Self-MRI

Claudiu Comsa
claudiu.comsa@gatech.edu

Yuriy Lichman
ylichman@gatech.edu

David Schmidlin
david.schmidlin@gatech.edu

Sergei Belenki
sbelenki3@gatech.edu

Georgia Institute of Technology
Atlanta, GA 30332, USA
May 4, 2021

Abstract

Magnetic resonance imaging (MRI) has long scan times which makes it both costly and prohibitive for some patients. By not sampling the entire spatial frequency both the scan time and the cost can be reduced. When scans are converted from frequency domain (k-space) to spatial domain, artifacts such as Shannon-Nyquist aliasing can appear due to this sub-sampling. Deep Neural Networks (DNN) can be used in a supervised setting scenario to perform an image-to-image reconstruction, from the sub-sampled scan to a full scan, but this would require ground-truth fully-sampled data which is expensive to obtain. Using self-supervised learning combined with transfer learning that this project is focusing on we were able to achieve 60% wall-time speedup for the image reconstruction process measured in total training time, comparing to the fastMRI benchmark, with matching performance on 4x-acceleration test data.

Keywords— *self-supervised learning, transfer learning, spatial domain, frequency domain*

1. Introduction/Background/Motivation

High-resolution MRI is an invaluable tool used by the medical community to diagnose various conditions. Acquiring high-resolution images is expensive and requires a long procedure that introduces additional patient discomfort. It is possible to decrease the cost and acquisition time by sub-sampling in k-space but this can lead to artifacts in the final image which reduces the usefulness of the image. Acquiring a large data set of sub-sample images along with their ground truth of fully sampled images has not yet been done and is a costly task to undertake. We attempt to use self-supervised learning (SSL) techniques along with U-Net deep learning techniques to train a U-Net style model to produce a high-resolution image from sub-sampled unlabeled data. This technique can reduce imaging time while maintaining image quality, thus allowing MRI to be used for new applications and with patients who are not able to maintain a stable position for the duration.

1.1. Data

The NYU fastMRI [7] single-coil knee dataset was obtained from the NYU Langone Health and includes 973 files in the training, 199 files in the validation, and 108 files in the test set. Each training and validation file is in the H5 format containing single-coil *kspace* [36, 640, 372] tensor emulated from the original multi-coil MRI k-space, *reconstruction_esc* [36, 320, 320] tensor with the ground truth image for single-coil k-space, and *reconstruction_rss* [36, 320, 320] tensor with the ground truth image for the original multi-coil k-space. The tensor dimensions are as following: [slice number, height, width]. Test files are in the same H5 format, but do not contain validation images. Each slice represents a capture of the same MRI scan for a particular moment in time, the number of slices varies from 33 to 37 per H5 file.

2. Method

The team chose to leverage the VISSL [5] framework from Facebook AI Research (FAIR) to perform the training of 1-channel U-Net or U-Net++ model using the SwAV, RotNET, SimCLR, and PIRL SSL tasks. Once a model had completed the pre-training (pretext task training), it was further fine-tuned using the fastMRI [10] framework, also from FAIR.

VISSL framework was designed around images, while fastMRI data set consisted of raw k-space data. To make VISSL work with fastMRI, we created an adapter that extracted images from selected slices of MRI h5 files. These

files were then fed into the existing VISSL image transformation pipeline.

Initially, the plan was to train a model end-to-end in the frequency domain, or k-space, of the MRI data. The team felt this would have been a novel approach to solving the problem but it was unclear what type of features the model would learn since there is a different data modality in the frequency domain. We were uncertain if convolution was even valid in that domain. Unfortunately, it was discovered that PyTorch does not support working with tensors of complex or imaginary values. There were at least 2 possible GitHub repositories that could add this functionality to PyTorch but the team felt given the time we had it was too large of a risk and we focused our efforts back on working in the spatial domain. At the same time, we discovered that the fastMRI library also works with the spatial domain (our initial assumption was that it was in the frequency domain).

Given that we needed to train models using data in the spatial domain and yet still apply a mask to a frequency domain image, to represent sub-sampling, 4 additional transforms were added to the VISSL code base. These new transforms were SpatialToFrequency, ApplyFrequencyMask, FrequenctyToSpatial and ToOneChannelTensor. They allowed the conversion between the frequency and spatial domains, the application of a mask to sub-sample a frequency image, and the ability to convert to a single-channel tensor. Most of the existing Image transformation in VISSL expected a 3-channel or RGB image, and yet the fastMRI library expects 1-channel, or grayscale, images. We intended to reuse as much of the existing infrastructure as possible. Thus, we first extracted images from the frequency domain in pseudo-RGB, which worked with VISSL. After all of the transformations were complete we converted them to single-channel images that matched how baseline U-Net was trained.

Our code with enhanced fastMRI and VISSL frameworks as well as additional script for running pre-trained and fine-tuned models can be found at this link: Self-MRI https://amzn.to/3vzjm2i

3. Experiments and Results

With two best configurations found in our experiments we were able to reach the same Structural Similarity Index (SSIM) [9] value as the U-Net baseline in a fraction of total (pre-training and fine-tuning) time - 37% speedup for RotNet/U-Net and 60% speedup for SimCLR/U-Net (see Table 1).

For PIRL, RotNet, and SimCLR a U-Net model was used and for SwAV a U-Net++ model was used. Both U-Net and U-Net++ were used with 1 input and 1 output channel and a varying number of top-level channels. For each SSL technique, the model was pre-trained for a set number of epochs with the VISSL framework and then fine-tuned using fastMRI on a smaller number of epochs. During the pre-training (PT) steps each technique had its loss method which was used to verify the models' convergence status. While fine-tuning (FT) the SSIM was used to measure training success. In some cases, a visual inspection of the models' output was done during the pre-training step to verify the model was working.

For each SSL method, there was a VISSL image augmentation pipeline and within each, the image was converted to the frequency domain a mask was applied and the image was then converted back to the spatial domain. To produce sub-sampling effect, a random mask with a center fraction of 0.08 and an acceleration of 4 was used for each SSL method.

3.1. PIRL

PIRL [8] stands for Pretext-Invariant Representation Learning and is a method for learning features that are invariant to transformations. It learns an image representation that is similar to a transformed image representation and different than representation for other images. The original representation size of a 128-dimensional vector was used. During pre-training the loss metric was a noise contrastive estimator [6] (NCE) loss with a memory size of 32000 and training was done using stochastic gradient descent with an initial learning rate of 0.0012 and an ending value of 0.0000012 was used. Several learning rates were used and a value of 0.0012 produced the best model output given the epochs used during pre-training. The authors of the paper used a ResNet model but here we are attempting PIRL with a U-Net model using 32 top-level feature channels. An experiment was run using 64 feature channels but the system ran out of memory.

During pre-training, it was observed that the model would output a white image with black splotches and the validation loss was not converging. This was thought to be due to the default inclusion of transforms in the VISSL pipeline that modified the image under the assumption that the image was a 3-channel RGB image. This assumption is not valid for this scenario as the MRI images are a 1-channel gray-scale. Removing the ColorJitter and ImgPilRandomPhotometric transforms from the VISSL transformation pipeline improved the output image and the validation loss began to converge. Once the frequency mask was applied the rest of the image transformation pipeline consisted of the creation of the image patches, a random horizontal flip, and a normalization step that produced a tensor with a mean of 0.5 and a standard deviation of 0.225.

For the pre-training (PT) phase PIRL was trained at 100, 200, and 500 epochs. As the number of pre-training epochs increased so did the visual quality of the image produced. This aligns with the original paper where several thousand

epochs were used. During the fine-tuning (FT) stage the model was trained for only 10 epochs. As the number of pre-training epochs increased so did the SSIM score of the model after 10 FT epochs. Further pre-training could have improved the model even more as evidence by the increasing SSIM trend as the number of PT epochs increased.

Even though in the experiments run PIRL performed worse than the baseline in terms of SSIM we think that this would improve if more pre-training was done. Given that the model produces images that are visually identifiable after just pre-training it was initially thought that PIRL would be a viable replacement for full training. But given that the length of time required for pre-training and fine-tuning, as shown in Table 1, is close to the baseline time and that the SSIM score is still lower than the baseline it seems that PIRL is not a viable alternative to full training. It is hypothesized that the additional training that PIRL requires, as compared to the other SSL methods, is due to the latent 128d vector space used and the NCE losses inherent requirement for many negative samples.

3.2. RotNet

RotNet, introduced by Gidaris *et al*. [4] is one of the simplest Self-Supervised tasks that rely on basic geometric transformations to extract high-level semantic features. The transformations are four image rotations of [0, 90, 180, 270] degrees. The underlying network is given a rotated image and needs to predict which one of the rotations are applied. Although the SSL performed basic transformations, the authors have shown state-of-the-art performance on various benchmarks.

A few modifications to the default configuration of RotNet (that presumably worked well for the ImageNet dataset) had to be made. The SGD optimizer with the multi-step auto-learning rate (LR) scaling was too aggressive and the run-time crashed with infinite loss, which indicated divergence. After several iterations, the best combination proved to be an auto-scaled LR that decreased from 0.0012 to 0.00012 gradually for each iteration. In addition, an ADAM optimizer was used instead of SGD. The loss function was left with a customary Cross-Entropy Loss. On top of rotations, the other modifications included applying a random mask and conversion to a single-channel tensor from a 3-channels input image. The SSL worked with a U-Net learner with 64 feature channels.

For the pre-training (PT) phase, RotNet's training and validation curves indicated good performance as both approached zero loss within 200 epochs: see Table 1. However, feeding an image into the pre-trained network before fine-tuning steps produced a blank image. On further examination, the feature map indicated an area of interest in the shape of a knee, as expected: see Figure 3 (c).

The output of RotNet was then fine-tuned for 10 epochs using the existing fastMRI framework. The network immediately produced almost as high values of SSIM as compared to the baseline that was trained for 50 epochs. The total wall-clock time for pre-training and fine-tuning was just under 6 hours, while the baseline ran over 9 hrs to produce similar SSIM results. The produced images were visually blurry but would likely improve further with more fine-tuning and additional epochs: see Figure 3 (d).

3.3. SimCLR

SimCLR [2] is a simple contrastive learning framework that aims to simplify various self-supervised representation learning methods on images. Through contrastive learning, the framework tries to learn representations by maximizing agreement in a positive pair (the same image transformed with two different transformations), and at the same time minimize agreement in a negative pair (two different images transformed). This contrastive loss objective leads to representations of corresponding views to "attract" each other, while the representation of non-corresponding views to "repel" each other.

The pipeline has been set up in such a way that each image in the dataset is transformed through random cropping, random horizontal flip, color distortion, and Gaussian blur. SimCLR will then learn representations on these positive and negative pairs using a U-Net CNN architecture, followed by a fully connected layer at the end to maximize the ability to identify different transformations of the same image. Both the CNN and the fully connected network are optimized with stochastic gradient descent.

In the pre-training phase, the SimCLR model was trained for 50, 100, and then 200 epochs. Different sizes for the random crop were experimented with (64, 112, 224, 320), finally settling on the original 224 which was used in the SimCLR paper. After the fine-tuning phase, it was observed that the similarity index (SSIM) achieved with the SimCLR pre-trained model is slightly better than the baseline model, starting with as few as 5 epochs of fine-tuning. The improvement over the baseline also comes in the form of reduced total time (pre-training and fine-tuning) from more than 9 hours for baseline to less than 4 hours for SimCLR. The reconstructed image is shown in Figure 3 (j), where it can be compared to the baseline reconstruction (k).

3.4. SwAV

SwAV, or Swapping Assignments Between Views is a self-supervised contrastive cluster assignment learning method developed by Caron *et al*. [1] from Facebook Research (FAIR) that doesn't require computationally intractable pairwise comparisons of augmented images used in other SSL. SwAV introduces soft clustering assignment based on Sinkhorn-Knopp algorithm [3] that may be used in the online contrastive learning setting, and multi-crop

augmentation when the same image cropped randomly into high resolution and low-resolution views that make the model trained on these views scale-invariant. Other standard augmentations that were used in our experiments in the SwAV pipeline were random horizontal flip with probability 0.5 and Gaussian blur with radius 0.5.

The particularity of our experiment with SwAV was the integration of U-Net++ [11, 12] that, compared to a regular U-Net, is considered more advanced architecture with redesigned skip-connections and deep supervision (Fig 1 in [11]). According to the authors SwAV was expected to take less computational resources than other SSL methods, and so we were hoping to get better pre-training results with U-Net++, while in our opinion it was not practical to train U-Net++ on our hardware using other, heavier SSL methods (like SimCLR). In the end, this combination of SwAV and U-Net++ produced very interesting results that we discuss later in this subsection 2. In our experiments, the U-Net++ was configured to run with 1 input and 1 output channel (for grayscale image), and we also ran several tries with a different number of top-level U-Net channels - 32, 64 - to explore the trend in the pre-training performance. Practically, on our hardware, we were able to run up to 64 top-level U-Net channels and have noticed that a higher number of the channels was corresponding with the better pre-training performance.

For the SwAV algorithm, we experimented with the following hyper-parameters (HPs): multi-crop sizes of 16, 32, 64, 80, 96, 224, 320 pixels, different number of crops in the multi-crop (between 2 and 8), and data batch sizes up to 64 (to be able to fit into the GPU memory.) The VISSL model head was adjusted accordingly with the used multi-crop sizes. In the end, we ran SwAV PT with U-Net++ for 100 epochs with the batch and crop sizes of 64, and 2 and 6 multi-crop sets.

Pre-training dynamics of SwAV was very interesting: its loss (Sinkhorn-Knopp loss) felt very quickly in the first 15-20 minutes of PT and after that stayed stable despite our attempts to reduce it further by configuring batch sizes and learning rate (we also tried to use Adam optimizer instead of the default VISSL RMSProp optimizer). We hypothesize that this is probably because our data batch size was not enough for this SSL, and we were unable to run with larger batch sizes because of available GPU memory limitations. One problem that was discovered during FT and that requires additional experimentation is that SwAV didn't respond well to our FT procedure (see Figure 2). We were trying to adjust the initial learning rate and schedule parameters, but probably because of the distinguished local optimum found by SwAV during PT its FT learning was poor and requires additional research. Overall, we believe that the outstanding ability of SwAV for fast learning during pre-training is very promising.

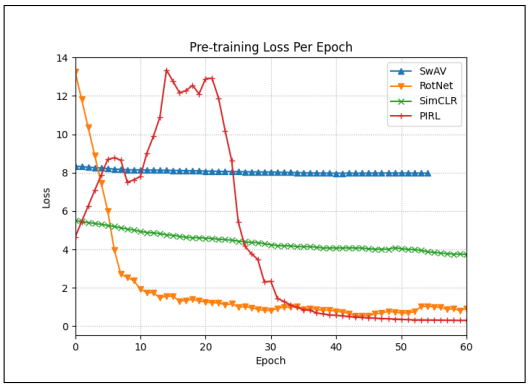

Figure 1. Pre-training loss per epoch.
NOTE: the loss is a SSL method-specific measure, and shown here to reflect SSL training dynamics (see Section 3 for the details.), PIRL loss is scaled by a factor of 160.

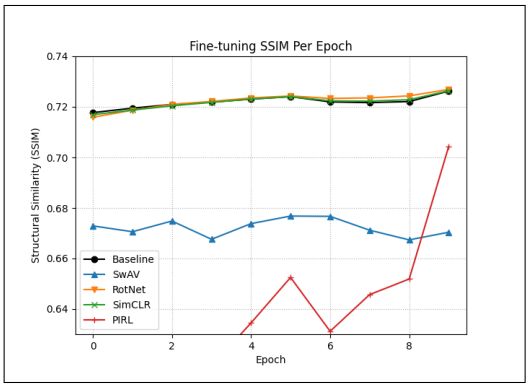

Figure 2. Structural Similarity (SSIM) for Fine-tuning process.

Figure 3 shows reconstruction results for U-Net++ after SwAV SSL pre-training on image (e), and fine-tuning on image (f).

3.5. Pre-Training and Fine Tuning

Our baseline U-Net model architecture was taken from the Facebook Research fastMRI repository at https://github.com/facebookresearch/fastMRI and trained for 50 epochs with the default parameters for 4x-acceleration specified in the fastMRI paper [7]. On the 4-GPU AWS instance (see 5), it took more than 9 hours to train (see the achieved results in the Table 1).
For pre-training, we integrated the new U-Net and U-Net++ modules, configurations, and transforms into the VISSL framework taken from the Facebook Research VISSL repository https://github.com/facebookresearch/vissl.
For fine-tuning we created scripts permitting us to load VISSL pre-trained U-Net and U-Net++ models, and train them in a manner comparable with the baseline training procedure, to receive interpretable results for our final analysis.

For details on parameters used during models pre-training and fine-tuning see related SSL model subsections (PIRL 3.1, RotNet 3.2, SimCLR 3.3, and SwAV 3.4).
In Table 1 you can find PT and FT results for each of the SSL methods (all wall-time values are in hours.) For examples of image reconstruction for inferences after PT, FT, see Figure 3.
NOTE: Two upper images were reconstructed manually from the training dataset to illustrate training input (sub-sampled at 4x-acceleration) and training target (extracted from the *reconstruction_esc* slice). All other images were produced using models pre-trained with SSL as well as fine-tuned models.

3.6. Testing

The test set provided by fastMRI was not acceptable for our testing as it contained sub-sampled k-space slices only, and no corresponding ground truth images for calculating SSIM, PSNR, and NMSE reconstruction metrics. To overcome this obstacle we ran our testing on the portion of validation part of the dataset that was not used in the pre-training or fine-tuning processes. The fully sampled k-space slices were converted into sub-sampled using 4x-acceleration mask with center fraction of 0.08 and, after tests executed, reconstructed imaged compared with their ground truth (*reconstruction_esc* slice images in the files).

4. Further Research

1. Colorize reconstruction results: train network to create 3-channel spatial reconstruction based on 1-channel k-space image. This will require that PyTorch support complex data types.
2. Continue to increase the number of pre-training epochs for PIRL to determine where it stops improving in regards to fine-tuning SSIM scores.
3. Super-fast pre-training with SwAV: we have found that the bulk of the learning during PT phase for SwAV was happening during the first 15 minutes (reaching estimated SSIM value in the range 50 for reconstruction without FT)

5. Environment

Our experiments were run on AWS G4 instances, which were powered by 4 NVIDIA T4 Tensor Core GPUs, 48 vCPUs and 192 GiB of RAM. The instances had Amazon Linux Operating System.

References

[1] Mathilde Caron et al. *Unsupervised Learning of Visual Features by Contrasting Cluster Assignments*. 2021. arXiv: 2006.09882 [cs.CV].

[2] Ting Chen et al. *A Simple Framework for Contrastive Learning of Visual Representations*. 2020. arXiv: 2002.05709 [cs.LG].

[3] Marco Cuturi. *Sinkhorn Distances: Lightspeed Computation of Optimal Transportation Distances*. 2013. arXiv: 1306.0895 [stat.ML].

[4] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. "Unsupervised Representation Learning by Predicting Image Rotations." In: *CoRR* abs/1803.07728 (2018). arXiv: 1803.07728. URL: http://arxiv.org/abs/1803.07728.

[5] Priya Goyal et al. *VISSL*. https://github.com/facebookresearch/vissl. 2021.

[6] Michael Gutmann and Aapo Hyvärinen. "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models." In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 297–304.

[7] Florian Knoll et al. "fastMRI: A Publicly Available Raw k-Space and DICOM Dataset of Knee Images for Accelerated MR Image Reconstruction Using Machine Learning." In: *Radiology: Artificial Intelligence* 2 (Jan. 2020), e190007. DOI: 10.1148/ryai.2020190007.

[8] Ishan Misra and Laurens van der Maaten. "Self-Supervised Learning of Pretext-Invariant Representations." In: *CoRR* abs/1912.01991 (2019). arXiv: 1912.01991. URL: http://arxiv.org/abs/1912.01991.

[9] van der Walt S amd Schönberger JL et al. *Structural Similarity Index*. 2014. URL: https://scikit-image.org/docs/dev/auto_examples/transform/plot_ssim.html.

[10] Jure Zbontar et al. "fastMRI: An Open Dataset and Benchmarks for Accelerated MRI." In: 2018. arXiv: 1811.08839.

[11] Zongwei Zhou et al. "Unet++: A Nested U-Net Architecture for Medical Image Segmentation." In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Springer, 2018, pp. 3–11.

[12] Zongwei Zhou et al. "UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation." In: *IEEE Transactions on Medical Imaging* (2019).

Algorithm	Pre-training			Fine-tuning						Time Total
	Time	Epochs	SSL Loss	Time	Epochs	Loss	SSIM	PSNR	NMSE	
U-Net (Baseline)	N/A	N/A	N/A	9.29	50	0.072	0.73	26.79	0.042	9.29
RotNet/U-Net	1.08	100	0.39	4.74	10	0.129	0.73	32.06	0.034	5.82
SimCLR/U-Net	1.8	200	2.96	1.9	10	0.129	0.73	32.02	0.034	3.7
PIRL/U-Net	6.41	1000	41.95	1.9	10	0.1645	0.69	30.32	0.044	8.31
SwAV/U-Net++	1.11	100	8.00	4.57	10	0.127	0.67	29.77	0.048	5.68

Table 1. Pre-training and fine-tuning results (time value is in hours). SSL Loss is a SSL method-specific loss (see Section 3 for the details.)

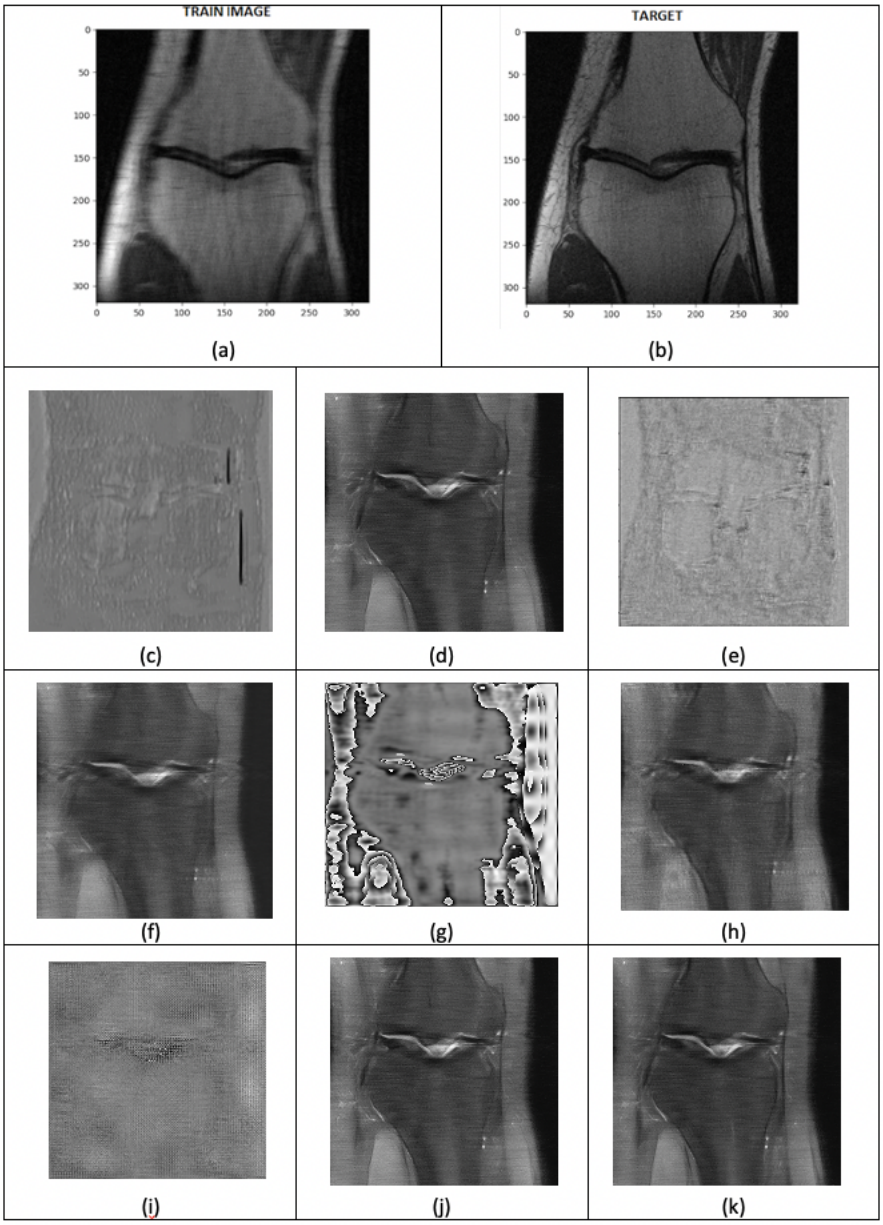

Figure 3. MRI Reconstruction Images: (a) - Sub-sampled Training Input, (b) - Training Target, (c) - RotNet/U-Net PT, (d) - RotNet/U-Net FT, (e) - SwAV/U-Net++ PT, (f) - SwAV/U-Net++ FT, (g) - PIRL/U-Net PT, (h) - PIRL/U-Net FT, (i) - SimCLR/U-Net PT, (j) - SimCLR/U-Net FT, (k) - U-Net Baseline