Single Image Super Resolution on Heart MRI Images

Horváth Bence – ET2EPO, Sipos Levente - NLLIEC

Abstract

The tradeoff between image quality and capture time continues to be a problem in biomedical imaging. To get higher resolution images, a longer exposure time is needed, which in turn increases the risk of the patient moving, creating unwanted artifacts. An effective solution is to take low resolution images and try to enhance them in post processing. Deep learning super resolution proves to be an effective image processing method. In this paper we created a model for upscaling heart MRI images, based on a mixture of previous methods.

Previous methods

The topic of super resolution is a highly researched area. There are plenty of different methods, with differing results. A method that works well for one problem might be less optimal for another. The larger categories of super resolution are single or whole image, generative or traditional training.

The difference between single or whole image super resolution is the size of the network. Single image can be relatively fast as our scans are around 256x256 pixels. However, when we introduce a third dimension the size of our network and therefore training time explodes. Whole image super resolution is better to be performed if a large computing network is available. With generative training there is a generator network, which creates the upscaled version of the image and there is also a discriminator network which distinguishes between real high-quality images and our generated images. Thus, the generator's error considers the discriminator's decision and that's how training is done. GANs can achieve higher quality images, but are notoriously difficult to train, as there are basically two networks to be trained simultaneously. To achieve manageable training times, we decided to implement single image super resolution.

After trying to train our GAN, we decided to train only train the generator network, failing to overcome the difficulties of GAN training.

Data and data preparation

We use the ACDC Challenge data set, which contains 100 scans of patients for training purposes and 50 for testing purposes. This is a diverse data set of healthy and diseased hearts. This looks like a small amount of data, but since we use single image processing, we can unroll the scans to get more data. Each scan is made up of around 256x256 pixel images in around 7 layers in the third dimension, and one scan contain around 30 shots, the length of a scan. These number are general values as they slightly differ across scans. To get uniform input for our network, we can unroll the images along the z and time axes and pad them to be 256x256 pixels. Some scans were not fit this kind of preprocessing, so they are removed during preprocessing. The downscaled images are acquired by k-space truncation. Images are saved for later use and to easily be able to use only parts of the database. After images are loaded, separated into training validation and testing sets, they are standardized and shuffled randomly. We also employ patching, which is a method of breaking up our 256x256 pixel images into smaller 64x64 pixel patches and reconstructing them into the full image after the upscaling is done.

Model

Our model is based on DenseNet, made of dense blocks with forward skip connections. Dense blocks are made up of convolutional layers with skip connections. Convolutional layers do the feature extraction part of the network, with different kernel sizes and multiple convolutional layers the network can obtain both high- and low levels features. The skip connections make it so that the network considers not only the features that the current convolutional layer extracts, but its relation to the previous features. The ReLU activation function is used in the dense blocks, as it has been shown to be very effective. Batch normalization layers help keep the data flowing through the network not to have too wide a range of values. These dense blocks are connected with skip connections as well and at the end of the network there is a last convolutional layer. We use the ADAM for optimization and MSE for the loss function. We used a batch size of 2 and each training ran for about 100,000 steps. For hyperparameter optimization, rather than changing the activation functions and such, we used what has already been established as solid options and instead changed a higher-level parameter, namely the number of dense blocks and the number of convolutional layers in them.

Results

We used PSNR a common image quality metric, to assess the results of the images. However, nor PSNR nor any other metric can perfectly describe the perceived quality of the human eye, so the best evaluation remains what we can see with our eyes.

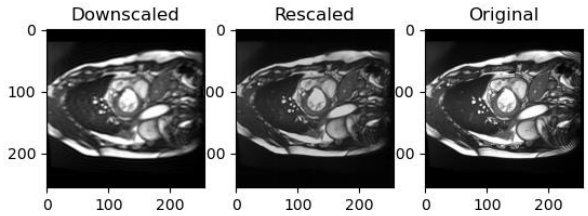

Figure 1. 2 dense blocks with 2 convolutional layers each

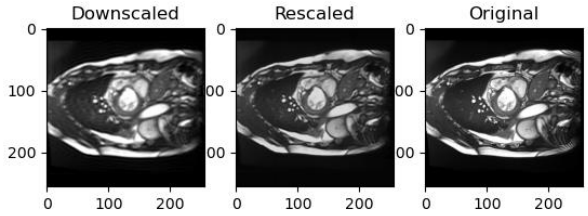

Figure 2. 2 dense blocks with 4 convolutional layers each

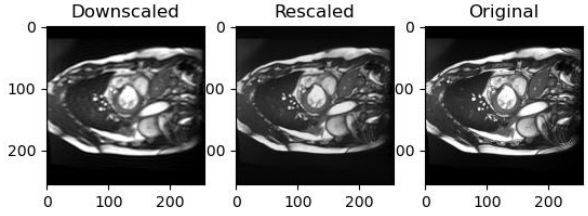

Figure 3. 4 dense blocks with 4 convolutional layers each

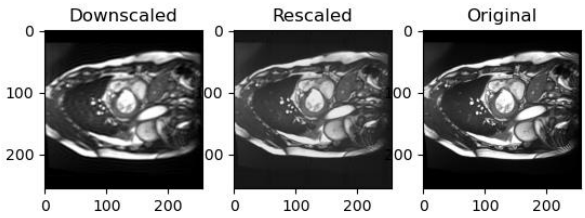

Figure 4. 6 dense blockss with 4 convolutional layers each

All the options perform well, but at 6 dense block we can see a lot of noise being introduced to the system. This might be because this network is so large, that the training takes far longer to converge, or because there are too many parameters.

References

- https://github.com/Hadrien-Cornier/E6040-super-resolution-project/blob/master/main.ipynb
- https://arxiv.org/ftp/arxiv/papers/1707/1707.05425.pdf
- https://arxiv.org/pdf/2003.01217.pdf
- https://arxiv.org/abs/1608.06993v5
- https://arxiv.org/pdf/1609.04802v5.pdf