Single image super-resolution for medical imaging
Final Project Report

Ioana-Sabina Stoian

École Normale Supérieure Paris-Saclay
ioana-sabina.stoian@ens-paris-saclay.fr

Abstract. Acquiring high-quality Magnetic Resonance Images (MRI) is time-consuming and expensive. The ability to reconstruct the high-resolution (HR) MRI image from its low resolution (LR) counterpart might be a very efficient solution for diminishing the cost of this procedure and for reducing the time spent with every patient [7].
The task of reconstructing the HR image from a single LR image is called single image super-resolution (SISR). CARN (Convolutional anchored regression network) [5] is a recently proposed deep learning (DL) architecture for SISR, which achieves state of the art tradeoff between computational efficiency and SR accuracy. In this report, we describe our application of CARN to a medical imaging super-resolution task, as well as enhancements to make it even more computationally efficient for medical imaging (MI) SR, without significant loss in accuracy.
First, we replace the layer with the highest computational complexity from CARN - a convolutional layer with 3x3 filters - with a convolutional layer with 1x1 filters. We also replace the L2-regularized convolutional layers from CARN with the convolutional layers with normalized filters proposed in [11], which allows us to train deeper networks (with up to 13 layers). In addition, we intersperse feature extraction layers with regression layers. The proposed architecture obtains better accuracy / computational complexity tradeoff on a knee MRI SR task [12]. This work is an extension of my Bachelor's thesis project, which was performed under the supervision of Lecturer Radu Timofte (ETH Zürich) and Professor Florin Leon (Gheorghe Asachi Technical University of Iaşi) at Gheorghe Asachi Technical University of Iaşi in 2019.

Keywords: Single Image Super-Resolution · Medical Imaging · Convolutional Neural Networks

1 Introduction

The process of acquiring Magnetic Resonance Images (MRI) of very high quality is a time-consuming and expensive procedure. The ability to reconstruct the high-resolution (HR) MRI image from its low resolution (LR) counterpart might be a very efficient solution for diminishing the cost of the whole operation and

for reducing the time spent with every patient. More precisely, the acquisition of low resolution images might be sufficient for a correct diagnosis (by using a deep learning approach that transforms the LR image into a HR one). Moreover, in the medical field, where the images can easily be affected by different geometrical deformations (e. g. the motion of the patient during an MRI procedure) or noise, image enhancement tools can be extremely helpful for doctors in order to make a correct diagnosis.

The task of reconstructing the HR image from a single LR image is called single image super-resolution (SISR). The focus of this project is to compare various promising Convolutional Neural Network (CNN) approaches to medical imaging for the SISR task and to compare their performance using multiple metrics: accuracy (using PSNR and SSIM), inference time and number of trainable parameters. Computational efficiency can be an especially efficient metric in practice, potentially allowing improved access to high-quality medical imaging for more people for a reduced cost. SR performed on less expensive devices might allow for faster and cheaper medical imaging procedures and might be even more important for lower quality, cheaper MRI devices.

For these reasons, we focus mainly on the CARN architecture [5], a recent CNN which achieves an excellent tradeoff between accuracy and computational efficiency, and also propose and test modifications to make CARN even more efficient. CARN is also efficient to run on CPU, so it has potential to allow for more democratized access to medical imaging (by being run on less expensive devices).

2 Problem Definition

Single Image Super-Resolution (SISR) is the task of reconstructing a high-resolution (HR) image from its low-resolution (LR) counterpart, in particular by reconstructing the high-frequency details. The difficulty of this task stems from the fact that many different HR images could result in the same LR image after downscaling through e.g. bicubic interpolation. In this work, all the LR images are generated from the corresponding HR images through bicubic interpolation with downscaling factor 4x.

In this work, we train CNNs end-to-end through gradient-based optimization to minimize the pixel-wise MSE loss between the ground-truth HR image and the CNN's reconstruction from the LR image. The metrics we will use to measure the quality of the reconstructed images are the well-known PSNR (Peak signal-to-noise ratio) and SSIM (Structural similarity index).

3 Related Work

Recently, deep networks and, especially, convolutional networks, have dominated SISR on several metrics, including accuracy or computational efficiency. Computational efficiency is particularly important to allow CNNs to be run on less

expensive devices where there might be strict limitations on the required memory or energy (battery) consumption. For example, at the PIRM 2018 Perceptual Image Enhancement on Smartphones competition, SR track, all the competitors have proposed CNNs [2].

Proposed in [1], SRCNN is the first notable CNN architecture that achieved state-of-the-art results for super-resolution. Differently from previous methods, SRCNN reduces the need for pre-processing and post-processing, simplifying the architecture of super-resolution systems. Unlike the previous methods consisting of complex pipelines with multiple components trained in different ways and for different objectives), SRCNN is trained end-to-end. Compared to more recent architectures, SRCNN is quite shallow, containing only three layers. We use a SRCNN variant as the simplest DL baseline in our experiments.

Convolutional Anchored Regression Network (CARN) [5], the convolutional architecture we modify, is inspired by the ANR (Anchored Neighborhood Regression) [8] and A+ [9] local linear regression methods. In contrast to them, CARN does not use hand-crafted features, but is an end-to-end CNN (all operations are converted to convolutions so that extracted features, anchors and regressors are learned together and not optimized separately). In [5], CARN was compared to other high-performance CNNs, including SRCNN [1] and SRResNet [4], and achieved the best compromise between speed and accuracy. This ability to reach a state of the art compromise between speed and accuracy was also proven by the 3rd place obtained by CARN at the PIRM 2018 SR challenge of Perceptual Image Enhancement on Smartphones.

WDSR [11] is a very recent CNN architecture that won the NTIRE 2018 Challenge on Single Image Super-Resolution [10] competition. The authors of [11] showed that WDSR obtains PSNRs similar to other CNNs, with lower or similar computational complexity. WDSR proposes a few architectural changes that lead to improved performance. Firstly, WDSR increases the number of feature maps before the activation function. At the same time, in order to preserve computational efficiency, WDSR factorizes the convolutional layer before the nonlinearity, managing to achieve a better accuracy and speed compromise than that obtained by other networks. In addition, proposes to use the convolution layer with standardized filters (an application of weight normalization introduced in [6]). This leads to gains in accuracy, especially for very deep networks, which become easier to train. WDSR [11] serves as inspiration for our proposed architectural modifications to CARN, described in the next section.

4 Methodology

In this section, we briefly describe the components of the CARN architecture [5], as well as our proposed modifications to it and the theoretical motivation behind our proposal.

4.1 CARN

The main components of the CARN architecture are the regression blocks, displayed in Fig. 1.

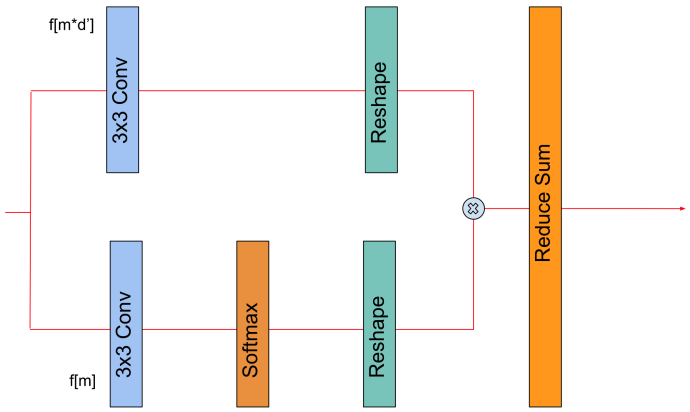

Fig. 1: Original CARN regression block.

We use the following notation: W = feature map width; H = feature map height; m = number of anchor points; d'= number of internal channels; w = convolutional filter width; h = convolutional filter height.

As shown in Fig. 1, the upper branch of the regression block contains a convolutional layer with c input feature maps and $m \times d'$ output feature maps of shape $W \times H$ and filters of size $w \times h$. It's easy to notice that the number of output feature maps can easily become very large (e.g. in [5], CARN uses, by default, $d' = 16$ and $m = 16$, leading to 256 output feature maps), which makes this convolutional layer very expensive. This is the regression branch of CARN's regression block.

Similarly, the bottom branch of Fig. 1, adapted from [5], contains a convolutional layer with filters of size $w \times h$ and m output feature maps. This operation corresponds to the anchor points branch of the regression block in CARN.

Finally, the two branches are combined. We will resize the output feature maps from the top branch as a 4-dimensional tensor, of shape $W \times H \times m \times d'$. We will also reshape the output feature maps from the bottom branch as a 4-dimensional tensor, of shape $W \times H \times m \times 1$. We then multiply element wise the two tensors (using broadcasting on the last dimension) and then the elements on the third dimension (reduce sum), as shown in Fig. 1. All the details and complete derivation are presented in [5].

The intuition behind is that the regression block maps features from a low-dimensional space (before the regression block) to a high-dimensional space (the

regressor branch, with large number of output channels) and then reduces the output back to the low-dimensional space by using the anchor point branch (which produces a weighted sum of the regressors).

The full CARN architecture is illustrated in Fig. 2, adapted from [5]. In CARN, in contrast to our proposal, the convolutional feature extraction layers are completely separate from the regression blocks and appear only as the first layers of the architecture.

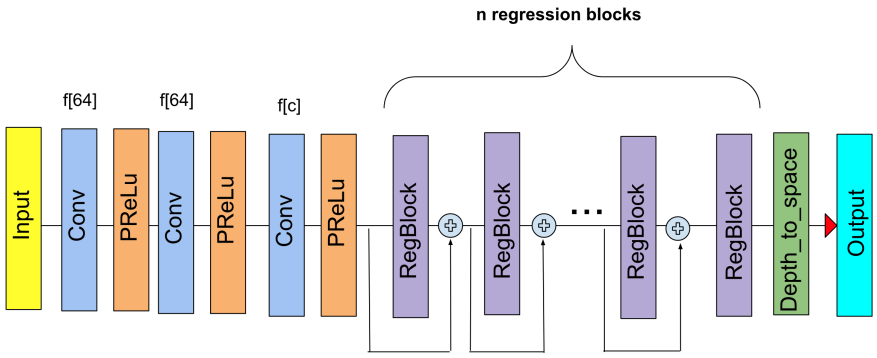

Fig. 2: Original CARN architecture.

4.2 Our proposed CARN modifications

The WDSR architecture and its success in the NTIRE 2018 competition have inspired the novel modifications we propose for the CARN architecture. More precisely, we also use a convolutional layer with 1x1 filters, which can be interpreted as factorizing the convolutional layer with 3x3 filters used in the initial CARN proposal, in order to increase computational efficiency. We use this modification for the layer with the highest computational complexity in CARN, the convolutional layer on the regression branch of the regression block. The new architecture of the regression block is illustrated in Fig. 3. In addition, we replace all the convolutional layers with L2 regularization from CARN with convolutional layers with normalized filters, which allows us to opt out of L2 regularization and to train deeper networks (up to 13 layers). We further add in each regression block a linear convolutional layer with 3x3 filters, which we interpret as providing new features for the next regression block. Beyond the modifications for the regression blocks, we also change the order of the feature extraction layers. In CARN, all the feature extraction layers came before all the regression blocks. In our proposal, the extraction of features and, respectively, the construction of regressors and anchors, are interspersed, the intuition behind this architectural choice being to provide each layer of regressors and anchors with

newly extracted features. The modified CARN architecture that we described above is presented in Fig. 4.

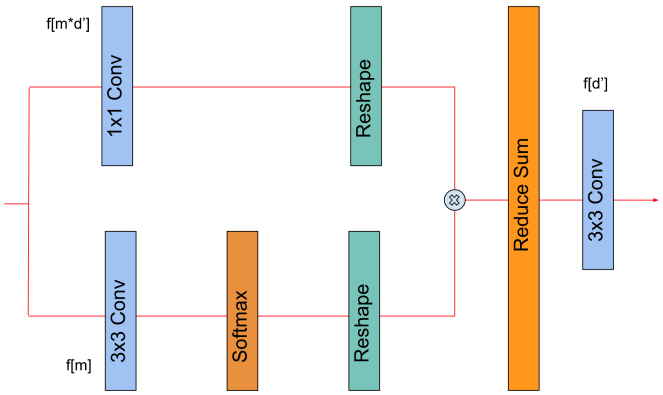

Fig. 3: Modified CARN regression block architecture (our proposal).

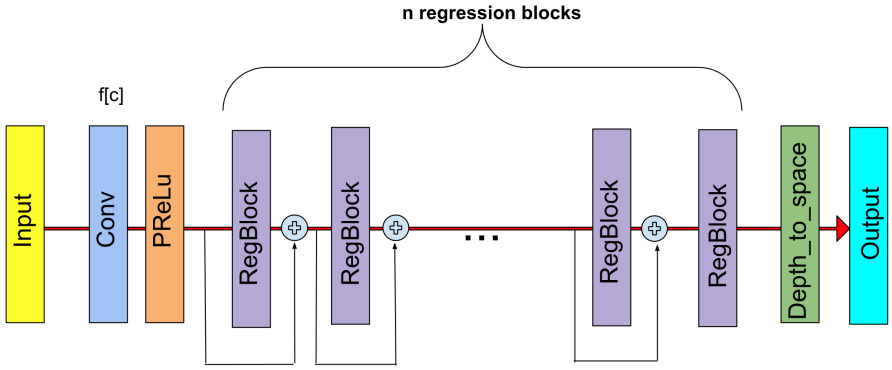

Fig. 4: Modified CARN architecture.

4.3 Theoretical motivation for the proposed CARN modifications

We consider the following notation:

$$H_{LR} = \text{LR image height}$$

W_{LR} = LR image width

numAnchors = number of achors in the regression block

numInnerChannels = number of channels in the regression block

f = filter height = filter width

For the original CARN regression block, the number of operations for the regressor branch is:

$$\text{N\_OPS\_Regressors} = H_{LR} \cdot W_{LR} \cdot f \cdot f \cdot numInnerChannels^2 \cdot numAnchors \quad (1)$$

For the anchor points branch, the number of operations is:

$$\text{N\_OPS\_Anchors} = H_{LR} \cdot W_{LR} \cdot f \cdot f \cdot numInnerChannels \cdot numAnchors \quad (2)$$

Due to the fact that the number of operations on the regressor branch is $numInnerChannels$ times greater than the number of operations on the anchor points branch, it is important to reduce this cost, because this is how the highest computational savings can be obtained. The costs of the other operations within the regression blocks can be neglected, because they are much lower compared to the costs of the convolution from the regressor branch and the convolution from the anchor points branch.

In our proposed regression block, the convolution on the regressor branch uses the following number of operations:

$$\text{N\_OPS\_Regressors} = H_{LR} \cdot W_{LR} \cdot numInnerChannels^2 \cdot numAnchors \quad (3)$$

For example, replacing a convolutional layer with a 3x3 filter with a convolutional layer with a 1x1 filter results in a decrease of the number of operations (FLOPS), as well as in the number of trainable parameters, by approximately 9 times. Considering that the convolutional layers on the regressor branch within the regression blocks consume the largest number of operations, our more efficient proposal leads to a significant decrease in the total number of operations performed by the network.

The convolution on the anchor points branch consumes the same number of operations as in the case of the original architecture.

The feature extraction layer uses the number of operations computed in Eq. (4).

$$\text{N\_OPS\_FE} = H_{LR} \cdot W_{LR} \cdot f \cdot f \cdot numInnerChannels^2 \quad (4)$$

The number of operations of the convolution layer for feature extraction in our proposal is $numAnchors$ times smaller than the one on the regressor branch of the original proposal, so the total number of operations for our entire proposed regression block remains much smaller than that for the original one.

Because the computational cost of the regression blocks is generally significantly larger than the cost of the feature extraction layers, the cost of both the original architecture and our architecture is roughly proportional to the cost of the regression blocks. Therefore, by diminishing this cost, we considerably reduce the overall computational complexity of the architecture. In addition,the total number of trainable parameters will decrease significantly.

5 Evaluation

5.1 Dataset

The dataset we have used is FastMRI [12] with Knee single-coil MRIs. Because the dataset is very large (the training set of the knee single-coil MRIs is about 88 GB), we only used a subset of it. More precisely, our training set consists of the first 10 volumes of data (rss files) from the Knee single-coil dataset. Each volume contains a different number of slices and we consider each slice as a grayscale image. In total, we have 361 grayscale images (of size 320x320), so 361 examples in the training set. For evaluation, we have used 2 other volumes of this type from the same dataset (Knee single-coil).

In addition, we have used data augmentation methods during training. At every epoch, we extract a random crop (96x96) from every image from the training set. Each crop is then flipped vertically and horizontally, with 0.5 probability each.

5.2 Results

For evaluation, we have used the conventional accuracy metrics for the SISR task: Peak Signal-to-Noise Ratio (PSNR) and SSIM (Structural Similarity Index). Moreover, we have assessed the running time for every CNN, as well as the number of trainable parameters. We have compared the results that we achieve with multiple SRCNN, CARN and new CARN (our proposal) architectures (with different hyperparameters), and bicubic interpolation as a simple baseline. Beyond the original SRCNN proposal, we also compare variants with fixed numbers of output channels (16, 32 or 64) for each of the 3 convolutional layers.

The results are displayed in Table 1, with the best PSNR, SSIM, number of trainable parameters and inference time bolded. We can observe that, generally speaking, the New CARN systems with 8 inner channels and 8 anchor points seem to obtain the best compromise between PSNR, SSIM, number of trainable parameters and inference time. New CARN-13-8-8 obtains the best PSNR and the best SSIM of all the methods, even if it contains less than a quarter of the number of trainable parameters of some other systems and its running time can also be 4x lower. While SRCNN with 16 channels has the fewest trainable parameters, we can see that the SRCNN systems are much more expensive in inference time and they don't seem to scale as well. The original CARN proposal, while inexpensive in inference time (compared to SRCNN), requires many more

Table 1: Comparison between SRCNN, CARN and New CARN (our proposal) on the test set, for upscaling factor x4. Legend: [Model (CARN – initial proposal / New CARN – our proposal)]-[number of regression blocks]-[number of inner channels]-[number of anchor points].

Model	PSNR (dB)	SSIM	Trainable parameters	Inference time (s)
CARN-1-8-8	30.11	0.8374	67432	0.0042
New CARN-1-8-8	29.96	0.8346	10336	**0.0023**
CARN-3-8-8	30.07	0.8365	72688	0.0060
New CARN-3-8-8	30.13	0.8377	15592	0.0045
CARN-5-8-8	30.05	0.8363	83200	0.0102
New CARN-5-8-8	30.17	0.8383	26104	0.0060
CARN-9-8-8	30.07	0.8365	104224	0.0192
New CARN-9-8-8	30.19	0.8388	47128	0.0105
New CARN-13-8-8	**30.20**	**0.8389**	68152	0.0146
CARN-1-16-16	30.09	0.8371	165264	0.0060
New CARN-1-16-16	29.91	0.8336	28032	0.0047
CARN-3-16-16	30.09	0.8371	204704	0.0161
New CARN-3-16-16	29.93	0.8340	67472	0.0109
CARN-5-16-16	30.07	0.8366	283584	0.0256
New CARN-5-16-16	29.99	0.8350	146352	0.0200
CARN-9-16-16	30.06	0.8365	441344	0.0453
New CARN-9-16-16	30.00	0.8352	304112	0.0315
New CARN-13-16-16	29.97	0.8346	461872	0.0459
SRCNN 16 channels	30.01	0.8354	**8129**	0.0130
SRCNN 32 channels	30.04	0.8359	29057	0.0273
SRCNN 64 channels	30.07	0.8365	109313	0.0765
SRCNN original	30.04	0.8361	57281	0.0414
Bicubic	29.628	0.825	-	-

parameters and doesn't achieve as good accuracy as new CARN. Curiously, neither CARN nor new CARN seemed to benefit from 16 inner channels and 16 anchors on this task.

6 Discussion

In this work, we have first benchmarked CARN, a recently proposed convolutional architecture with excellent accuracy / computational complexity tradeoff, for a Knee single-coil MI SR task [12]. This excellent compromise makes CARN very suitable to be run on less expensive devices, as demonstrated by it achieving 3rd place in the PIRM 2018 SR challenge of Perceptual Image Enhancement on Smartphones.

However, as we have shown in this paper, CARN can be made even more efficient in terms of computational complexity and number of trainable parameters. Our proposed changes have significantly decreased the number of trainable pa-

rameters, without significant degradation of the quality of the generated images, and with a slight decrease in the execution time on CPU.

Due to time and computational constraints, our benchmarking has, for the time being, only focused on a single dataset. Some potential ideas for future work include:

1. testing on more SR MI datasets;
2. using extra metrics for computing the SR accuracy, such as MOS (Mean Opinion Score)
3. using GANs to improve the perceptual quality, similarly to SRGAN [4].
4. adapting the architecture, so that a single network can perform SR for multiple different upscaling factors, as achieved by VDSR [3].

References

1. Dong, C., Loy, C.C., He, K., Tang, X.: Image Super-Resolution Using Deep Convolutional Networks. vol. 38, p. 295–307. Institute of Electrical and Electronics Engineers (IEEE) (Feb 2016), http://dx.doi.org/10.1109/TPAMI.2015.2439281
2. Ignatov, A., Timofte, R., Van Vu, T., Luu, T.M., Pham, T.X., Van Nguyen, C., Kim, Y., Choi, J.S., Kim, M., Huang, J., et al.: PIRM Challenge on Perceptual Image Enhancement on Smartphones: Report. p. 315–333. Springer International Publishing (2019), http://dx.doi.org/10.1007/978-3-030-11021-5_20
3. Kim, J., Lee, J.K., Lee, K.M.: Accurate image super-resolution using very deep convolutional networks. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Jun 2016). https://doi.org/10.1109/cvpr.2016.182, http://dx.doi.org/10.1109/CVPR.2016.182
4. Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Jul 2017). https://doi.org/10.1109/cvpr.2017.19, http://dx.doi.org/10.1109/CVPR.2017.19
5. Li, Y., Agustsson, E., Gu, S., Timofte, R., Van Gool, L.: Carn: Convolutional anchored regression network for fast and accurate single image super-resolution. In: The European Conference on Computer Vision (ECCV) Workshops (September 2018)
6. Salimans, T., Kingma, D.P.: Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural networks (2016)
7. Sood, R., Topiwala, B., Choutagunta, K., Sood, R., Rusu, M.: An application of generative adversarial networks for super resolution medical imaging. 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA) (Dec 2018). https://doi.org/10.1109/icmla.2018.00055, http://dx.doi.org/10.1109/ICMLA.2018.00055
8. Timofte, R., De, V., Gool, L.V.: Anchored neighborhood regression for fast example-based super-resolution. In: 2013 IEEE International Conference on Computer Vision. pp. 1920–1927 (2013)
9. Timofte, R., De Smet, V., Van Gool, L.: A+: Adjusted anchored neighborhood regression for fast super-resolution. vol. 9006, pp. 111–126 (04 2015). https://doi.org/10.1007/978-3-319-16817-3_8

10. Timofte, R., Gu, S., Van Gool, L., Zhang, L., Yang, M.H.: Ntire 2018 challenge on single image super-resolution: Methods and results. pp. 965–96511 (06 2018). https://doi.org/10.1109/CVPRW.2018.00130
11. Yu, J., Fan, Y., Yang, J., Xu, N., Wang, Z., Wang, X., Huang, T.: Wide activation for efficient and accurate image super-resolution (2018)
12. Zbontar, J., Knoll, F., Sriram, A., Muckley, M.J., Bruno, M., Defazio, A., Parente, M., Geras, K.J., Katsnelson, J., Chandarana, H., Zhang, Z., Drozdzal, M., Romero, A., Rabbat, M., Vincent, P., Pinkerton, J., Wang, D., Yakubova, N., Owens, E., Zitnick, C.L., Recht, M.P., Sodickson, D.K., Lui, Y.W.: fastMRI: An open dataset and benchmarks for accelerated MRI (2018)