Context-Aware Medical Image Super-Resolution Using Convolutional Neural Networks

Abstract

The use of medical imaging such as computed tomography (CT) and magnetic resonance imaging (MRI) is tremendous in the clinical setting for patient diagnosis and treatment. The utility of a medical image is dependent on its quality. This presents a challenge as the high-resolution medical images require large radiative dosages for extended exposure times potentially putting the patient at risk. Extended scans leave images prone to motion artifacts. Super-resolution's ability to artificially generate a higher resolution scan eliminates the need for harmful, high-power scans. To alleviate these problems, we propose a novel, context-aware, deep learning model for single image super resolution (SR): Our novel context-aware solution, Context-aware super-resolution using convolutional neural networks (C-SRCNN). Our model takes low-resolution images as its input and generates high-resolution images as its output. The proposed model directly learns an end-to-end mapping between low- and high-resolution images. C-SRCNN differs from existing deep learning based SR methodologies through the incorporation of contextual information. This information allows for increased performance and superior image reconstruction with minimal additional computational load. We have also investigated the effects of different implementations of context and have explored applications in other fields. Our experimental results on benchmark datasets show that our context-aware model outperforms other super-resolution methods.

1 Introduction

The super-resolution (SR) problem, especially single image super resolution, has held a foothold in the field of computer vision for decades. Single image super resolution aims to artificially generate a high-resolution image (HR) from a single low-resolution image (LR) through pixel interpolation.

Recent state-of-the-art approaches have used deep learning technologies, e.g., convolutional neural networks (CNNs). These new techniques such as SRCNN [1] have vastly outperformed traditional ones, such as sparse coding, by using a deep CNN to represent the sparse coding solution pipeline. Patch extraction and aggregation are done by using convolutional layers, which involves them in the optimization process. However, SRCNN performs a one-to-one mapping from low-resolution to high-resolution patch and does not consider neighborhood data in its calculations.

In this paper, we show that a context-aware solution to super-resolution improves the performance of deep learning SR models. The use of context is extremely beneficial in SR pertaining to medical images. Context allows the use of the similarities between anatomical structures within one image as well as across images to better predict the output patch, surpassing the practical and physical limitations imposed upon current medical imaging systems [2, 3] while drastically reducing scan time. Our super-resolution model increases levels of patient comfort and quality of diagnosis while reducing risk [4, 5]. We use a multi-channel input as well as a deeper network to increase the performance of our super-resolution model. We also investigate multiple implementations of utilizing contextual information and the effect of hyper-parameters on our model. Experimentally, we confirm that our model outperforms existing approaches on both general and medical benchmarking image datasets.

2 Related Works

2.1 Example-based Techniques

The SR problem is challenging due the multiplicity of solutions for the inverse operation for a given pixel. Essentially, more than one solution algorithm can be applied to the SR problem. The lack of a unique solution leads to the search for the best such algorithm in a potentially large set of solutions.

Traditionally, the issue of multiplicity is handled by constraining the solution space through the use of prior knowledge. These techniques, such as sparse coding, focus on the use of example-based algorithms. These algorithms take pairs of low- and high- resolution patches in pairs and attempt to learn a mapping to translate the low-resolution patches into high resolution ones. Example pair techniques are able to be created for both general images as well as for a specific type of image, say medical images, depending on the training set of examples provided. However, the existence of multiple low-resolution images needed for example-based super-resolution methods is rare and limits the possibilities and applications of super-resolution. The state-of-the-art representative example-based super-resolution model is sparse coding.

Sparse coding models [6, 7, 8, 9] first crops overlapping patches from a given origi-

nal image and pre-processes them (normalization etc.). These images are encoded with a low-resolution dictionary. The sparse coefficients are then passed through a high-resolution dictionary which generates high-resolutions patches that are aggregated to reconstruct a full high-resolution image. This pipeline is common to most external example-based techniques and improvements have focused on the optimization of dictionaries and the mapping function, only considering the patches and not the super-resolution pipeline as a whole. The model also does not learn and improve patch extraction and synthesis, considering the steps as parts of pre-processing. A unified pipeline would allow for a streamlined optimization of the entire super-resolution process.

2.2 Convolutional Neural Networks for Super Resolution

Super-resolution using Convolutional Neural Networks(SRCNN) uses three fully convolutional layers to extract/represent patches, perform non-linear mappings between low/high resolution patches, and reconstruct patches to form a complete, high-resolution output image. This methodology allows for the the unification of the optimization pipeline, allowing the performance of the model to scale with the amount of data provided for training.

SRCNN uses a CNN to integrate the complete pipeline into optimization. CNNs have great promise recently because of their application in image classification problems. The ideas from image classification are applied to SR with the HR image serving as the self-label to the LR image. The advantage of CNN comes from the use of the Rectified Linear Unit (ReLU) [10] and larger datasets, allowing for the modeling of complex functions and increased performance with sufficient amounts of data.

3 Context-Aware Medical Image Super-Resolution with Convolutional Neural Networks (C-SRCNN)

3.1 The Proposed C-SRCNN Model

The proposed C-SRCNN model is a single image super resolution framework focusing on medical imaging that takes both low-resolution patches as well as their contextual information to generate high-resolution patches, which are eventually reconstructed to form a final, higher resolution medical image. There are several alternatives to integrate context into the model. The most straight forward solution is to take extremely large patches. Although large patches introduce contextual information into modeling, the relationship in between the context is not utilized. The alternative approach is to take all adjacent patches and stack them vertically into a multi-channel fashion. Since the anatomical structure of different humans are similar, the multi-channel input enables our model to learn the structure relationship between corresponding locations. In this work, we study the performance of both C-SRCNN with extended patches (Sec. 3.1.1) and multi-channel C-SRCNN (Sec. 3.1.2).

Once we get the input, either large patches or multi-channel patches, they are passed through three convolution layers with filter size 9×9, 5×5, and 5×5, respectively. ReLU

activation function is applied after the first two convolution layers. We take the output of the last convolution layer as the generated patch of the proposed model.

3.1.1 C-SRCNN with Extended Patches

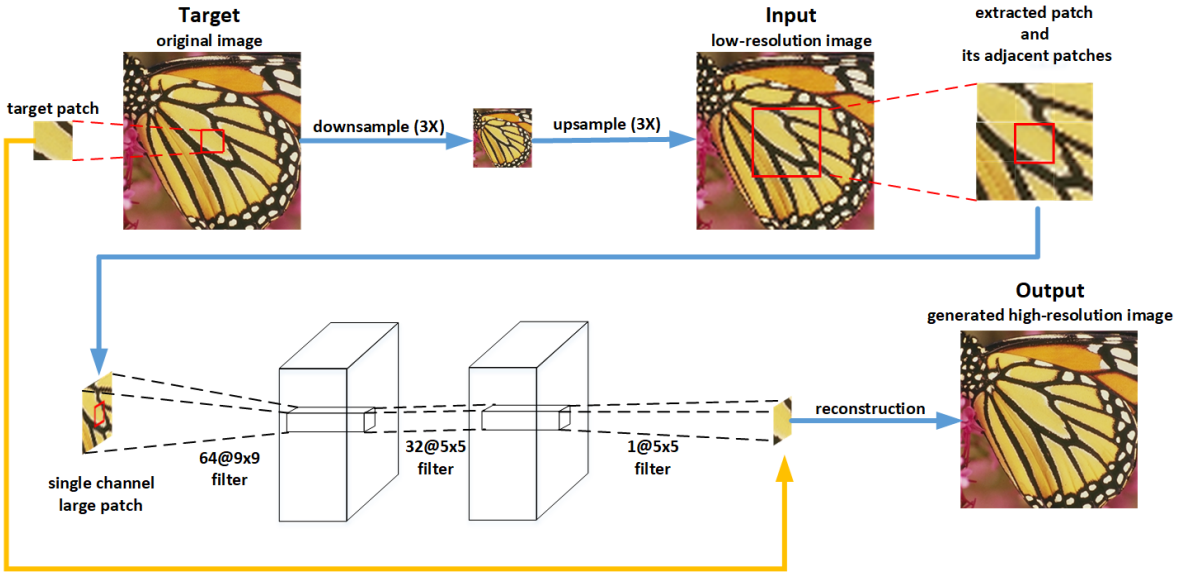

Figure 1: The workflow of C-SRCNN with extended patches.

The workflow of C-SRCNN with extended patches is shown in Fig. 1. Given an image, we first downscale and then upscale it to get its corresponding low-resolution images of the same size. Second, we use sliding windows to extract high- and low-resolution patch pairs from the same location of original images and low-resolution images, respectively. The low resolution patches are of size 9 times that of the high-resolution patches. These extended patches are then used as input of the aforementioned three-layer convolutional network, and the output of the network are the generated high-resolution patches. Finally, we reconstruct a final, high-resolution images from the generated patches. This model is designated as E-SRCNN. The performance is reported in Sec. 4.

3.1.2 Multi-channel C-SRCNN

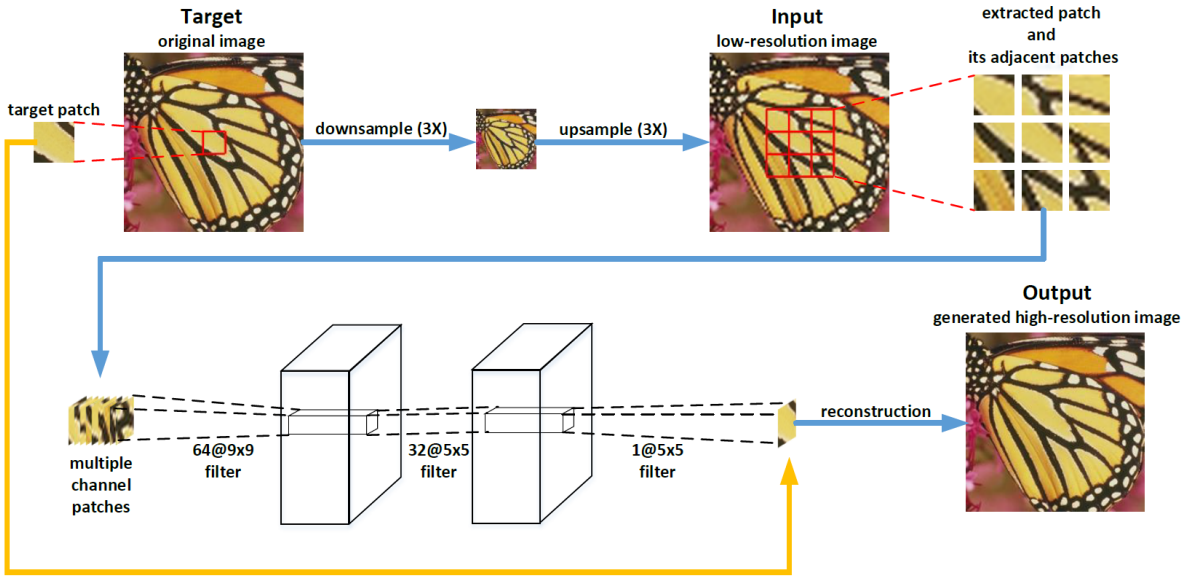

Figure 2: The workflow of C-SRCNN.

The workflow of multi-channel C-SRCNN is shown in Fig. 2. Given an image, we first downscale and then upscale it to get its corresponding low-resolution images of the same size. Second, we use sliding windows to extract high- and low-resolution patch pairs from the same location of original images and low-resolution images, respectively. Here, each low-resolution patch and its adjacent patches are stacked into a multi-channel patch under the same order: upper left patch at the top and lower right patch at the bottom. Third, the stacked multi-channel patches are then used as input of the aforementioned three-layer convolutional network, and the output of the network are the generated high-resolution patches. Finally, we reconstruct final, high-resolution images from the generated patches. In this study, we investigate two variants of multi-channel C-SRCNN, 4-neighbor version (5 channels in total) and 8-neighbor version (9 channels in total). The performance is reported in Sec. 4.

3.2 Relationship to SRCNN

The C-SRCNN model builds upon the work of SRCNN as a light-weight yet powerful model for SR. C-SRCNN differs from SRCNN mainly in its multi-channel input, drastically changing the processes of patch extraction. The additional patches C-SRCNN collects are merged together in the in the non-linear mapping. We can quantify the common goal as the following:

> Find mapping F so that $F(\mathbf{X})$ resembles $\mathbf{Y}$, where $\mathbf{X}$ represents the low-resolution input and $\mathbf{Y}$ represents the original high-resolution input.

We can split this problem into three distinct steps:

1. **Patch Extraction:** From the low resolution image we extract patches from the low resolution image $\mathbf{X}$. This patch is represented as an array of intensity values. We also consider the surrounding patches in this step to create a multi-channel hybridized patch. Each of these multidimensional arrays comprises a set of feature maps.

2. **Convolution:** We convolve the arrays to produce other arrays. In this manner we introduce a non-linear mapping between input and output. Each of these arrays represents an intermediate feature map.

3. **Aggregation:** We take outputted high-resolution patches and synthesize them using weighted averages to generate the final high-resolution image. This image should be similar to our target image $\mathbf{Y}$.

These three steps form a bijection with our C-SRCNN model. Patch Extraction is represented by the input layer. Convolution as the hidden layer. Finally, Aggregation as the output layer.

3.2.1 Patch Extraction

From our input array we implement convolution and ReLU activation ($\max(0, x)$) to generate a basis. This patch extraction and representation of the feature maps can be represented as a function of the form:

$$F_1(\mathbf{X}) = \max(0, W_1 * \mathbf{X} + B_1)$$

where W_1 and B_1 represent the weights and biases of the first layer, respectively. So we perform the convolution operation '$*$' and activate the layer using ReLU. We can see that W_1 corresponds to n_1 filters with size $c \times f_1 \times f_1$. Where c, f_1 are the number of input channels and the filter size, respectively. Thus, W_1 applies n_1 convolutions with kernel size $c \times f_1 \times f_1$ to create an n_1 dimensional feature map. B_1 is an n_1 dimensional bias vector with each element associated to a filter.

3.2.2 Convolution

We introduce a nonlinear mapping on these n_1 dimensional features to create n_2 dimensional features of a high-resolution image. These operation is accomplished through a nonlinear mapping on 5×5 patches of the feature map. This can be represented as

$$F_2(\mathbf{X}) = \max(0, W_2 * F_1(\mathbf{X}) + B_2)$$

As before, W_2, B_2 are the weights and biases of the layer. We perform ReLU activation and this introduces non-linearity. W_2 has n_2 filters of size $n_1 \times f_2 \times f_2$ with B_2 having n_2 dimensions. More layers can be added for increased non-linearity but at the increasing expense of computational load leading to diminishing returns.

3.2.3 Aggregation

We note that we just need to turn our dictionary of n_2 high-resolution features into an image. To do this we implement our third and final convolution to aggregate the features into an image. Thus, we have

$$F(\mathbf{X}) = W_3 * F_2(\mathbf{X}) + B_3$$

with W_3 and B_3 they weights and biases, respectively. W_3 is consists of a filter of size $n_2 \times f_3 \times f_3$ and B_3 with B_3 having only one dimensions so we return the desired one channel output.

In this manner, the third layer represents a linear filter that serves to average the high-resolution patches if they are in the image domain and projects them into the image domain if they are not.

Each of these layers is motivated by a different process; however, each is represented by a convolutional layer of similar design. Together these layers form the convolutional neural network where we are able to optimize all the weights and biases. In this manner, C-SRCNN serves as an extension of the simple yet fundamental SRCNN model [1].

3.3 Training

To learn the end to end mapping between the low-resolution and high-resolution images we must estimate our set of parameters $\Theta = \{W_1, W_2, W_3, B_1, B_2, B_3\}$. To do this we wish to minimize our loss function J between the reconstructed images using Θ, $F(\mathbf{X}; \Theta)$, and the corresponding ground-truth high-resolution image $\mathbf{Y}$. As a measurement of loss we use MSE (Mean-Squared Error), a form of L2 loss. For a set of low-resolution inputs $\{\mathbf{X}_i\}$ and corresponding high-resolution images $\{\mathbf{Y}_i\}$ we define MSE for a given Θ as

$$J(\Theta) = \frac{1}{n} \sum_{i=1}^{n} (F(\mathbf{X}_i; \Theta) - \mathbf{Y}_i)^2$$

where n is the number of images in the batch. It must be noted that we are not limited in our choice of loss function. Unlike hand-crafted techniques, a convolutional model is able to adapt to a better metric if one becomes apparent during training. However, we choose MSE as our loss function because a low MSE predicts a high PSNR (Peak Signal to Noise Ratio). PSNR is a widely-used metric for quantitatively evaluating image restoration quality and is indicative of perceived quality as well. MSE also serves as a favorable indicator of SSIM (Structural Similarity Index). SSIM is a robust measure of image restoration focusing on the reconstruction of structural details [11].

The weights and biases are optimized using Adam optimizer [12] initialized with a learning rate, η, of 0.001, exponential decay rates with β_1 to be .9 and β_2 to be .999, and ϵ of 10^{-8}. The Adam optimizer utilizes adaptive moment estimates and individualized learning rates for each parameter. This provides better performance and efficiency than traditional stochastic gradient descent optimization. The weights at epoch t are updated in the form

$$\Theta_t = \Theta_{t-1} - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

$\hat{m}_t, \hat{v}_t$ are the bias corrected first and second order moments, respectively. Each is derived from the stochastic gradient of the previous weights. The weights are initialized using a truncated normal distribution centered around 0.

During training, we prepare batches of high-resolution patches of size $f_{sub} \times f_{sub} \times c$. This is our set $\{\mathbf{Y}_i\}$. We use a Gaussian kernel to down-sample these patches by the scaling factor, then upsample using bicubic interpolation. This forms the corresponding low-resolution patches $\{\mathbf{X}_i\}$.

To prevent the loss of any contextual information we use padding to prevent any data loss. This means our model returns an image of the same size compared to the input.

We implement our model using the Tensorflow framework [13] in Python. Cuda and cuDNN were used for accelerated computation on our Nvidia GTX 980Ti GPU.

4 Experiments

We investigate the role of contextual data in the model's performance. We also look at the model's performance in different datasets. In our experimentation, we compare the performance of our model with traditional and deep-learning based super-resolution methods on benchmark datasets both quantitatively and qualitatively.

4.1 General Images

We first evaluate the addition of context in the setting of general images. We expect there to be a slight gain in performance as general images do not share a great deal of self-similarity. Nevertheless, we still do expect a gain in performance as a result of the fact that we are supplying the model with contextual information. We will use bicubic interpolation and SRCNN as benchmarks.

We use the General-100 Dataset [14] for training and Set-5[15] and Set-14 [8] for testing, a similar setup to past literature for comparison. We use the same hyper parameters for each model including patch size of 33, stride of 14. From the dataset we derive over 90,000 patches and use 9,000 for validation.

The training convergence curves are shown in Figure 3. We see that at the end of training we achieve a PSNR of **30.25** with the the eight neighbor model in Set 5 and a PSNR of **25.481** in Set 14. We see that the context-aware models surpass both SRCNN and bicubic interpolation. This indicates that context does perform favorably compared to both traditional and cutting-edge technique. The data is summarized in Tables 1 and 2 with other super-resolution methods. Some sample images are shown in Figures 4 and 5 with a detail of a smaller section.

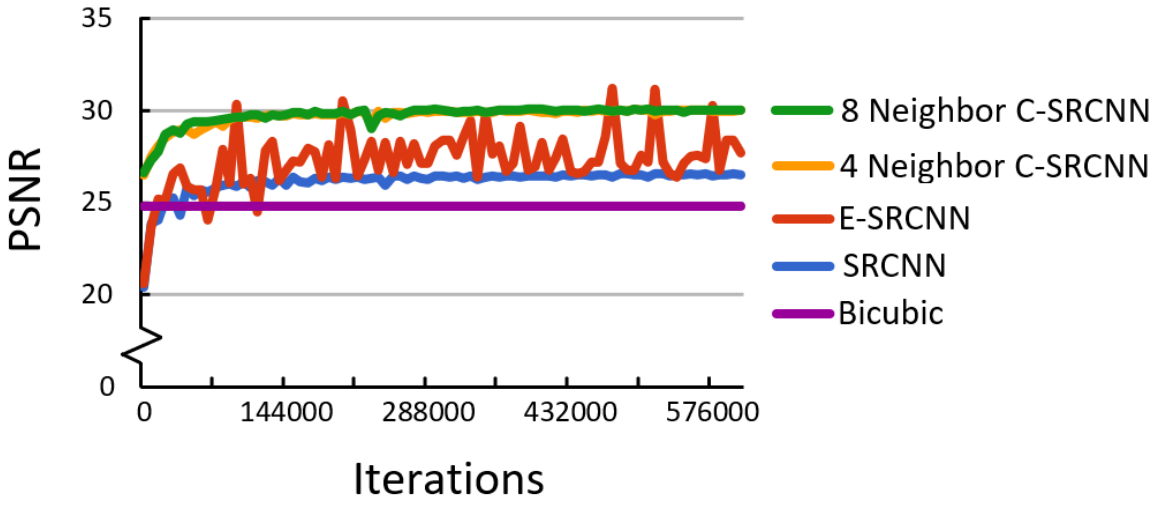

Figure 3: Training PSNR(dB) over iterations.

Dataset	Nearest Neighbor	Bicubic	SRCNN	E-SRCNN	4 Neighbor C-SRCNN	8 Neighbor C-SRCNN
Set 5	25.033	26.902	29.910	28.642	30.038	30.253
Set 14	22.275	24.152	25.319	22.130	25.160	25.481

Table 1: Testing PSNR(dB) for general datasets.

Dataset	Nearest Neighbor	Bicubic	SRCNN	E-SRCNN	4 Neighbor C-SRCNN	8 Neighbor C-SRCNN
Set 5	0.7637	0.7928	0.8791	0.8760	0.8786	0.8811
Set 14	0.6816	0.6948	0.7948	0.7709	0.7947	0.7953

Table 2: Testing SSIM for general datasets.

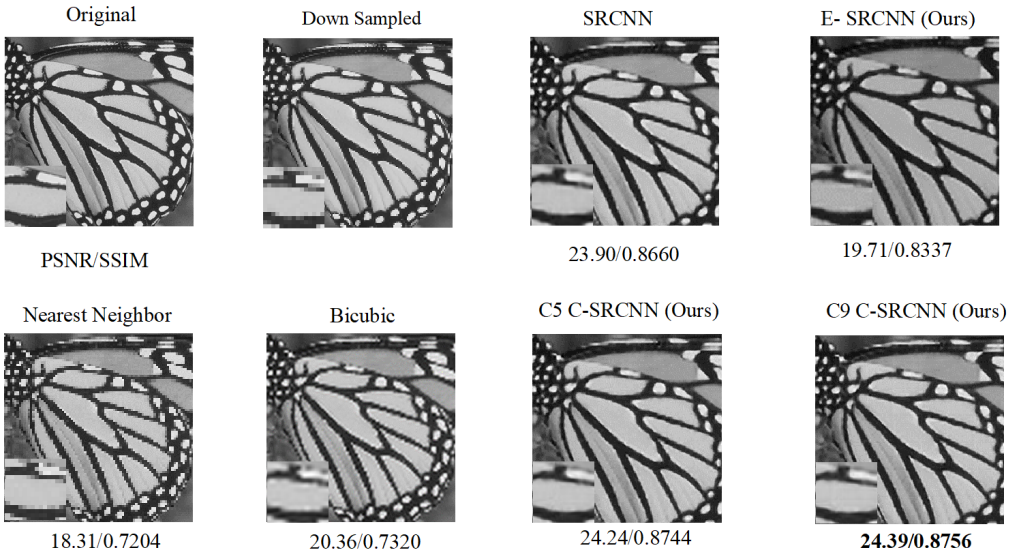

Figure 4: Example of general image testing.

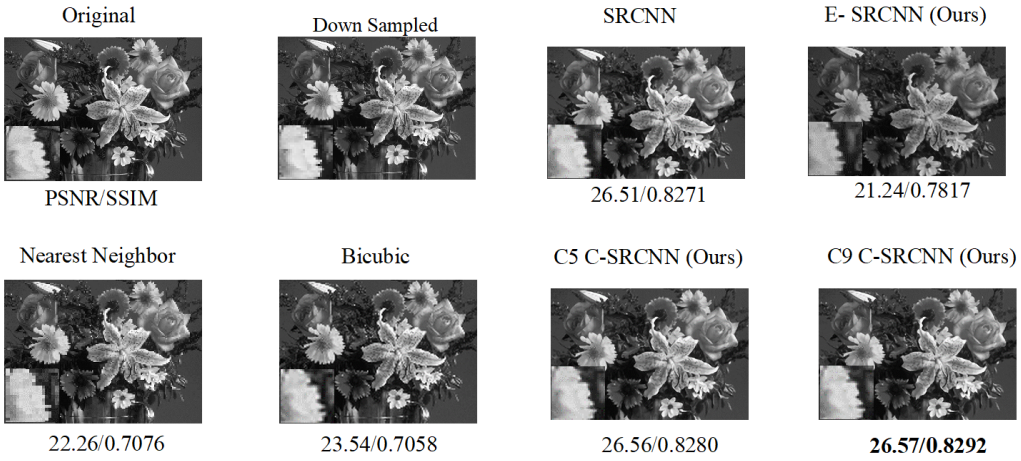

Figure 5: Example of general image testing.

4.1.1 Learned Weights

We present all 64 nine-channel first layer filters as 576 images in Figure 6. We see that each filter learns a specific function. Some filters may represent Gaussian transforms, other specialize in edge or texture detection. The feature maps of the first layer contain mostly structural information such as edges and second layer features differ based on intensity.

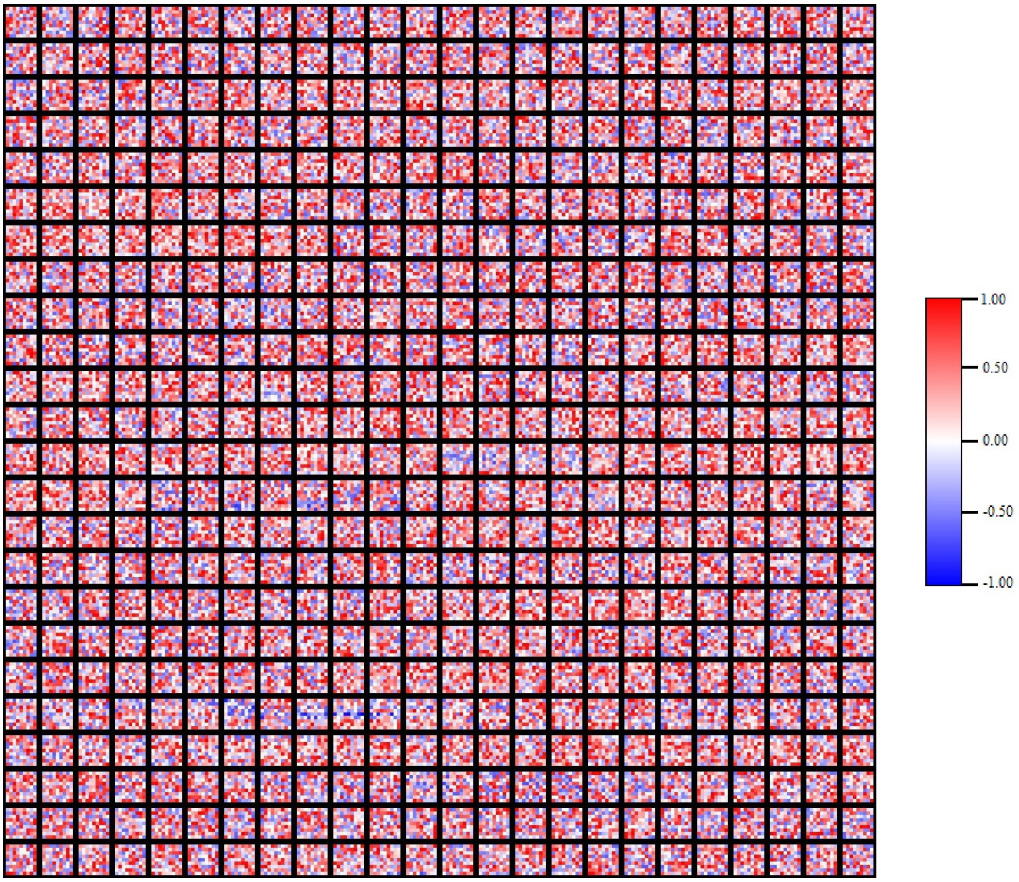

Figure 6: Scaled layer 1 filters for eight neighbor C-SRCNN on general images.

4.2 Medical Images

We now look at performance comparisons in the application of C-SRCNN models using a medical data set [16]. Large gains in performance are expected as we can exploit the self-similarity of medical images. Self-similarity allows us to efficiently use context as neighboring information is a predictor for the patch. We will use bicubic interpolation and SRCNN as benchmarks.

Our training set comes from DICOMLibrary with a variety of CT scans represented. From these, we take a random 10% for validation and 10% for testing. We use the same hyper parameters for each model including patch size of 32, stride of 16. From the dataset we derive over 240,000 patches with 24,000 as validation.

The test convergence curves are shown in Figure 7. We see that at the end of training we achieve a PSNR of **32.49** on the testing set. Compared to both SRCNN and bicubic interpolation, C-SRCNN models deliver outstanding performance. Thus, contextual information is critical to the models' performance. The data is summarized in Tables 3 and 4 with other interpolation methods. Some sample images are shown in Figures 8 and 9.

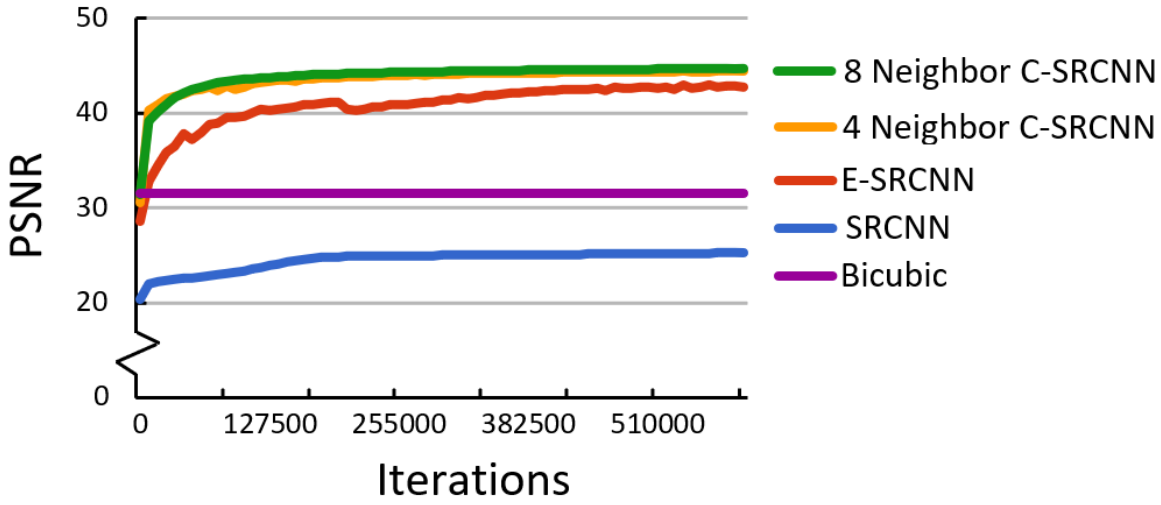

Figure 7: Training PSNR(dB) over epochs.

Dataset	Nearest Neighbor	Bicubic	SRCNN	E-SRCNN	4 Neighbor C-SRCNN	8 Neighbor C-SRCNN
DICOM Library	29.094	29.784	30.673	29.643	32.367	32.493

Table 3: Testing PSNR(dB) for CT datasets.

Dataset	Nearest Neighbor	Bicubic	SRCNN	E-SRCNN	4 Neighbor C-SRCNN	8 Neighbor C-SRCNN
DICOM Library	0.8913	0.8994	0.9312	0.9245	0.9441	0.9471

Table 4: Testing SSIM for CT datasets.

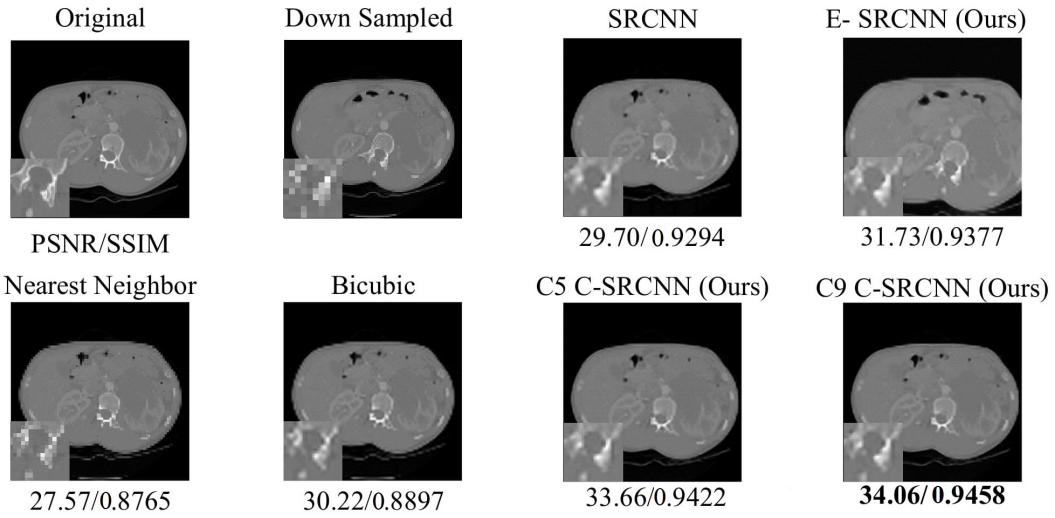

Figure 8: Example of CT image testing.

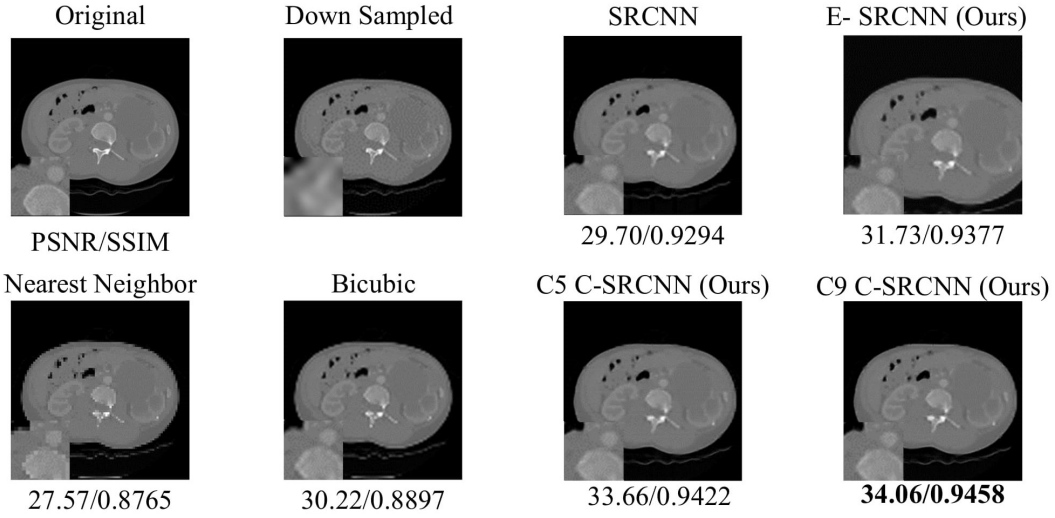

Figure 9: Example of CT image testing.

4.2.1 Learned Weights

We present all 64 nine-channel first layer filters as 576 images in Figure 10. We see that each filter learns a specific function. Some filters may represent Gaussian transforms, other specialize in edge or texture detection. The feature maps of the first layer contain mostly structural information such as edges and second layer features differ based on intensity.

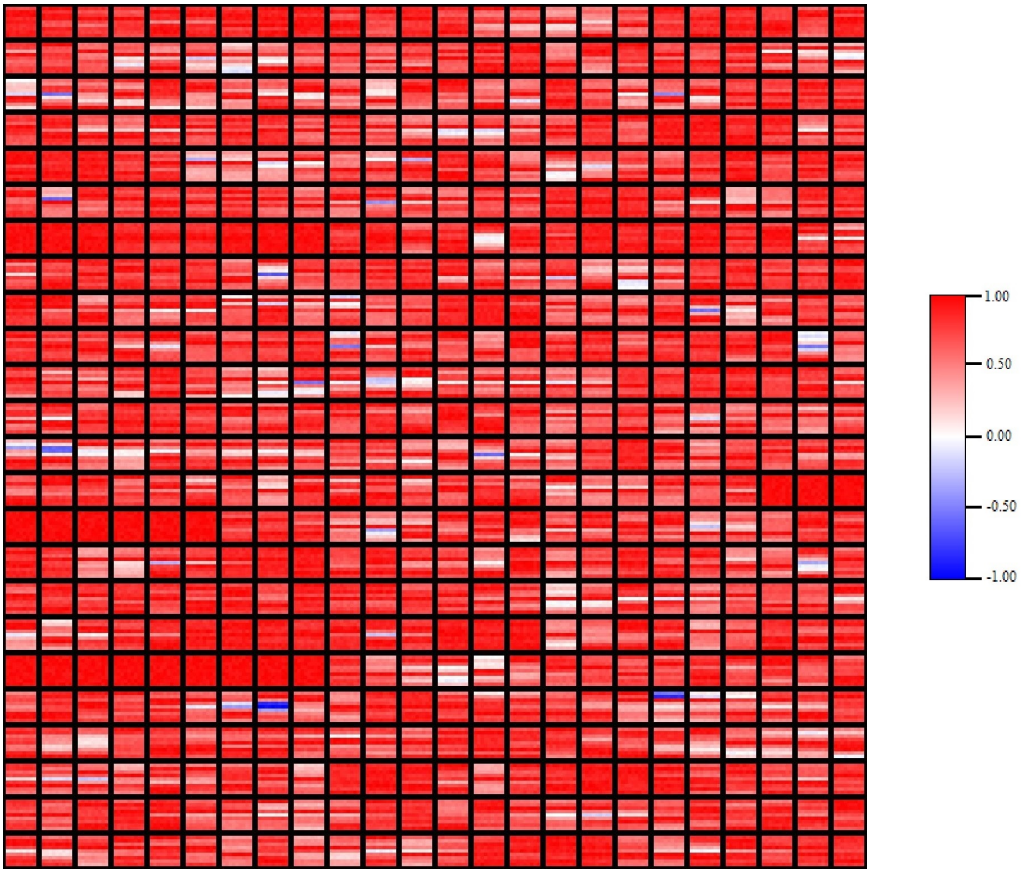

Figure 10: Scaled layer 1 filters for eight neighbor C-SRCNN on medical images.

4.3 Radiologist Evaluations

In a survey aiming to evaluate the real-world applications of our model, we randomized the order of images generated by many interpolation methods and asked for radiologists to rank the images based on usefulness. Aggregated results had C-SRCNN place first and SRCNN far behind. Subjective comments stated that generated by C-SRCNN have greater contrast, sharper margins, and less noise. Other models received comments such as: "The left kidney demonstrates hydronephrosis but has noise within the collecting system" indicating excess noise generated can greatly hinder patient treatment. The survey results are summarize in Table 5.

		Ranking		
Radiologist	Image Set	Bicubic	SRCNN	C-SRCNN
1	Average	1.75	2.50	1.75
	1	1	3	2
	2	3	1	2
	3	2	3	1
	4	1	3	2
2	Average	1.50	2.50	2.00
	1	1	3	2
	2	3	2	1
	3	1	2	3
	4	1	3	2
3	Average	2.00	2.50	1.50
	1	1	3	2
	2	3	2	1
	3	3	2	1
	4	1	3	2
4	Average	3.00	1.50	1.50
	1	3	1	2
	2	3	2	1
	3	3	2	1
	4	3	1	2
Overall		2.0625	2.2500	1.6875

Table 5: Summary of Radiologist Evaluations.

4.4 Summary

Testing PSNR/SSIM is shown in Tables 6 and 7. We can see that for each dataset. C-SRCNN models generally outperform both SRCNN and traditional techniques. We see that the addition of context has a large impact in the performance of our model. Looking at context-aware models, we find that the extended patch model displays large amounts of inconsistency. This can be explained by the fact that we are supplying the model with contextual information but not in a format that is suggestive to the model. In contrast, the multi-channel input allows the model to learn the inter-patch relationships between contextual patches and the central patch, providing for consistently better performance compared to existing models.

Dataset	Nearest Neighbor	Bicubic	SRCNN	E-SRCNN	4 Neighbor C-SRCNN	8 Neighbor C-SRCNN
DICOM Library	29.094	29.784	30.673	29.643	32.367	32.493
Set 5	25.033	26.902	29.910	28.642	30.038	30.253
Set 14	22.275	24.152	25.319	22.130	25.160	25.481

Table 6: Testing PSNR(dB) comparison for various models and image sets.

Dataset	Nearest Neighbor	Bicubic	SRCNN	E-SRCNN	4 Neighbor C-SRCNN	8 Neighbor C-SRCNN
DICOM Library	0.8913	0.8994	0.9312	0.9245	0.9441	0.9471
Set 5	0.7637	0.7928	0.8791	0.8760	0.8786	0.8811
Set 14	0.6816	0.6948	0.7948	0.7709	0.7947	0.7953

Table 7: Testing SSIM comparison for various models and image sets.

5 Conclusion

We have presented a context-aware deep learning approach for single image super resolution, C-SRCNN, which takes low-resolution patches and their context as input and generates high-resolution and realistic patches as output. The proposed C-SRCNN utilizes the structure self-similarity of human body in medical images and enlarges receptive field by taking the contextual information into the model.

Our approach outperforms baseline interpolation methods and another deep learning framework SRCNN on both photo-like datasets and medical image datasets when evaluated with the widely used measures of PSNR and SSIM. Using extensive MOS testing , we have confirmed that CASRCNN reconstructions for a large upscaling factor ($3\times$) are more realistic and clinically useful than reconstructions obtained with the other compared methods.

Recently, very deep convolutional networks have led to breakthroughs of state-of-the-art single image super resolution approaches such as VDSR [17], SRGAN [18], DBPN [19] and EDSR [20]. In the future, we will integrate the proposed context-aware approach with these very deep architectures and we conjecture that additional performance can be further gained.

References

[1] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 2, pp. 295–307, 2016.

[2] H. Greenspan, "Super-resolution in medical imaging," *The Computer Journal*, vol. 52, no. 1, pp. 43–63, 2008.

[3] K. J. Murphy, J. A. Brunberg, and R. H. Cohan, "Adverse reactions to gadolinium contrast media: a review of 36 cases.," *AJR. American journal of roentgenology*, vol. 167, no. 4, pp. 847–849, 1996.

[4] A. J. Einstein, "Medical imaging: the radiation issue," *Nature Reviews Cardiology*, vol. 6, no. 6, p. 436, 2009.

[5] P. Milanfar, *Super-resolution imaging*. CRC press, 2010.

[6] H. Chang, D.-Y. Yeung, and Y. Xiong, "Super-resolution through neighbor embedding," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. I–I, IEEE, 2004.

[7] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *IEEE Conference on Computer Vision and Pattern Recognition)*, 2015.

[8] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *International conference on curves and surfaces*, pp. 711–730, Springer, 2010.

[9] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 1794–1801, IEEE, 2009.

[10] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

[11] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[12] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[13] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga,

S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[14] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *European Conference on Computer Vision*, pp. 391–407, Springer, 2016.

[15] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," 2012.

[16] S. C. Park, M. K. Park, and M. G. Kang, "Super-resolution image reconstruction: a technical overview," *IEEE signal processing magazine*, vol. 20, no. 3, pp. 21–36, 2003.

[17] J. Kim, J. Kwon Lee, and K. Mu Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1646–1654, 2016.

[18] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," *arXiv preprint*, 2016.

[19] M. Haris, G. Shakhnarovich, and N. Ukita, "Deep back-projection networks for super-resolution," *arXiv preprint arXiv:1803.02735*, 2018.

[20] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, vol. 1, p. 3, 2017.