©RSNA, 2020
10.1148/radiol.2020192173

Appendix E1

Supplemental Materials and Methods

Neural Network Training Data

From 400 short-axis examinations, we used 37,700 1.5T images + 32,380 3.0T images for training, 10,720 1.5T images + 32,380 3.0T images for validation, and 5,167 1.5T images + 4,740 3.0T images for testing. All reported statistical analyses are based on performance on the test set.

Recovery of Fourier Data from DICOM Files

To recover measured k-space data from source Digital Imaging and Communications in Medicine (DICOM) files, we first queried the Acquisition Matrix (0018,1310) field to determine the extent of Fourier zero-padding used by the manufacturer. We effectively reversed this zero-padding by transforming the DICOM pixel data to k-space and removing interpolated data beyond the acquisition matrix geometry.

Fourier Downsampling of Multiframe Data

To generate synthetic multiframe training data, we performed the same downsampling strategy at the time frame of interest and adjacent flanking time frames. The three downsampled frames were stacked as a single volume to provide an input for the multiframe variants of the neural networks.

Hybrid Loss Function

We used the Tensorflow implementation of multiscale structural similarity index (MS-SSIM) and its default settings for filter size = 11, filter sigma = 1.5, k1 = 0.01, and k2 = 0.03. Due to relatively small 128 × 128 matrix size for training data, we only used the first four default MS-SSIM power factors and renormalized them resulting in the weights [0.0517,0.3295,0.3462,0.2726]. For multiframe experiments, we added the 3D L1-loss to the mean of the MS-SSIM losses calculated for each of the three adjacent timeframes.

Other Training Parameters

We used the hyperbolic tangent as the final activation function for all cognitive neural networks (CNNs). Additionally, we used the Adam optimizer with a learning rate of 1e-4. We performed training with early-stopping for a maximum of 25 epochs. We trained a unique set of UNets and SRNets for multiple degrees of upsampling, from 2× to 64×.

Super-Resolution of Low-Resolution Images

To predict a high-resolution image from low-resolution inputs, we z-padded outer k-space of the low-resolution acquisitions, transformed them to the image domain, retained the central 128 × 128, and scaled pixel values to [0,1] prior to CNN inference.

Super-Resolution of Full-Resolution Images

To further super-resolve full-resolution acquisitions, we divided each full-resolution image into tiles, transformed the tiles into k-space, z-padded them to 128 × 128, converted them to image space, and scaled pixel values to [0,1] prior to CNN inference. We then concatenated the super-resolved tiles to form a higher resolution image.

As an example, to super-resolve a 128 × 128 image by a factor of 4 (2× along the row direction and 2× along the column direction), we first subdivide the source image into 4 64 × 64 tiles. We then transform each tile to k-space and z-pad them to 128 × 128. We then convert each tile back to image space, scale the pixel values to [0,1], and use k-UNet to super-resolve each tile. Following super-resolution, we have 4 128 × 128 tiles which, in this case, correspond to the four quadrants of our input image. We then concatenate the tiles relative to their original position in the source image to generate a final 256 × 256 super-resolved image.