pair from Set 2.

Finally, these scores for the loss metrics which is arrived at through quantitative analysis, is used to form a 'weighted loss' metric which is a weighted combination of MSE, NMSE, and SSIM based on their scores. As PSNR is seen to perform the weakest as per the scores (Table 1), it is not considered in the 'weighted loss'. The whole idea behind a weighted loss is that individual loss metrics maybe good at a certain kind of similarity like structural similarity while it may not be good at other kinds of similarity like pixel-wise similarity, so having a weighted loss enables the proper consideration of each of the 'abilities' or advantages of each loss metric. This weighted loss formulated from the scores comes out as,

$$loss_{train} = 13.5 * MSE + 15 * NMSE + 12 * SSIM$$

This 'weighted loss' is used to train the baseline U-Net. For the purpose of training and evaluating, a 20% subset of the fastMRI training set is used for 5 epochs, and the validation set is used as such. The network configuration

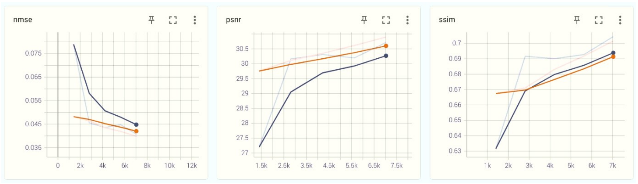


Figure 3. Performance of weighted loss with MC-SSIM

was with 64 channels and for training, learning rate was kept at 0.01. This was to shorten the training time without hindering proper observations. Also, the MRI dataset is seen to be a difficult challenge that it's hard to overfit with a smaller network like this one and the validation results later obtained is convincing so it was proceeded this way.

The weighted loss was first attempted in a normalised form where the weights for each loss metric was normalized and also, each metric was squashed linearly to have a value range between 0 and 1, to which the scores were multiplied as weights. Contrary to original expectation, this created a very poor result, it totally spoiled the reconstruction. This would be because the linear squishing to such a small range, affected the effective loss value that in turn is responsible for the gradient learning step. Removing this normalisation and just using the metrics as such, multiplied by the scores as weights showed a very slight improvement in validation SSIM scores.

It was further observed that having a multi-component SSIM (MC_SSIM) helped to give a further boost. The multi-component form of SSIM that was found to enhance the results is a sum of SSIMs of kernel sizes 6, 7, 9 and 11. This changed the training loss to,

$$loss_{train} = 13.5 * MSE + 15 * NMSE + 3 * MC\_SSIM$$

Here the weight of SSIM is reduced to 3 from earlier 12 because the MC_SSIM used varies between 0 to 4 while SSIM varies between 0 to 1. It is noted that training with SSIM of kernel sizes less than or equal to '5' exaggerates textural aspects of the MR image, this is understood as being due to the lower kernel size narrowing the area of attention of SSIM. The performance of the weighted loss with MC_SSIM when trained for 5 epochs with 20% of the training data is seen in Fig. 3. The orange line represents the baseline run while the violet line represent the result of weighted loss with MC_SSIM. The fainter version of these lines in the plot represent the actual values while the darker version represents the smoothed values. NMSE attained at the end of the 5 epochs on the validation data isn't better than before, it is very slightly poorer, this would be as there are a number of metrics at play and not just L1, like in case of the the baseline (orange).

The score for PSNR isn't taken with a similar importance as the quantitative analysis of the metrics show better corre-

Algorithm 1 Comparison of ranking by each loss metric to the visual ground truth ranking

```
1:  for $loss\_metric$ in [SSIM, MSE, PSNR, NMSE] do
2:      $score\_for\_metric$ = 0
3:      for each element in $metric\_ranking$ by $loss\_metric$ do
4:          Find index of same element, in $visual\_ranking$
5:          $distance$ = Difference in index of element between $metric\_ranking$ & $visual\_ranking.$
6:          if $distance$ <5 then
7:              $score\_for\_metric$ += 1
8:          else
9:              if $distance$ >7 then
10:                 $score\_for\_metric$ -= 2
11:             else
12:                 if $distance$ >4 then
13:                     $score\_for\_metric$ -= 1
14:                 end if
15:             end if
16:         end if
17:     end for
18:     Save $score\_for\_metric$ for $loss\_metric$
19: end for
```

lation with visual quality for NMSE and SSIM rather than PSNR. In case of SSIM scores on the validation set, we can see a slight improvement with respect to the baseline (Fig. 3) and observing the reconstructions visually, slight improvement in representation of the features of the original image in the reconstruction is observed (Fig. 4).

3.3. Pytorch Edge Detection and Edge Loss

The calculation of 'Edge Loss' is done by multiplying the binary edge map(obtained from edge detection) of the ground truth image with the L1 loss, whereby the loss along the edges alone gets extracted. The edge detection is carried out the classical way by applying a series of filters, first a Gaussian filter to smooth-out the noise (kernel size 3, sigma 1) followed by a Sobel filter (kernel size 3), the output of which was then 'thinned' using a thinning kernel (size 3) for obtaining a thin edge map. Edge map is taken both with and without thinning to obtain thick and thin edge maps (Appendix: Fig. **??**). In each case, the thresholds needed to be tuned for the obtaining an optimal edge map. It is seen that for different white levels in the MR image, a different thresholding is required (the fat suppressed MR images had lower while levels). To meet this thresholding need, the average pixel value is first taken, and it is checked if it falls in the lower, middle or higher category, and accordingly a specifically tuned thresholding is applied. The thresholding for each of these categories is tuned over iterations by visually observing which works and which doesn't. One significant bit about this edge detection is that it's implemented in Pytorch in a differentiable way and hence the loss gradient can be computed and back-propagated.

Initially, the training was attempted with edge loss using a thick edge map combined with weighted loss , and given the small pixel count available within the image, the edge map turned out to be too thick and the results were poor. The noted observation with using a thick edge map is that the network is in a way learning the sharp transitions in the image (area around edges) in a overly prioritised way that the reconstructions created by the network poorly represent the smooth transitions. The smooth transitions in the image were turning into sharp transitions upon reconstruction. For example a gradual texture change becomes like a sharp texture change in the reconstructed image. To tackle this the thinning was applied and the thin edge map created was used in the edge loss instead of the thick one.

The Edge Loss with thin edge map was used in combination with weighted loss for the training. The optimal weightage for Edge Loss is observed and found out manually over multiple trials.

$$\text{Edge\_Loss} = \text{Binary\_Edge\_Map}_{thin} * (Target - Output)$$
$$\text{loss}_{train} = Weighted\_Loss + 200 * Edge\_Loss$$

In Fig. 6 we get to see the results on applying the above loss. There isn't much difference in PSNR or NMSE but we we can note the slight increase in SSIM.

Edge Loss with thin edge map, though it is a solution for the issues posed when using the thick edge loss, the thickness was quite low, literally a pixel in thickness. Considering that this could be a limitation, a way of having an optimally thick edge map was devised. The idea is to upsample the ground truth image (bicubic) and then apply

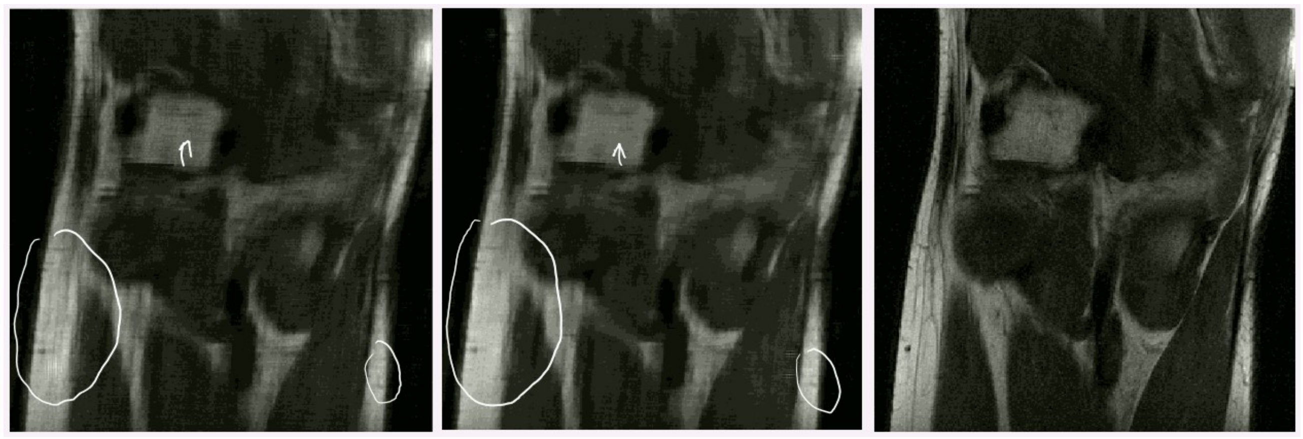


Figure 4. Visual signs of improvement with 5 epochs on 20% training data.
Left: Reconstruction with weighted loss with MC_SSIM; Middle: Baseline; Right: Ground truth

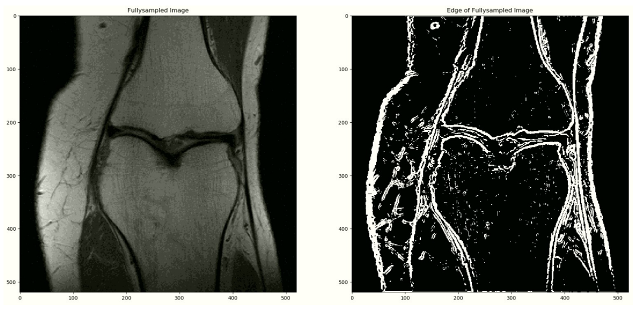


Figure 5. Thick edge map on resized image for Edge Loss

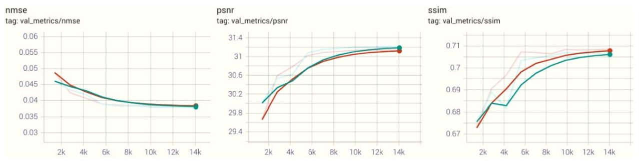


Figure 6. Result using thin edge map for Edge Loss

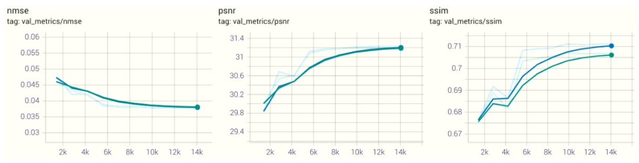


Figure 7. Result of applying thick edge map after upscaling

the edge detection such that effectively the thickness of the edge map is optimal (Fig. 5). The Edge Loss with thick edge map on resized ground truth (to 520x520 from 320x320) was used in combination with weighted loss for the training. The optimal weightage for Edge Loss is observed and found out manually over multiple trials.

$$\text{Edge\_Loss} = \text{Upscaled\_Binary\_Edge\_Map}_{thick} * (Target - Output)$$
$$\text{loss}_{train} = Weighted\_Loss + 100 * Edge\_Loss$$

In Fig. 7, we can observe the benefit of applying the upscaling and then taking the thick binary edge map. Compared to the thin edge map case (Fig. 6) we can see the NMSE and PSNR values are closer and more importantly, there is a greater increase in SSIM scores. An increase in SSIM score by 0.05 with respect to the baseline at the end of 10 epoch on 20% of the fastMRI training is noted. Also, this improved score is seen consistent and slightly increasing over the epochs (Fig. 7) which would be a good indication of better performance on longer training sessions.

3.4. Deblurring Filter Trials

Here on a slice from the dataset, different filters are tried out. A deconvolution as a solution of normal equations,

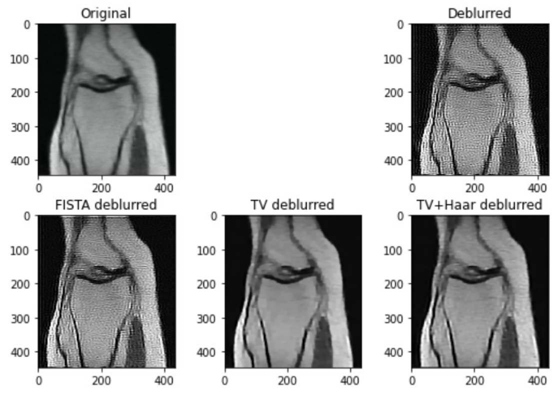


Figure 8. Deblurring Filters on the subsampled image

FISTA deblur, Total Variation (TV) deblur and Total Variation plus Haar deblur were tried (Fig. 8). For all filters, the 'pylops' library in python is used. It is observed that the filters other that Total Variation, causes severe artifacts in the deblurring process. Between TV and TV+Haar, TV+Haar is seen to provide a bit more of clarity, however, the restricting

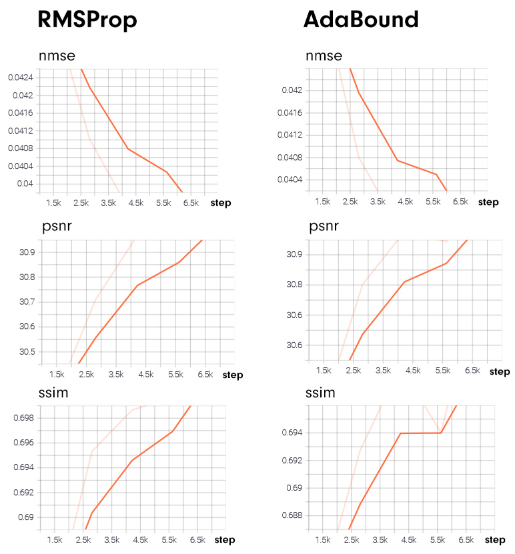


Figure 9. NMSE, PSNR, and SSIM curves for the RMSProp and AdaBound optimizers.

issue faced was that despite the perceived higher clarity, the image extracted this way is seen to suffer from a 'pixelated' sort of effect. This would be because of the low resolution of the MR images upon which the kernel of the deblur algorithm is running, so upscaling of the images is attempted before applying the filters, but still it did not solve the issue.

Different parameters and different kernel sizes for each of the deblur algorithms were also attempted but the processed image could not qualify to be good enough on visual examination, so the plan to use filters was then dropped.

3.5. RMSProp vs AdaBound: Validation Metrics

Observingthe Fig. 9, NMSE, PSNR, and SSIM curves for RMSProp and AdaBound seem just slightly different. the two optimization methods yield almost the same values at the end of roughly 7,000 steps of the 5 training epochs but if we take a closer look at the fainter lines (the true value of the metric without smoothing) in the figure, we can see that the rate at which the value drops or rises is slightly higher in case of AdaBound. This is in lines with the targets of AdaBound in that it is aimed at being as fast as ADAM and as good as SGD.

This suggests that AdaBound can indeed be faster than similarly adaptive optimization methods like RMSProp in our baseline U-Net model and justifies a follow up exploration of AdaBound's performance against RMSProp in later-stage training.

4. Conclusion

In this study, explorations pertaining to three major aspects of the U-Net and its training were completed:

- The architecture: a transformer-inspired modification was attempted and evaluated.
- The loss function: the loss metrics were explored and the new training loss using a combination of weighted loss with multi-component SSIM and Edge Loss is proposed.
- The optimizer: the impact of the optimizer on the training of the network for better reconstructions was assessed.

Though our experimental results do not show revolutionary improvements, they do point to promising avenues of future research.

The transformer-augmented U-Net we explored did not produce better results than the baseline, but there are still many contexts in which transformers may benefit MRI reconstruction including self-supervised learning. More research with larger datasets are needed to conclude that convolutional networks remain superior.

The following training loss is proposed as per the experiments conducted:

$$\text{loss}_{train} = 13.5MSE + 15NMSE + 3MC\_SSIM + 100Edge\_Loss$$

Here, MC_SSIM is the sum of SSIMs of kernel sizes 6,7,9 and 11, and Edge_Loss is the edge loss calculated using thick edge map on the spatially resolved ground truth MR image resized to 520x520 from 320x320. This loss achieves an SSIM score improvement of 0.05 over 10 epoch on 20% of the fastMRI training data. A very slight improvement is observed in the reconstructions as well.

The change in optimizer does create a difference but it did not create a significant change in the final validation scores. The reason could be the very short training span that was possible. Still, the difference in speed could be observed and finally, additional comparisons of optimizers beyond RMSProp are warranted.

5. Work Division

The expansive task of exploring the possible ways of improving each of the three main aspects of model and its training is divided among the three members of the team. The aspects in consideration are: the architecture, the loss and the optimizer. Ian Liu handled the exploration and comparison of a transformer-inspired architectural modification for the baseline U-Net. Lakshmi S. handled the exploration of loss metrics and of image enhancement before training. Ben Mackenzie handled the exploration into the impact of the optimizer on the learning. This is neatly explained in contribution table.

References

[1] Nikolas Adaloglou. Transformers in computer vision. *https://theaisummer.com/*, 2021. 2

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers, 2021. 3

[3] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L. Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation, 2021. 1

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020. 1, 3

[5] Mark Griswold, Peter Jakob, Robin Heidemann, Mathias Nittka, Vladimir Jellus, Jianmin Wang, Berthold Kiefer, and Axel Haase. Generalized autocalibrating partially parallel acquisitions (grappa). *Magnetic Resonance in Medicine*, 47:1202–10, 06 2002. 1

[6] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021. 1

[7] Liangchen Luo, Yuanhao Xiong, Yan Liu, and Xu Sun. Adaptive gradient methods with dynamic bound of learning rate. May 2019. 1

[8] Michael Lustig, David Donoho, and John Pauly. Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic Resonance in Medicine*, 58:1182–95, 12 2007. 1

[9] Matthew J. Muckley, Bruno Riemenschneider, Alireza Radmanesh, Sunwoo Kim, Geunu Jeong, Jingyu Ko, Yohan Jun, Hyungseob Shin, Dosik Hwang, Mahmoud Mostapha, and et al. Results of the 2020 fastmri challenge for machine learning mr image reconstruction. *IEEE Transactions on Medical Imaging*, page 1–1, 2021. 1

[10] Olivier Petit, Nicolas Thome, Clément Rambour, and Luc Soler. U-net transformer: Self and cross attention for medical image segmentation, 2021. 1

[11] Klaas Pruessmann, Markus Weiger, Milan Scheidegger, and Peter Boesiger. Sense: sensitivity encoding for fast mri. *Magnetic Resonance in Medicine*, 42:952–62, 12 1999. 1

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015. 1, 2, 4

[13] George Seif and Dimitrios Androutsos. Edge-based loss function for single image super-resolution. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1468–1472, 2018. 2

[14] Daniel Sodickson and Warren Manning. Simultaneous acquisition of spatial harmonics (smash): Fast imaging with radiofrequency coil arrays. *Magnetic Resonance in Medicine*, 38:591–603, 10 1997. 1

[15] Anuroop Sriram, Jure Zbontar, Tullie Murrell, Aaron Defazio, C. Lawrence Zitnick, Nafissa Yakubova, Florian Knoll, and Patricia Johnson. End-to-end variational networks for accelerated mri reconstruction, 2020. 1, 3

[16] Burhaneddin Yaman, Seyed Amir Hossein Hosseini, Steen Moeller, Jutta Ellermann, Kamil Ugurbil, and Mehmet Akcakaya. Self-Supervised Learning of Physics-Guided Reconstruction Neural Networks without Fully-Sampled Reference Data. *Magnetic Resonance in Medicine*, 84(6):3172–3191, Dec 2020. 3

[17] Jure Zbontar, Florian Knoll, Anuroop Sriram, Tullie Murrell, Zhengnan Huang, Matthew J. Muckley, Aaron Defazio, Ruben Stern, Patricia Johnson, Mary Bruno, Marc Parente, Krzysztof J. Geras, Joe Katsnelson, Hersh Chandarana, Zizhao Zhang, Michal Drozdzal, Adriana Romero, Michael Rabbat, Pascal Vincent, Nafissa Yakubova, James Pinkerton, Duo Wang, Erich Owens, C. Lawrence Zitnick, Michael P. Recht, Daniel K. Sodickson, and Yvonne W. Lui. fastMRI: An open dataset and benchmarks for accelerated MRI. 2018. 1, 3

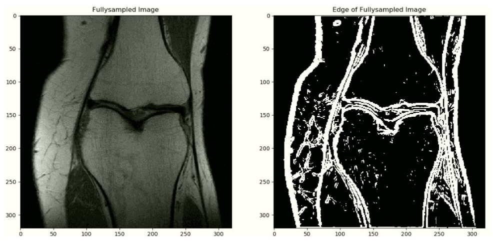
6. Appendix

6.1. Code Repositories

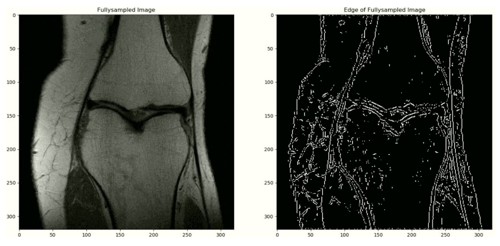
Architecture exploration: Google Docs

Analysis of loss metrics & Edge Loss Pytorch implementation: Google Docs

6.2. Figures



(a) Thick Edge Map



(b) Thin Edge Map

Figure 10. Thick vs Thin binary edge maps for Edge Loss

Student Name	Contributed Aspects	Details
Ian Liu	Architecture experiments & analysis	Assessed SSDU, TransUNet, and U-Net-Transformer codebases. Adapted fastMRI U-Net code for TransUNet experiments. Wrote paper sections related to architecture exploration.
Ben Mackenzie	Optimizer experiment & analysis	Experimented with the effect of the optimizer on training performance. RMSProp is tested against AdaBound. Wrote paper sections on optimizer exploration.
Lakshmi S.	Loss Experiments, Evaluation & Deblur Trials	Experimentation, quantitative analysis and formulation of weighted loss based on NMSE, multi-component SSIM, MSE and Edge Loss. It's evaluation and pre training image enhancement trials.

Table 2. Contributions of team members.

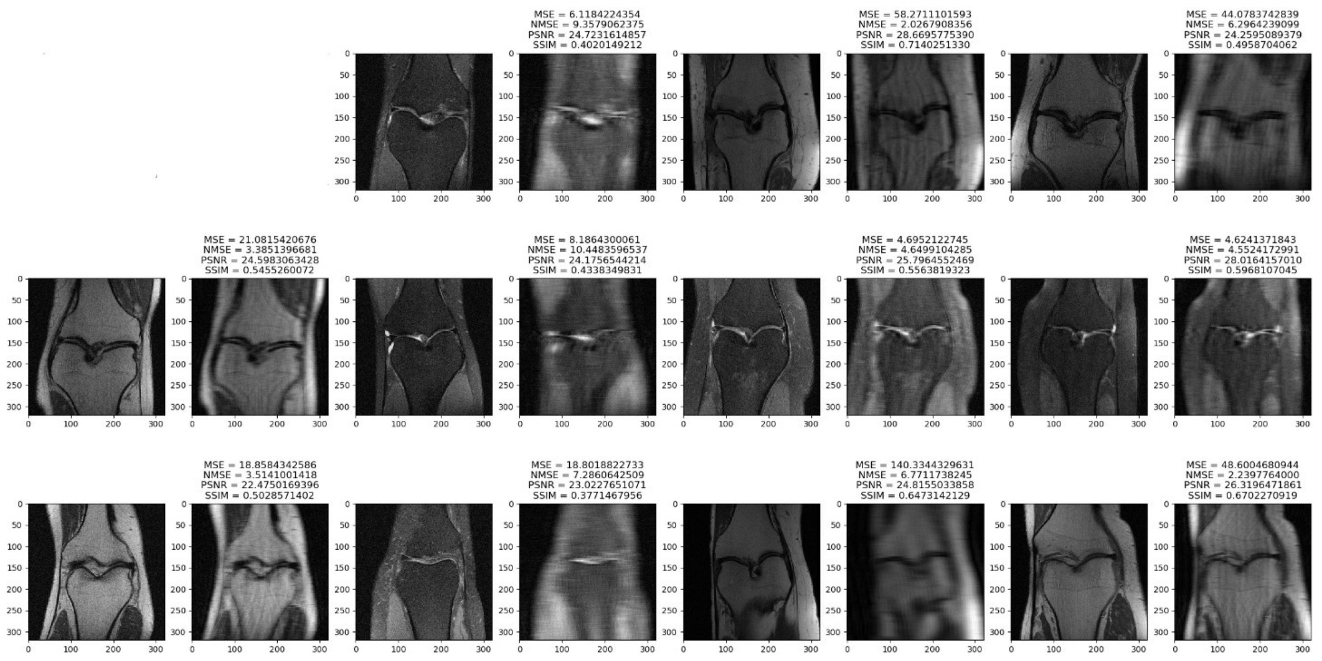


Figure 11. The 11 pairs of Set 1

Rank	Visual	SSIM	MSE	PSNR	NMSE
1	12_1	3_1	8_1	3_1	3_1
2	5_1	12_1	7_2	8_1	12_1
3	7_2	1_2	7_1	10_2	1_2
4	8_1	11_1	2_1	7_2	5_1
5	2_2	10_2	6_2	1_2	9_1
6	7_1	8_1	6_1	12_1	7_2
7	9_1	6_2	4_2	6_2	10_2
8	6_2	7_2	2_2	7_1	6_2
9	4_2	2_2	10_1	2_2	8_1
10	1_2	7_1	9_1	11_1	7_1
11	3_1	5_2	10_2	2_1	9_2
12	10_2	5_1	5_1	5_1	8_2
13	9_2	8_2	9_2	4_1	5_2
14	10_1	9_1	4_1	6_1	2_2
15	8_2	4_1	11_2	4_2	4_1
16	6_1	4_2	12_1	10_1	3_2
17	11_2	3_2	1_2	9_2	11_1
18	5_2	9_2	3_1	9_1	10_1
19	4_1	6_1	3_2	4_2	11_2
20	11_1	2_1	8_2	8_2	2_1
21	3_2	10_1	5_2	3_2	4_2
22	2_1	11_2	11_1	11_2	6_1

Table 3. Ranking of the 22 pairs according to each loss metric vs their visual ranking

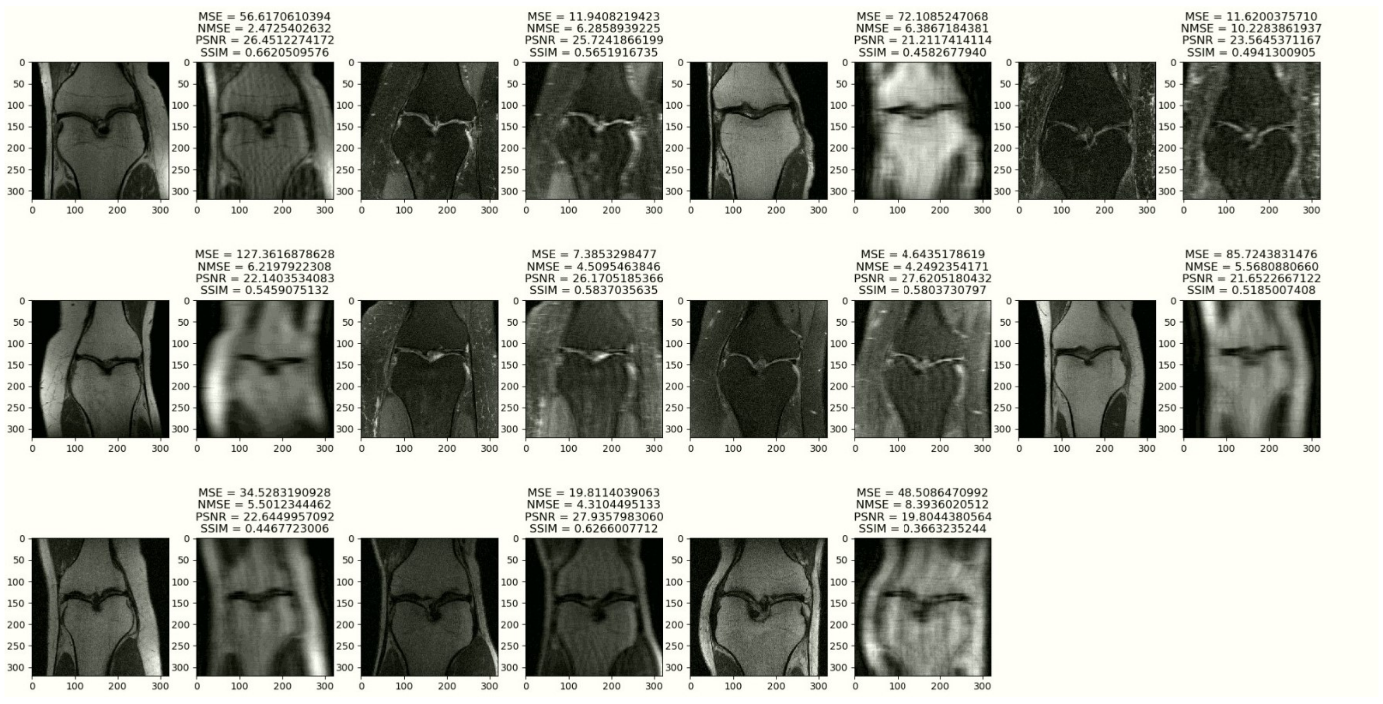


Figure 12. The 11 pairs of Set 2