Journal of Computer and Communications, 2022, *, *-*
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Super-Resolution using Enhanced U-Net for Brain MRI Images

Dalei Jiang, Zifei Han, Xiaohan Zhu, Yang Zhou, Han Yang

Zhejiang University - University of Illinois at Urbana-Champaign Institute, ZJUI, Haining, China
Email: Dalei.19@intl.zju.edu.cn (Dalei Jiang)

Abstract

Super-resolution is an important technique in image processing. It overcomes some hardware limitation failing to get high-resolution image. After machine learning getting involved in, the super-resolution technique gets more efficient in improving the image quality. In this work, we applied super-resolution to the brain MRI images by proposing an enhanced U-Net. Firstly, we used U-Net to realize super-resolution on brain Magnetic Resonance Images (MRI). Secondly, we expanded the functionality of U-Net to the MRI with different contrasts by edge-to-edge training. Finally, we adopted transfer learning and employed convolutional kernel loss function to improve the performance of the U-Net. Experimental results have shown the superiority of the proposed method, e. g., the resolution on rate was boosted from 81.49% by U-Net to 94.22% by our edge-to-edge training.

Keywords

Image Super-Resolution, Machine Learning, Transfer Learning, Convolutional Kernel

1. Introduction

Traditionally, Magnetic Resonance Imaging (MRI) experiments need to optimally balance image resolution, Signal-to-Noise Ratio (SNR), and acquisition time. A higher resolution image shows clearer details, but typically reduces SNR and requires longer scanning time [1]. Moreover, although the cost of low-field MRI (LF-MRI) system is efficient, it does not perform well because the loss of image details when using low image resolution and SNR.

For natural images, Super-Resolution (SR) techniques that improve image quality have been widely studied. Conventional methods can be roughly divided into two groups, i.e., model-based methods like interpolation algorithms [2][3], and learning-based methods such as dictionary learning methods [4][5]. In recent years,

deep learning has exhibited outstanding performance in image super-resolution and has shown great potential for further development [6][7]. The most widely used convolutional neural networks produce superior SR results with higher sharpness and fewer artifacts.

Inspired by the great achievements of deep learning in natural image SR [8], researchers have greatly explored the application of neural networks in MRI, which are expected to improve the quality of MR images without any hardware modification costs.

In this work, the main contributions are four-hold:
1) Apply model-based methods to get rough Super-Resolution results.
2) Use U-Net to achieve super-resolution which has better results than model-based Super-Resolution on T1(longitudinal relaxation time) brain MRI.
3) Remove low-frequency domain, and pick high-frequency parts as the training samples, to extend the network's efficiency to MRI with different contrast.
4) Adopt transfer learning and convolutional kernel in current network to improve the performance.

2. Problem Definition and Theoretical Basis

2.1. Definition of the Resolution

In the image processing, the resolution is sometimes defined as the number of pixels in a picture. However, in this work, we defined the resolution as "the influence range of one unit sample point on the picture". Here, we give a detailed quantitate definition.

We set the spatial linear length of the object that we try to sample as L, which is the linear scale of the object. And we represent the signal as the sum of Fourier series [9]:

$$s(x) = \sum^{\infty} C_n \cdot e^{-j2\pi n\Delta f x} \tag{1}$$

Here, the Δf is the sampling frequency in the spatial domain. According to the Nyquist requirement, Δf should satisfy that $2\pi\Delta f \le L$. Usually, we set $\Delta f = L/2\pi$. In the above equation, we just pick one period in the series to reconstruct the original signal.

When we do the sampling towards the object, the signal we store in the system will be represented as the array in the frequency domain $S(n\Delta f)$. Here, n represents the index in the frequency domain. According to the Fourier transform [10]:

$$s(\omega) = \int_{-\infty}^{\infty} s(t)e^{-j\omega t}dt \tag{2}$$

By applying the discrete form of the formula, we adjust the variable and set:

$$S(n\Delta f) = \int_{-\infty}^{\infty} s(t)e^{-j2\pi n\cdot\Delta f t}dt \tag{3}$$

where the left-hand-side part is what we use to reconstruct the original signal. We mark it as $S(n\Delta f)$. Our goal is to use $S(n\Delta f)$ to form the reflection towards a possible $S(n\Delta f)$, which represents the spatial signal in the object.

Now, we start to construct the function:

$$f : S(n\Delta f) \to \hat{s}(mx) \tag{4}$$

Using the Fourier series, we know that:

$$C_n = c\int_{-\infty}^{\infty} s(t)e^{-j2\pi n\cdot\Delta ft}dt \tag{5}$$

And here,

$$\int_{-\infty}^{\infty} s(t)e^{-j2\pi n\cdot\Delta ft}dt = S(n\Delta f) \tag{6}$$

Thus,

$$C_n = c\cdot S(n\Delta f) \tag{7}$$

The variable c is the constant governed by the system and remains still in one certain background.

After applying the result into the expression of $s(x)$, we get:

$$s(x) = c\cdot\sum_{}^{\infty} S(n\Delta f)\cdot e^{-j2\pi n\cdot\Delta fx} \tag{8}$$

Until now, we have declared how to reconstruct the original signal $s(x)$ using $S(n\Delta f)$. When the sampling point ranges from negative infinity to positive infinity, the result of $s(x)$ will be accurate without any error.

However, infinite sampling for computer system is impossible. Thus, we mark the amount of the sampling points as N. The expression of $s(x)$ becomes:

$$\hat{s}(m\Delta x) = c\cdot\sum_{n=-\frac{N}{2}+1}^{+\frac{N}{2}} S(n\Delta f)\cdot e^{-j2\pi n\cdot\Delta fm\Delta x} \tag{9}$$

Here, $\hat{s}(m\Delta x)$ represents that the signal is discrete and has some error value.

Based on the Nyquist requirement [11],

$$\frac{1}{\Delta x} \le \frac{N\cdot 2\pi\Delta f}{2\pi} \tag{10}$$

Usually, we just set:

$$\Delta x = \frac{1}{N\Delta f} \tag{11}$$

Then, apply equation (11) to (9), we get:

$$\hat{s}(m\Delta x) = c\cdot\sum_{n=-\frac{N}{2}+1}^{+\frac{N}{2}} S(n\Delta f)\cdot e^{-\frac{j2\pi mn}{N}} \tag{12}$$

This formula is just the expression of the Discrete Fourier Transform (DFT) of array $S(n\Delta f)$, and n ranges from $-N/2+1$ to $+N/2$.

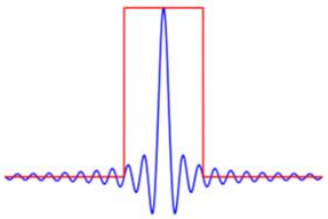

Figure 1 Sampling process in the frequency domain

In Figure 1, we show the sampling process. The graph is in the frequency domain, and the system just pick the points inside some kind of red box, and the points on the blue line outside the red box is not in the range we can sample. This is just like applying a box filter in the frequency domain. According to the inverse Fourier transform, this process equals to convolve a Sinc function in the image domain, where the function shows like:

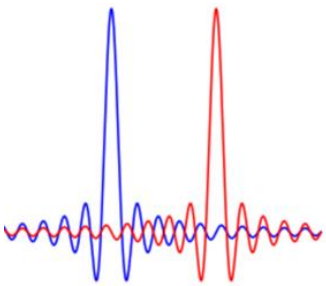

Figure 2 Effect of one single sampling point

Now, we are able to define the resolution in a mathematical way. Figure 2 shows the effect of one single sampling point on the image. Generally, we use the concept "full width at half maxima" (FWHM) to measure the resolution. However, in this work, we mainly focus on the frequency domain, and use the distribution of the points in frequency domain to measure the resolution of the image. We will introduce several parameters in the later sections.

2.2. Definition of the Problem

Using the high-resolution image I_h, we transform it into k-space to obtain the frequency matrix F_h using Fast Fourier Transform (FFT) and shifting. We set the size of I_h to be $N \times N$, and the size of F_h is also $N \times N$.

To construct the low-frequency part, we pick the center part in F_h, and set it as:

$$F_l = F_h[\frac{N}{2} - \frac{M}{2}, \frac{N}{2} + \frac{M}{2}] \tag{13}$$

This means, we choose $M \times M$ sampling points in frequency domain to construct the low frequency image, I_l, using inverse Fourier Transform.

Our problem is when using I_l to do the Super-Resolution to reconstruct the $\hat{I}_h$, it requires to make $\hat{I}_h$ as similar with I_h as possible.

2.3. Evaluation

2.3.1 PSNR

PSNR is a parameter to describe the ratio between the maximum possible power of a signal and the power of corrupting noise that affects the fidelity of its representation. In this research, we use PSNR both on image domain and frequency domain to measure the performance of the network.

2.3.2 SSIM

Structural similarity index measure (SSIM) [12], is used to measure the similarity between the structure of two images as

$$SSIM(x,y)=\frac{(2\mu_x\mu_y+c_1)(2\sigma_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_2)} \tag{14}$$

2.3.3 The weighted sum of the frequency domain point

This value is used to measure the weighted proportion of points in the frequency domain at high frequencies, which will be weighted here in several different ways.

3. Network and Algorithm

3.1. U-Net

U-Net was first proposed by Olaf Ronneberger, et al. [8] in 2015. It is a simple and effective convolution neural network (CNN) widely used for biomedical image segmentation. The U-Net architecture consists of two symmetric parts, a contracting path and an expanding path, as shown in Figure 3. Each path consists of five repeated convolution modules. In the contracting path, the downsampling operation will extract the feature information and reduce spatial information. The downsampling convolution modules consist of one max pooling layer and two 3x3 convolution layers following the ReLU activation function. The upsampling operation will combine the spatial and feature information in the expanding path. The upsampling convolution modules consist of one up-convolution layer and two 3x3 convolution layers following the ReLU activation function. Finally, the output will pass one 1x1 convolution layer to map the feature vector to the desired number of classes.

The advantage of U-Net is that it can use very little training data and yield precise high-resolution images. We use data augmentation to generate sufficient input images from limited sources. For instance, we divide one large image into smaller patches and feed them into the model. Then, we combine the output patches to generate a high-resolution image. Additionally, during the up-sampling operation, the context information is propagated to higher resolution layers. The integration of spatial and feature information makes the predicted high-resolution output more precise.

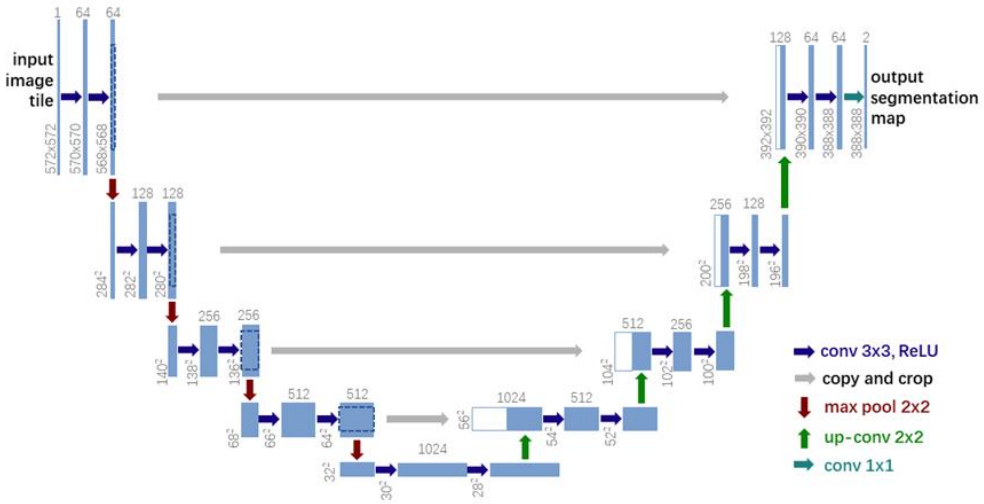

Figure 3 U-Net architecture[8]

3.2. Experimental Setting

3.2.1 Adjustment on Structure

In our implementation of U-Net, we have two modifications. The first is to ignore the 1x1 convolution layer in the final output because we use U-Net to generate a high-resolution image, not image segmentation. Therefore, we do not need to map the output image to class. The second modification is using four repeated convolution modules instead of five. Since the input of our model is minimal, 64x64, over extracting feature information does not further improve the super-resolution performance. Our experiments prove that our enhanced U-Net model is efficient for brain image super-Resolution.

3.2.2 Creating the Training Set

In the training set, original image size is 1760x1760. With 1280 of these images, we can create sufficient patch sets. The images we have are only high resolution images, and to create low resolution images, we transform the image into k-space, and truncate the middle parts. Namely, the low resolution's k-space:

$$F(I_l) = F(I_h)[792:968, 792:968] \tag{15}$$

And other entries in $F(I_l)$ are all zeros. We use the 64x64 patches that are created from the low-resolution training sets.

4. Basic Mathematical Background for Convolutional Kernel

4.1. Instruction for convolutional kernel

In this work, we use convolutional kernel as an extra item to create convolutional kernel loss function. We use it as an item in the loss function and explain what the convolutional kernel is in signal processing.

When we pick the center part of the k-space to create the low-resolution image from high resolution image, we are actually applying a 2-D Sinc function to each pixel in the image space. As we know, the Sinc function has infinite domain, and will produce ripples in the picture. Thus, here we need to create a kernel to convolve the original image, making it generate similar images just like what we get by box filter.

4.2. Visible Function of Convolutional Kernel

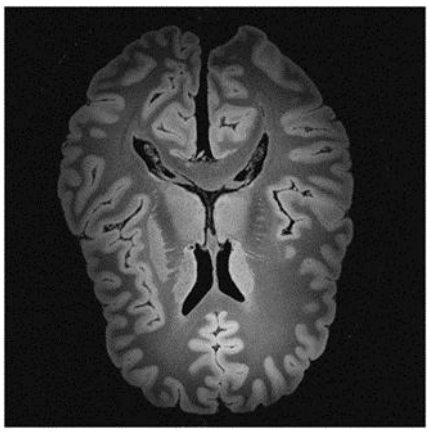

Figure 4 High-resolution brain image

Figure 4 is a high-resolution brain image. When applying the box filter to this image, we will get the low-resolution brain image in Figure 5.

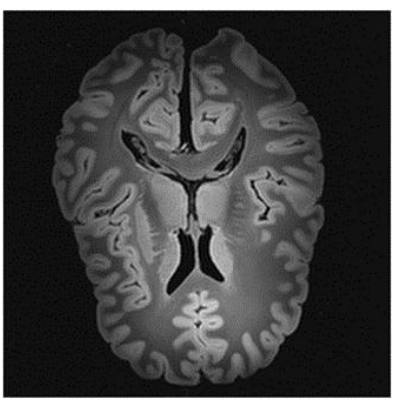

Figure 5 Low-resolution brain image

When using the convolution kernel, the ideal result is shown below:

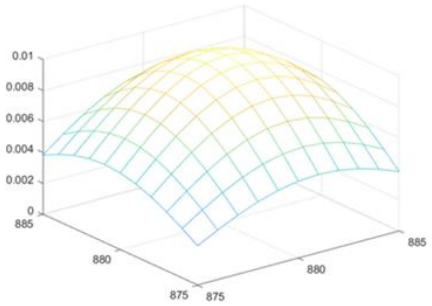

Figure 6 Results of using convolution kernel

However, our result of convolution kernel is discrete and is not ideal. To create our convolution kernel, we list the major procedures below:

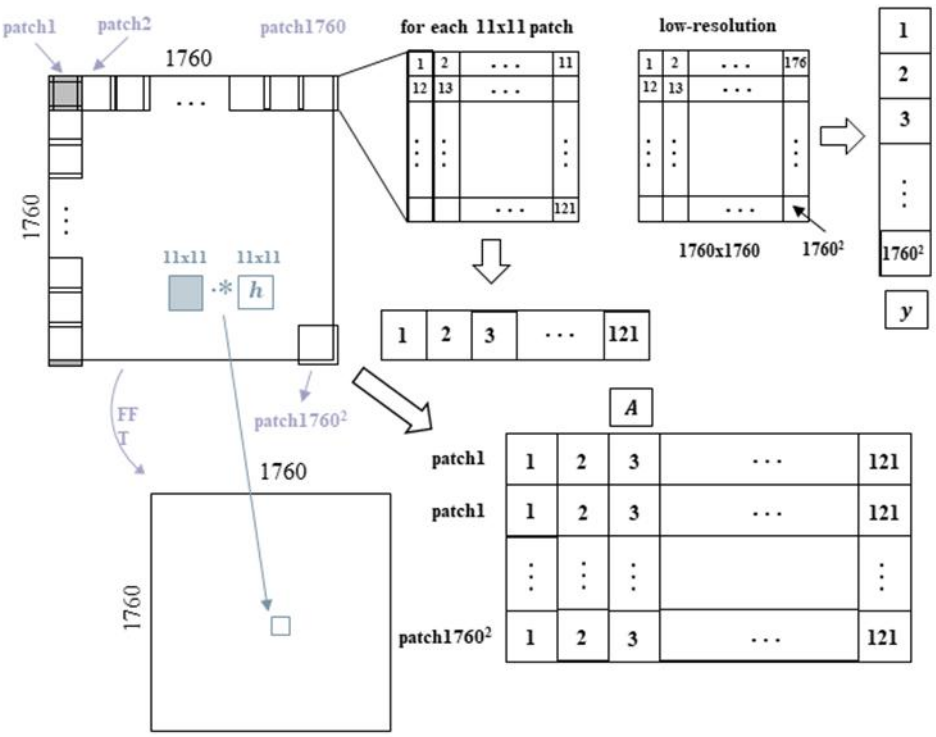

Figure 7 Main procedures of using convolution kernel

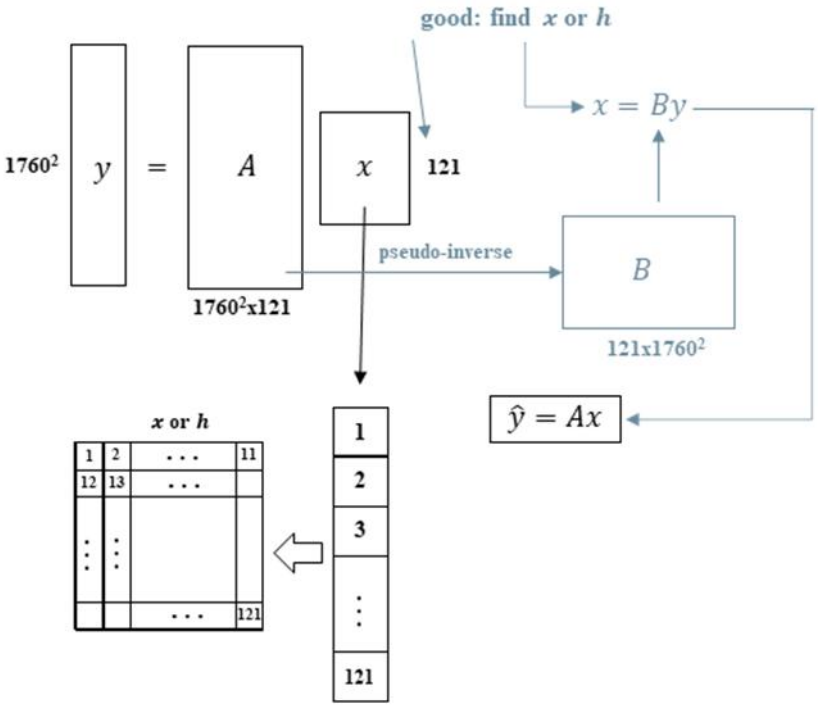

Figure 8 Linear analysis of convolution kernel

In conclusion, we need to solve a linear equation to get specific size of convolutional kernel.

Using the pseudo-inverse, we get the result in Figure 9.

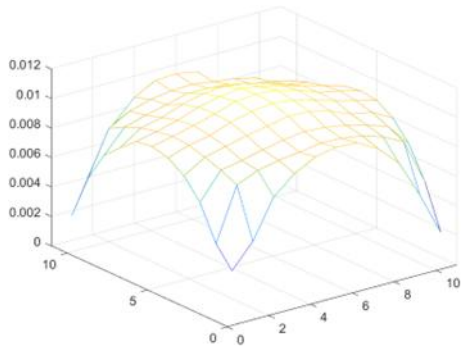

Figure 9 Result by using the pseudo-inverse

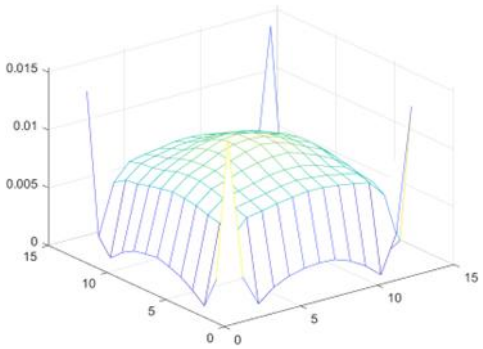

Figure 10 The convolutional kernel

For example, Figure 10 is the convolutional kernel we get with the size 11x11 and 13x13. And the images created by convolution are respectively shown in Figure 11 and Figure 12.

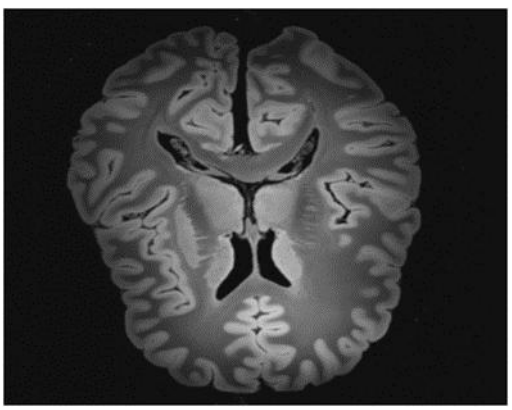

Figure 11 Image created by 11x11 convolution

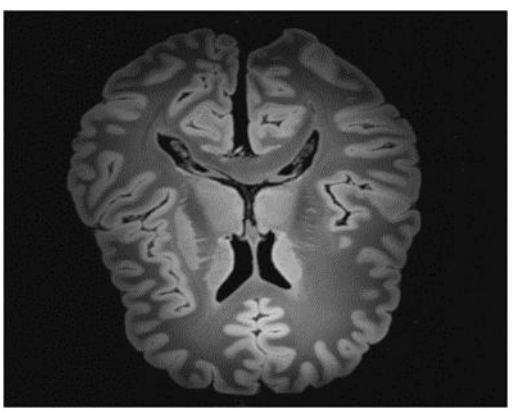

Figure 12 Image created by 13x13 convolution

After comparison, we notice that the resulting images for both kernels have relatively higher resolutions compared with low resolution but also have relatively lower resolutions compared with high resolution. Both kernels could

remove the obvious ripples near the edges in the low-resolution image, which should be the main duty of this kernel h. Furthermore, the difference images between the resulting images and the low-resolution image is also explored; we subtract the low-resolution image from the resulting images and get the two figures of difference images above, which shows the rough contour of the ripples. By detailed observation, the 13×13 kernel performs slightly better in eliminating ripples than the 11×11 one. Thus, in the further research, we will use the 13×13 kernel as our sample.

4.3 Supplement on the mathematical basis of convolution kernels

4.3.1 Solving Ax=y

Four situations of y = Ax, suppose $A \in R^{m \times n}$:

Case 1. Given any m dimensional y vector, which means every y in m dimensional space exists a solution. In this case, A needs to make sure that its column vectors' linear combination should cover all m dimensional space, in another word, A is non-singular and invertible. If we need all base vectors to be enough to form an m dimensional space, so n $\geqslant$ m, and the reminded column vectors could be dependent on other m column vectors, which cause the solution being non-unique, since the dependent vectors have infinite linear combinations. Also, rank(A) = m, because if rank(A) is smaller than m, all base vectors $\{a1, \cdots, an\} \in R^{rank(A)}$ cannot form an m dimensional space.

Case 2. For some m dimensional vector y, there exists a unique solution. When all m dimensional column vectors cannot form an m dimensional space, it indicates that n < m and only can form at most n-dimensional space.

For example, with m = 3 and n = 2, we have two 3-dimensional column vectors, which cannot form the whole 3-dimensional space but only a plane or a line or a point. However, for the y that falls on that specific plane (or line or point), it has the corresponding unique solution x. And since this solution is unique, rank(A) = n. Because if rank(A) is smaller than n, there should exist custom variables that make the solutions not unique, to be specific, one variable can take any value while another variable will change when this variable changes. Only when rank(A) = n, there is no custom variable, x is unique.

Case 3. For every m dimensional vector y, there exists a unique x. This is the same situation as case 1, we n m-dimensional base vectors {a1, ..., an}, only when m = n and all the base vectors are independent, it can form an m-dimensional space. So, A must be a full rank matrix. That is, rank(A) = m = n, A is invertible.

Case 4. For some m dimensional vector y, there exists a non-unique x. This is the same situation as case 2, when rank(A) < n, we have already proved that the

solution is not unique.

4.3.2 Pseudoinverse

Based on the knowledge of linear algebra and our exploration of the four cases about the matrix equations, not all matrix equations have a certain solution; they may have multiple solutions or have no solution.

However, we would like to obtain the kernel h that comes from the reverse operation of convolution (which could be calculated equivalently with an approach of matrix multiplication), and thus getting a "solution" of any matrix equations is quite significant for the determination process of h. Such a solution will be approximate enough such that the least square condition is satisfied.

$$x_{approx} = \underset{\cdot\cdot}{argmin} \left\| y - Ax \right\|^2 \tag{16}$$

For the mostly applied approach to solving matrix equations, the inverse matrix is entailed and plays a crucial role, but the inverse matrix is sometimes unavailable (for the system of no solution) or not so powerful (for the system of multiple solutions). In order to deal with these two cases, we take the Moore-Penrose pseudoinverse, or pseudoinverse for short, into account. This is a generalized version of the inverse matrix, and by utilizing it, we are able to get the "solution" indicated in the previous paragraph.

In this part, we mark the pseudoinverse as A+.

For the case when a matrix equation has no solution for x, we get the most fitting solution that satisfies the least square condition by computing $x_{approx} = A + \cdot b$. That is, though such a x_approx is not a solution to the matrix equation, yet it minimizes$|A \cdot x_{approx} - b|$. For the case when a matrix equation has more than one solution, we get the one with the smallest magnitude. That is, $x_{approx} = A + \cdot b$ has the least $|x_{approx}|$ while satisfying $A \cdot x_{approx} = b$.

5. Training Result

In the test, we find that the training will reach the edge of overfitting at approximately 500 rounds of training. Thus, we train for 500 rounds, and change the training set, and continue to train. After 6 periods, namely, 3000 rounds in total, we terminate the training.

In summary, we use the following parameters to measure the performance of the U-Net:

a) PSNR for the overall result
b) PSNR for the low-resolution part

c) PSNR for the high-resolution part

d) PSNR for the overall k-space

e) the degree of the resolution [13]:

$$\sum_{}^{1759}\sum_{}^{1759} Hammingdisance(i,j)*K[i,j] \tag{17}$$

where Hamming-distance means the Hamming-distance from the target pixel to the center.

We firstly use the common training to get the result:

Table 1 Results of the common training

Methods	Resolution on rate	Image PSNR	Frequency PSNR	Low Frequency PSNR	High Frequency PSNR
Low	18.90	28.86	80.91	90.84	27.56
U-Net	81.49	26.06	80.07	62.13	29.99
U-Net & TT	92.25	28.44	83.10	75.83	30.94
U-Net & CK	92.47	29.51	84.55	79.91	31.33

As shown in Table 1, it is obviously that the U-Net has great contribution in improving the resolution of the image. Here, the resolution rates mean the rates high resolution point in k-space occupying the weighted sum of all the points in the k-space. We use the Hamming distance as the weighted value of all the pixels. After applying the transfer learning and convolutional kernel, the results get even better. We list the result of the resolution rate in Figure 13:

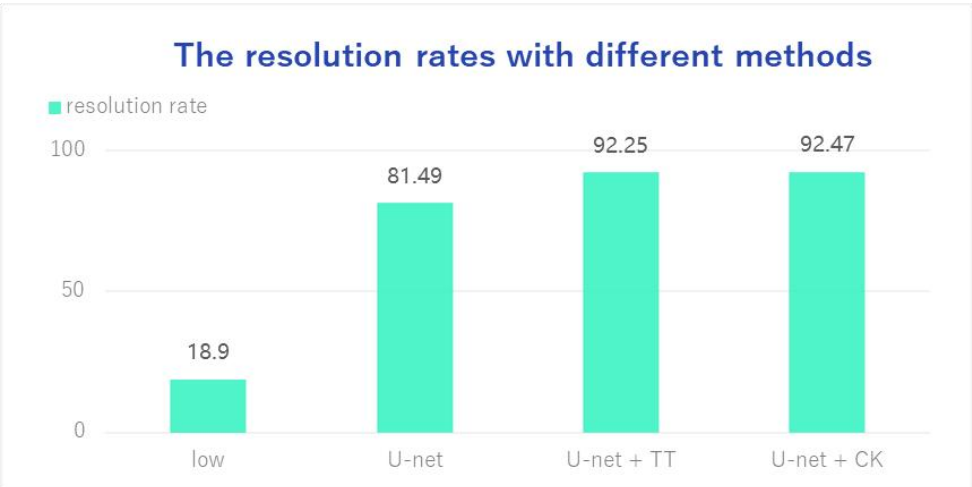

Figure 13 Resolution results of different methods

However, we find that one value may have quite poor performance, image PSNR. This may be caused by inappropriate variables. We have good resolution rates, but the image PSNR does not show reasonable values. It means that the super-resolution causes some distortion on the images. To analyze it in more detail.

We turn to the k-space to find the answer.

We notice that, in the k-space, the value of the PSNR increases compared with the low-resolution image as shown in Figure 14. Nevertheless, the PSNR on the low frequency part decreases sharply after the training, while the high frequency part in rising. It means that our training works. We have increased the images' high-resolution part, and the convolutional kernel has best performance among them. Then, the reason why the PSNR decreases appears in the low-frequency part. We found that the PSNR of the low-frequency part decreases for about 26 in all three method results. Which means we have changed the low-frequency part greatly while training. To avoid great change to the center part of the image, we introduce the edge-to-edge training method.

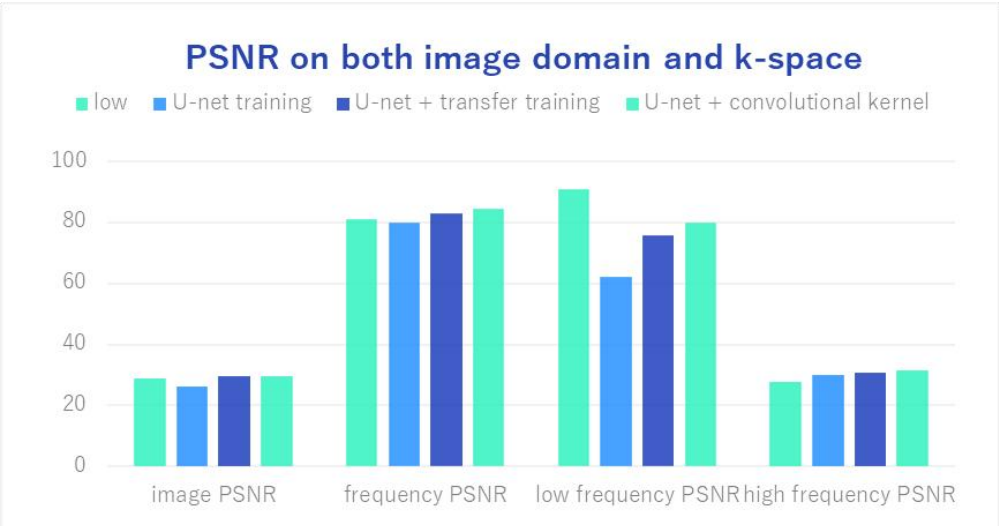

Figure 14 PSNR in both image domain and k-space

6. Edge-to-edge Training

6.1 The Importance of Edge Training

In the experiment, the edge training is quite important in two ways.

Firstly, in the training process, the low-resolution part will be changed and will have a loss rate of approximately 35%. This serious loss in the center part of k-space will result in the poor performance in its PSNR value. Thus, actually in the practical use, we will use the edge-to-edge training. We dig the center of the k-space and mark it as "CORE". After training, we put the CORE back to for the eventual image. In this way, the center part of the k-space will not be changed greatly, and one another benefits is that, if we unpatchify all the patches directly, usually the image will have clear seams between different patches. However, if we use the edge training, the final result will be quite smooth and without the traces of unpatchify.

We need to propose here that in MRI imaging, we apply strong magnetic fields to atoms. When the nucleus returns to its original energy state through some relaxation process, we can get some very useful information of energy radio frequency. Through the different signals emitted by different material properties of human tissues, we can realize the contrast of MRI images. We can notice that when we apply magnetic fields to different properties, we will get different kinds of "feedback" from different matter. This feedback is usually concluded into two parameters, one is the time constant T1 for the excited protons to return to the original energy state, and the other is the transverse relaxation time, T2. T2 is used to describe the time of the spin protons to lose phase coherence. Different human tissues will have different time constants T1, T2.

When we create MRI images, the different magnetic field strengths, frequencies causes that we cannot apply one set of MRI properties to another MRI images. That is, the low-frequency signal features of MRI are not shared. It makes our training unstable. However, although the feedback signals within the tissue are not determined, the boundaries are always fixed, and these fixed features are what we call high-frequency signals.

Due to the different property of the MRI scanning frequency, the training in one kind of MRI image may not be efficient in another MRI image with different contrast. Thus, by digging the center of the k-space, we will only train the high-resolution part of the image. The low frequency part will have different property, however, the part with sharp edges will have similar location distributions in all kinds of MRI images. It makes that the training in different MRI image will share similar properties and make edge-to-edge training a quite useful method in super-resolution. In the following experiment, we will use the edge-to-edge training to re-train the part we have done previously.

Some different parts are that, after digging the center of the k-space, the image domain will appear complex numbers. In this case, we only train the magnitude part of the image, and put the phase part back. Because the phase part will not have great error between high resolution and low resolution, we do not need to train it.

To train the real part of the image, we need to change the activate function of U-Net. The ReLU function matches all the negative value to zero, but in real part, we have many negative values. Thus, we change it to the identity function. By contrast, the real-imaginary pair training is not a good choice, since the imaginary parts does not have clear attributes for U-Net to fetch, and the training result is not good. Therefore, we choose the magnitude-phase training eventually.

6.2. Training Results

Table 2 Results of the edge-to-edge training

Methods	Resolution on rate	Image PSNR	Frequency PSNR	Low Frequency PSNR	High Frequency PSNR
Low	18.90	28.86	80.91	90.84	27.56
U-Net	81.49	26.06	80.07	62.13	29.99
U-Net & TT	92.25	28.44	83.10	75.83	30.94
U-Net & CK	92.47	29.51	84.55	79.91	31.33
Edge	94.22	28.96	83.98	91.49	30.63

Table 2 shows the results of the edge-to-edge training. Obviously, the edge-to-edge training has evolution on the resolution rate. Besides, the rates of the PNSR have different improvements. This may be caused by the core part inserting in the k-space directly.

One thing to notice is that, we do not apply transfer learning or convolutional kernel. However, some rates have similar values to the rate shown by transfer learning or convolutional kernel. Even some rate, such as the image PSNR and frequency PSNR has better performance than the transfer learning. In a word, edge-to-edge learning has quite great improvements on the performance of the U-Net, and can keep reality at the same time.

7 Conclusion

7.1 Achievements

Super-resolution is an important field in computer vision and image signal processing. In this work, we adopt the U-Net to achieve the super-resolution on MRI image. Firstly, we use the U-Net to do the normal training. The resolution of the image has arisen greatly. However, the PSNR of the image domain and frequency domain do not have obvious improvement.

Then, we use the transfer training and convolutional kernel loss function to help the U-Net to do the training. The results are that the transfer learning slightly improve the net-work result in PSNR value on the image and k-space. One thing is more important, and that is it greatly improves the resolution of the image. When we try to use the convolutional kernel, the results get even better, and it improves the super-resolution quality by about 5%. However, one thing is influencing the result. The low frequency part has been changed greatly and it causes the distortion on the final images.

Therefore, we introduce the edge-to-edge training. By applying the digging of the

k-space, we keep the center part of the k-space unchanged, and only do the training on the high-resolution part. In this way, we found that the result gets much better, even without the transfer learning and the convolutional kernel.

7.2 Drawbacks and Further Developments

Due to the time limitation, we do not achieve significant performance gains on the U-Net. The U-Net should have more powerful performance on super-resolution. We have improved the resolution of the image greatly, but the image still has serious distortion. This is the direction we need to improve.

Our next step of the research can be the development of some result processing strategies. For example, we find that when we want to apply the convolutional kernel loss function item to the edge-to-edge training, we need to create a filter which is the combination of two box filter, and then apply a new kind of convolutional kernel rather than Sinc function. It needs further mathematical works.

Besides, in the process of unpatchify the image. We find that if we combine all the patches directly, the picture will have seams between the patches. However, if we use the average value in the overlapping part between the patches, the resolution of the image will sharply decrease. Thus, we need to find a better way to reconstruct the image from the patch sets.

References

[1] D. Brunet, E. R. Vrscay and Z. Wang. On the mathematical properties of the structural similarity index[J]. IEEE Transactions on Image Processing, 2011, 21(4):1488-1499.

[2] G. Yang, L. Zhang, M. Zhou, A. Liu, X. Chen, Z. Xiong and F. Wu. Model-guided multi-contrast deep unfolding network for MRI super-resolution reconstruction[C]. In Proceedings of the 30th ACM International Conference on Multimedia, 2022:3974-3982.

[3] T. Peleg and M. Elad. A statistical prediction model based on sparse representations for singles image super-resolution[J]. IEEE Transactions on Image Processing, 2014, 23(6):2569-2582.

[4] S. Wang, L. Zhang, Y. Liang and Q. Pan. Semi-coupled dictionary learning with applications to image super-resolution and photo-sketch synthesis[C]. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012:2216-2223.

[5] L. He, H. Qi, R. Zaretzki. Beta process joint dictionary learning for coupled feature spaces with application to single image super-resolution[C]. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013:345-352.

[6] C. R. Helmrich, M. Siekmann, S. Becker, S. Bosse, D. Marpe and T. Wiegand. Xpsnr: A low-complexity extension of the perceptually weighted peak signal-to-noise ratio for high-resolution video quality assessment[C]. In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020:2727-2731.

[7] A. Hore and D. Ziou. Image quality metrics: PSNR vs. SSIM[C]. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2010:2366-2369.

[8] O. Ronneberger, P. Fischer and T. Brox. U-Net: Convolutional networks for biomedical image segmentation[C]. International Conference on Medical Image Computing and

Computer-Assisted Intervention(MICCAI), 2015:234-241.

[9] Tolstov G P. Fourier series[M]. Courier Corporation, 2012.

[10] Bracewell R N. The fourier transform[J]. Scientific American, 1989, 260(6): 86-95.

[11] Por E, van Kooten M, Sarkovic V. Nyquist–Shannon sampling theorem[J]. Leiden University, 2019, 1: 1.

[12] Dosselmann R, Yang X D. A comprehensive assessment of the structural similarity index[J]. Signal, Image and Video Processing, 2011, 5(1): 81-91.

[13] Norouzi M, Fleet D J, Salakhutdinov R R. Hamming distance metric learning[J]. Advances in neural information processing systems, 2012, 25.

[14] F. Shi, J. Cheng, L. Wang, P. Yap and D. Shen. LRTV: MR image super-resolution with low-rank and total variation regularizations[J]. IEEE Transactions on Medical Imaging, 2015:2459-2466.

[15] F. Shi, J. Cheng, L. Wang, P. Yap and D. Shen. Low-rank total variation for image super-resolution[C]. International Conference on Medical Image Computing and Computer-Assisted Intervention(MICCAI), 2013:155-162.

[16] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity[J]. IEEE Transactions on Image Processing, 2004:600-612.

[17] S. C. Park, M. K. Park and M. G. Kang. Super-resolution image reconstruction: a technical overview. IEEE Signal Processing Magazine, 2003, 20(3):21-36.

[18] P. Purkait and B. Chanda. Super resolution image reconstruction through Bregman iteration using morphologic regularization[J]. IEEE Transactions on Image Processing, 2012, 21(9):4029-4039.

[19] M. O. Camponez, E. O. T. Salles and M. Sarcinelli-Filho. Super-resolution image reconstruction using nonparametric Bayesian INLA approximation[J]. IEEE Transactions on Image Processing, 2012, 21(8):3491-3501.

[20] M. Protter, M. Elad, H. Takeda and P. Milanfar. Generalizing the nonlocal-means to super-resolution reconstruction[J]. IEEE Transactions on Image Processing, 2008, 18(1):36-51.