Rethinking the CSC Model for Natural Images
236862 Final Project

Adir Rahamim

March 2020

1 Introduction

This paper introduces a new way to represent natural images using Convolution Sparse Coding(CSC) model. The CSC model has shown great success in several natural image processing tasks, such as image separation [1], Image fusion [2], super resolution [3] and more, however, two common properties darken this great success, first the CSC model is used only as a complementary component, modeling only the texture part of the image, second they assume noiseless images, and indeed when it comes to basic image restoration task, such as denoising, it falls behind patch averaging based methods. The paper discusses the reasons why the CSC model fails in image denoising task, and suggests new concept of the CSC model to bring state of the art results in image denoising, while using much fewer learned parameters.

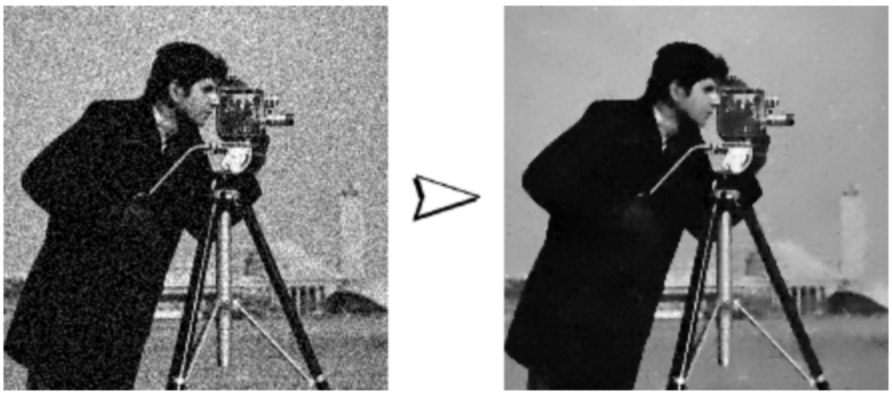

Figure 1: Image denoising task

2 Background - Convolution Sparse Coding

The basic spare representation model is about finding the sparsest vector, $\alpha \in \mathbb{R}^M$, that best represents a given vector, $X \in \mathbb{R}^N$,using a given dictionary $D \in \mathbb{R}^{N\times M}$, s.t $X = D\alpha$, which can be formulated as the following optimization problem - P_0^ϵ:

$$\min_{\alpha} \|\alpha\|_0 \ s.t. \ \|D\alpha - X\| \leq \epsilon$$

This problem in known to be NP hard, however there are many approximation methods, such as OMP, BP and many more, that under several conditions guaranteed to find the optimal α.

The CSC model can be the derived from the classic spare representation model by substituting matrix multiplication with the convolution operator - assume that a signal $X \in \mathbb{R}^N$ is a sum of m convolutions of sparse feature maps $\{Z_i\}_{i=1}^m$ and band limited filters $\{d_i\}_{i=1}^m$, thus the CSC model refers to solving the following optimization problem:

$$\min_{\{Z_i\}_{i=1}^m} \sum_{i=1}^m \|Z_i\|_0 \; s.t. \; X = \sum_{i=1}^m d_i * Z_i$$

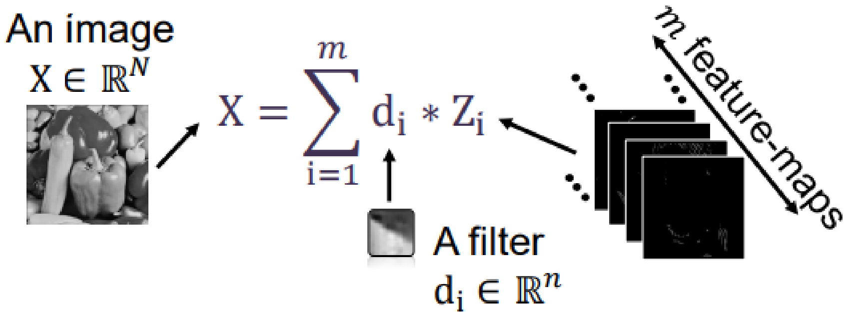

Figure 2: The CSC model as convolution

The CSC model can be written as matrix multiplication form as follows: By interlacing all feature maps into column vector $\Gamma \in \mathbb{R}^{Nm}$, and by defining $C_i \in \mathbb{R}^{n\times m}$ as a local dictionary - a banded circulant matrix, containing a single atom d_i with all its shifts , then the CSC model can also be formulated as:

$$X = \sum_{i=1}^m C_i \Gamma_i = [C_1 ... C_m] \begin{bmatrix} \Gamma_1 \\ \vdots \\ \Gamma_m \end{bmatrix} = D\Gamma$$

with the following optimization problem:

$$\min_{\Gamma} \|\Gamma\|_0 \; s.t. \; \|D\Gamma - X\| \leq \epsilon$$

For further needs(proof that PA is an approximation to the CSC MMSE estimator), the author uses [6] to introduce new definition to the CSC model. The sparse representation vector Γ can be thought of N concatenated vectors $\alpha_i \in \mathbb{R}^m$,term needles, each describes the contribution of the m filters when aligned to the i-th element in X, i.e

$$X = D\Gamma = \sum_{i=1}^N P_i^T D_L \alpha_i = \sum_{i=1}^N P_i^T s_i$$

where we defined $s_i = D_L \alpha_i$ and P_i^T places the $D_L \alpha_i$ in the i-th location in the constructed image.

CSC model gain great success in several image processing tasks, leading to superior results over patch-based algorithms. However, in all those great applications there are two points need to be mentioned - the input image is first separated into smooth and non-smooth images and the CSC only models the detail reach of the image, another assumption in those recent applications is that the input image is noiseless, and indeed applying CSC on noisy images leads to disappointing results, far behind patch based algorithms, as can be seen in [4, 5], this is true for any image processing task where noise cannot be neglected, such as image deblurring.

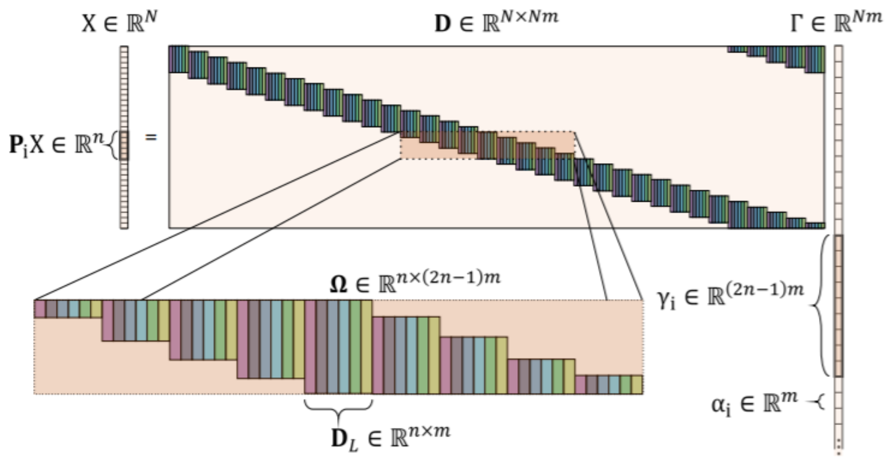

Figure 3: The CSC model as matrix multiplication

Next we will discuss two reasons the writer gives for the poor performance(compared to patch based algorithms) of the CSC model in image denoising:

1. Poor Coherence - In [6] it has been shown that the stability guarantee for the CSC problem is the condition: $\|\Gamma\|_{0,\infty} < \frac{1}{2}\left(1 + \frac{1}{\mu(D)}\right)$, where $\|\Gamma\|_{0,\infty}$ is the maximum number of local non-zero elements in $\|\Gamma\|$, and $\mu(D)$ is the mutual coherence of the dictionary, thus we need D to have low mutual coherence in order to allow a large number of active filters in the signal while keeping the constraint .However natural images usually do not obey this constraint - on one hand, the characteristic of natural images is mainly consists by smooth and piecewise smooth regions, thus the dictionary contains smooth and piecewise smooth filter in order to enable sparse representation. On the other hand, having smooth or piecewise smooth filters in the dictionary leads to highly correlated filters in the global dictionary, thus the mutual coherence of D is big, these two facts contradict each other and therefore the above constraint not met.

2. Bayesian explanation - The author proofs that as oppose to a convolution pursuit(being MAP estimator, as shown in [7]), patch averaging performs an approximation to the CSC MMSE estimator w.r.t the entire image, thus brings superior results in MSE terms. First the author formalize the PA approach: PA obtains an estimate for each image patch and average the overlaps regions, thus under a local dictionary D_L we solve N independent optimization problems using BP in

its Lagrangian form:

$$\left\{ \hat{\alpha}_i = \arg\min_{\alpha_i} \lambda_i \left\| \alpha_i \right\|_1 + \frac{1}{2} \left\| D_L \alpha_i - P_i Y \right\|_2^2 \right\}_{i=1}^{N}$$

The final image obtained by placing the clean patches in their place in the image while averaging overlaps regions : $\hat{X} = \frac{1}{n} \sum_{i=1}^{N} P_i^T D_L \alpha_i$.

The MMSE estimator of the CSC model is a weighted sum of all possible supports weighted by their prior probability:

$$\hat{\Gamma}_{MMSE} = \mathbb{E}\left\{ \Gamma \mid Y \right\} = \mathbb{E}_S \left\{ \mathbb{E}\left\{ \Gamma \mid Y, S \right\} \right\} = \sum_{s \in \theta} P\left(S\right) \hat{\Gamma}_S$$

S stands for the support of Γ , $P\left(S\right)$ is the prior probability of such a support and θ is the set of all possible supports. Computing this sum is impractical, thus the author approximates it - assume the case where $D\Gamma$ results in non-overlapping tangent patches, there are n such arrangements uphold this assumptions differing only in the initial shift k. Lets denote the solution for each such shift as Γ_k, which can be represented using convolution strided dictionary, D_k, and the first non-zero needle in Γ is located in the k-th index $1 \leq k \leq n$, the solution for each shift can be done using BP in its Lagrangian form:

$$\hat{\Gamma}_k = \arg\min_{\Gamma} \lambda \left\| \Gamma \right\|_1 + \frac{1}{2} \left\| D_k \Gamma - Y \right\|_2^2$$

$$= \arg\min_{\Gamma = \{\alpha_k; \alpha_{k+n}; \ldots\}} \sum_{i=0}^{\frac{N}{n}-1} \left\{ \lambda \left\| \alpha_{k+in} \right\|_1 + \frac{1}{2} \left\| D_k \alpha_{k+in} - P_{k+in} Y \right\|_2^2 \right\}$$

each optimization problem can be solved separately:

$$\alpha_{k+in} = \arg\min_{\alpha} \lambda \left\| \alpha \right\|_1 + \frac{1}{2} \left\| D_k \alpha - P_{k+in} Y \right\|_2^2$$

looking back to the MMSE term we got:

$$\hat{\Gamma}_{MMSE} = \sum_{i=1}^{n} P\left(i\right) \hat{\Gamma}_i \approx \frac{1}{n} \sum_{i=1}^{n} \hat{\Gamma}_i$$

If we also assume that all the prior probability are equal, thus we got same optimization problem we solved for patch averaging, thus PA is approximation of CSC MMSE, and this explain its superior result in MSE term.

3 The Model

3.1 Generalizing the MMSE Approximation

The author suggests to generalize the non-overlapping slices assumption and allow for a smaller stride. In patch averaging we use stride size equals to the size of the filters themselves, $q = n$, thus the filters do not overlap, in the standard CSC we use stride size $q = 1$, the author suggests to use stride size $1 \leq q < n$. By using stride size between 1 to n we can achieve two improvements:

1. When $q < n$ each estimate allowing for overlapping slices, implying that the pursuit must be done globally on all the involved slices together, this necessarily leads to a global agreement between those slices, as we take them into account in the pursuit, as opposed to PA ($q = n$), when each patch(slice) is estimated separately.

2. When q is sufficiently large, low mutual coherence of the global dictionary can be preserved even for natural images, in contrast to the standard CSC pursuit($q = 1$), since the filters only partially overlap.

In order to justify those claims, the author performs a denoising experiment on images from the Set12 dataset contaminated with white Gaussian noise with standard deviation $\sigma \in \{15, 25, 50, 75\}$. He used both PA($q = 11$), and the proposed strided CSC using various strides $1 \leq q < n = 11$, where $q = 1$ is the standard CSC, followed by an averaging operation.

	CSC - stride size (q)									
σ	1	2	3	4	5	6	7	8	9	10	PA
15	28.99	29.27	30.01	30.66	31.06	31.21	31.31	31.39	31.45	**31.46**	31.23
25	25.78	26.11	26.94	27.72	28.26	28.50	28.64	28.75	28.84	**28.88**	28.73
50	21.49	22.11	23.17	23.83	24.52	24.86	25.05	25.29	25.47	**25.56**	25.32
75	18.83	19.58	20.95	21.81	22.43	22.75	22.97	23.25	23.51	**23.66**	23.28

Table 1: Denoising results(PSNR) using PA and CSC. CSC Results surpass PA are marked in blue. Best results are bold

As expected from the 2 claims, the best results achieved when the stride is large(to allow small mutual coherence), but smaller than the filter size(to allow overlaps between filters).

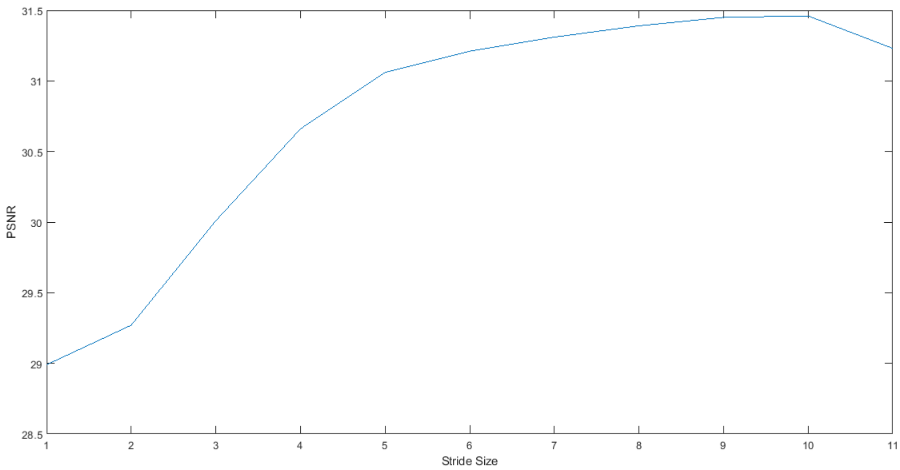

Figure 4: Stride size VS PSNR ($\sigma = 15$)

3.2 CSCNet - a Supervised Denoising Model

In this section the author uses an improvement for the popular BP optimization solver - the iteratively ISTA algorithm, defined as:

$$\Gamma_{k+1} = \mathcal{S}_{\frac{\lambda}{c}}\left(\Gamma_k + \frac{1}{c}D^T\left(Y - D\Gamma_k\right)\right)$$

where $c > \sigma_{max}\left(D^T D\right)$ - $\sigma\left(\cdot\right)$ stands for singular value, and $\mathcal{S}_\tau$ is the soft-thresholding operator extended to work element-wise. The author mentions the fact that the converges of the ISTA algorithm is pretty slow, thus he uses two improvements from [8, 9] in order to overcome this burden. In [9] they introduced the LISTA algorithm, Which approximate the sparse coding process by learning the parameters of non-linear recurrent encoder that follows L iterations of the iterative process, [8] has extended this to the convolution settings as follows:

$$\Gamma_{k+1} = \mathcal{S}_\tau\left(\Gamma_k + \frac{1}{c}A\left(Y - B\Gamma_k\right)\right)$$

And the estimated clean image obtained by:

$$\hat{X} = C\Gamma_L$$

where A stands for the convolution operator, and B, C are deconvolution. The matrices A, B and C and the threshold vector τ are learned in the supervised manner. The author mentions an important fact of this model - the number of learned parameters does not grow with L, the number of iterations.

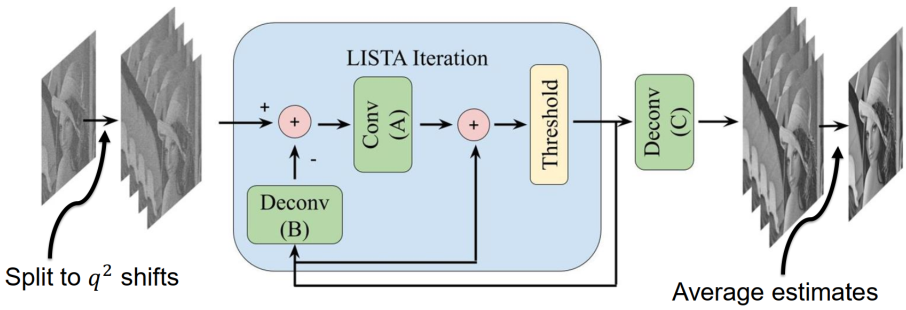

Figure 5: The CSCNet architecture

The author combines this sparse coding approximation with the MMSE approximation from the previous part - the learned matrices will use a strided convolution structure using a constant stride q, he further expend it - in order to obtain estimate for each possible shift the input image is duplicated q^2 times, than the output image is the average of the estimations.

4 The Author Experiments

The author trained his model using clean images from the Waterloo Exploration Dataset and 432 BSD images, the noisy inputs were obtained by adding white Gaussian noise with constant standard deviation σ, he trained 4 models, one for different noise levels $\sigma \in \{15, 25, 50, 75\}$. For each model he learned 175 filters of size 11×11, set stride size $q = 8$ and $L = 12$. To learn the model parameters he employ the ADAM optimizer and minimize the ℓ_2 loss. The learning rate he used was 10^{-4} and decrease it by a factor of 0.7 every 50 epochs and iterate over 250 epochs.

The performance of the algorithm was tested on the BSD68 dataset that was not in the training set.

Table 2: Denoising performance (PSNR) on the BSD68 dataset.

σ	BM3D	WNNM	TNRD	MLP	DnCNN	FFDNet	CSCNet
15	31.07	31.37	31.42	–	31.72	31.63	31.57
25	28.57	28.83	28.92	28.96	29.22	29.19	29.11
50	25.62	25.87	25.97	26.03	26.23	26.29	26.24
75	24.21	24.40	–	24.59	24.64	24.79	24.77

Table 2: Denoising performance (PSNR) comparison

The proposed model outperforms BM3D, TNRD, WNNM and MLP, while being on par with DnCNN and FFDNet, however the author mentions big advantages of the proposed CCNet over the two leading methods:

As it has been shown, the number of parameters of the proposed model does not grow with the depth of the model(L), thus it uses much fewer parameters compared to the two other leading methods, as he shows in the next table:

Model	First layer	Last layer	Mid layers	Total
DnCNN	$3 \times 3 \times 1 \times 64$	$3 \times 3 \times 64 \times 1$	$(3 \times 3 \times 64 \times 64 + 128) \times 15$	556,032
FFDNet	$3 \times 3 \times 5 \times 64$	$3 \times 3 \times 64 \times 4$	$(3 \times 3 \times 64 \times 64 + 128) \times 13$	486,080
CSCNet	–	$11 \times 11 \times 175 \times 1$	$(11 \times 11 \times 175 \times 1) \times 2 + 175$	63,700

Table 3: Comparison of number of parameters

The author also tested the affect of different stride size values, for that he trained 6 models, each one with different stride value and constant noise level $\sigma = 25$, and he got similar results to table 1 - setting the stride too high results in independent patch based processing, and setting $q = 1$ leads to standard CSC with high correlated atoms, best results are for $q = 7$ or $q = 8$.

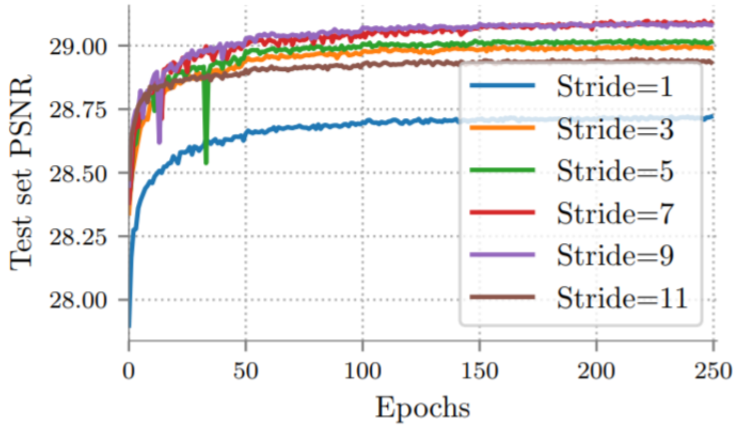

Figure 6: CSCNet error for different stride size

Last experiment the author did was with color image denoising, with similar experiments as before. The test was performed on the color BSD68 dataset. The CSCNet results are on par with the two leading algorithm:

σ	CBM3D	CDnCNN	FFDNet	CSCNet
15	33.52	33.89	33.87	33.83
25	30.71	31.23	31.21	31.18
50	27.38	27.92	27.96	28.00
75	25.74	24.47	26.24	26.32

Table 4: Denoising performance on color dataset

5 Contribution

This section will include three sub-sections: paper review, limitation and consistency check and improvements.

5.1 Paper Review

I credit this paper for the following achievements:

1. The paper gives deep explanations to the reasons why the CSC model gives weaker results than patch based algorithms in one of the most basic image restorations task - image denoising.

2. The paper offers a Bayesian connection between the CSC model and patch-based methods - showing that local patch based algorithms are actually the MMSE approximation of the global CSC model, explaining the superior results of patch based algorithms over CSC based algorithms in MSE term.

3. Propose new MMSE approximation method for the CSC model that provide big improvement and superior results compared to patch based algorithms.

4. The paper introduced the CSCNet - a supervised method based directly on the CSC model, that achieved state of the art results(on par with the two other leading supervised methods) while using much fewer parameters($\sim 90\%$ lower).

5. The author have done a great job in investigating the problem and the reasons it failed so far, and incorporating techniques from various fields and papers to achieve a big improvement.

5.2 Limitations

While working on paper I paid attention to the next facts:

1. There is no explanation about how determine the optimal stride size q, except of trail and error. As mentioned in the paper, a small stride leads to highly correlated filters reducing the performance of the model. On the other extreme, when the stride is too large, the performance reduces due to the lack of sufficient overlap between the filter, thus choosing the optimal q can lead to significant change in results, as can be seen in figure 5.

2. The paper lacks a comparison of run time and training time of the algorithm, as can be found in different papers, for example [8].

3. There is no explanation about how to determine the optimal depth of the network, i.e. the number of LISTA iterations used, in terms of run time VS performance. We can expect that a deeper network leads to higher PSNR results, however also leads to higher training time. I think the author should have discussed this manner and mention whether the improvement saturate at a certain depth and how to choose the optimal depth.

4. There is no reference to the number of epochs. In the author experiment he trained the model for 250 epochs, and demonstrate its denoising performance on the BSD68 test set as can be seen in Figure 5. From the figure, it seems that it is possible to improve the denoising results further, i.e achieve higher PSNR values, by running more epochs. Therefore, I think the author should have mention if training the program for more than 250 epochs can achieve better results, and what is the optimal number of epochs in terms of run time VS performance.

5.3 Consistency Check

I checked the performance of the model on several more test images, to check its consistency.

First, I checked denoising performance of the proposed method, alongside other leading denoising algorithms on the Set12 dataset. We can clearly see that the method is indeed on par with current leading methods:

Image	BM3D	WNNM	MLP	TNRD	DnCNN	DDFNet	CSCNet
$\sigma = 15$							
C.man	31.91	32.17	-	32.19	32.61	32.42	32.27
House	34.93	35.13	-	34.53	34.97	35.01	34.79
Peppers	32.69	32.99	-	33.04	33.30	33.10	33.20
Starfish	31.14	31.82	-	31.75	32.20	32.02	31.87
Monarch	31.85	32.71	-	32.56	33.09	32.77	32.73
Airplane	31.07	31.39	-	31.46	31.70	31.58	31.53
Parrot	31.37	31.62	-	31.63	31.83	31.77	31.86
Lena	34.26	34.27	-	34.24	34.62	34.63	34.45
Barbara	33.10	33.60	-	32.13	32.64	32.50	32.22
Boat	32.13	32.27	-	32.14	32.42	32.35	32.29
Man	31.92	32.11	-	32.23	23.46	32.40	32.31
Couple	32.10	32.17	-	32.11	32.47	32.45	32.32
Average	**32.37**	**32.70**	-	**32.50**	**32.86**	**32.75**	**32.65**
$\sigma = 25$							
C.man	29.45	29.64	29.61	29.72	30.18	30.06	29.99
House	32.85	33.22	32.56	32.53	33.06	33.27	32.92
Peppers	30.16	30.42	30.30	30.57	30.87	30.79	30.76
Starfish	28.56	29.03	28.82	29.02	29.41	29.33	29.08
Monarch	29.25	29.84	29.61	29.85	30.28	30.14	30.14
Airplane	28.42	28.69	28.82	28.88	29.13	29.05	29.04
Parrot	28.93	29.15	29.25	29.18	29.43	29.43	29.33
Lena	32.07	32.24	32.25	32.00	32.44	32.59	32.38
Barbara	30.71	31.24	29.54	29.41	30.00	29.98	29.33
Boat	29.90	30.03	29.97	29.91	30.21	30.23	30.11
Man	29.61	29.76	29.88	29.87	30.01	30.01	29.95
Couple	29.71	29.82	29.73	29.71	30.12	30.18	30.02
Average	**29.97**	**30.26**	**30.03**	**30.06**	**30.43**	**30.43**	**30.26**
$\sigma = 50$							
C.man	26.13	26.45	26.37	26.62	27.03	27.03	26.87
House	29.69	30.33	29.64	29.48	30.00	30.43	30.12
Peppers	26.68	26.95	26.68	27.10	27.32	27.43	27.37
Starfish	25.04	25.44	25.43	25.42	25.70	25.77	25.25
Monarch	25.82	26.32	26.26	26.31	26.78	26.88	26.63
Airplane	25.10	25.42	25.56	25.59	25.87	25.90	25.86
Parrot	25.90	26.14	26.12	26.16	26.48	25.58	26.45
Lena	29.05	29.25	29.32	28.93	29.39	29.68	29.46
Barbara	27.22	27.79	25.24	25.70	26.22	26.48	25.73
Boat	26.78	26.97	27.03	26.94	27.20	27.32	27.19
Man	26.81	26.94	27.06	26.98	27.24	27.30	27.19
Couple	26.46	26.64	26.67	26.50	26.90	27.07	27.04
Average	**26.72**	**27.05**	**26.78**	**26.81**	**27.18**	**27.32**	**27.10**
$\sigma = 75$							
C.man	24.32	24.60	24.63	-	25.07	25.29	25.09
House	27.51	28.24	27.78	-	27.85	28.43	28.21
Peppers	24.73	24.96	24.88	-	25.17	25.39	25.48
Starfish	23.27	23.49	23.57	-	23.64	23.82	23.47
Monarch	23.91	24.31	24.40	-	24.71	24.99	24.71
Airplane	23.48	23.74	23.87	-	24.03	24.18	24.20
Parrot	24.18	24.43	24.55	-	24.71	24.94	24.70
Lena	27.25	27.54	27.68	-	27.54	27.97	27.91
Barbara	25.12	25.81	23.39	-	23.63	24.24	23.71
Boat	25.12	25.29	25.44	-	25.47	25.64	25.58
Man	25.32	25.42	25.59	-	25.64	25.75	25.70
Couple	24.70	24.86	25.02	-	24.97	25.29	25.27
Average	**24.91**	**25.23**	**25.07**	**-**	**25.20**	**25.49**	**25.33**

Then I compared the performance of the model on BSD68 dataset:

σ	BM3D	WNNM	TNRD	MLP	DnCNN	DDFNet	CSCNet
15	31.07	31.37	31.42	-	31.72	31.63	31.57
25	28.57	28.83	28.92	28.96	29.22	29.19	29.11
50	25.62	25.87	25.97	26.03	26.23	26.29	26.24
75	24.21	24.40	-	24.49	24.64	24.79	24.77

Table 5: Denoising performance(average in PSNR) on BSD68 dataset

I got the same results as in the paper.

5.4 Improvements

The author suggests several further improvements to the model:

1. Addition of batch-normalization.
2. Use of multi-scale architecture.
3. Adoption of a local error constraint as in PA.
4. exploiting self similarity.

I decided to test other improvement that was not mentioned by the author:

Instead of training only one stride size(in the author experiment he trained the model only for q=8) I decided to train simultaneity the three best performance stride sizes, and on the output of them do weighed average, where the weights are also trained in supervised manner, accordingly the updated architecture:

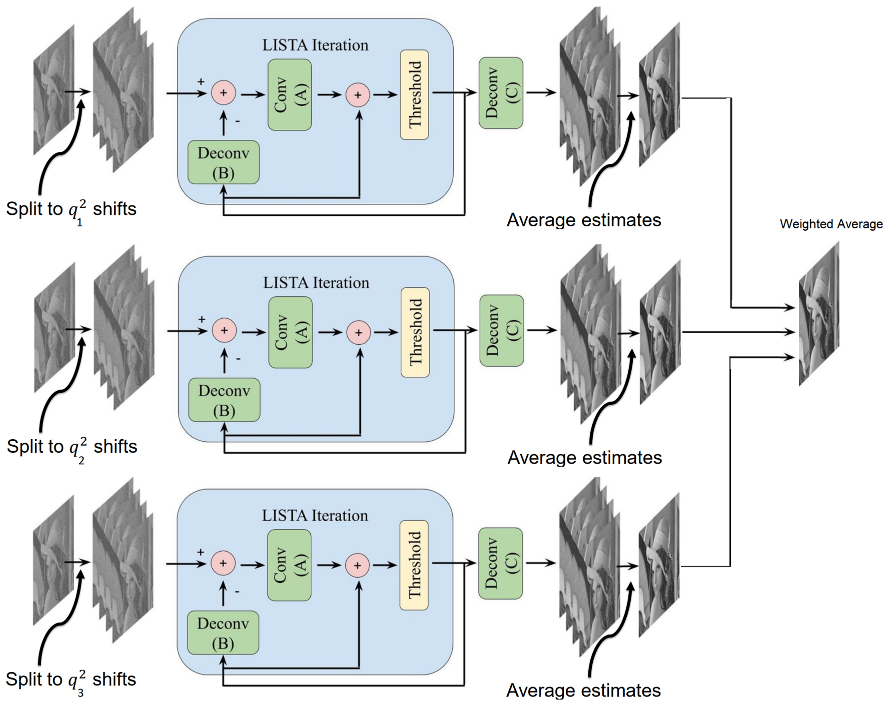

Figure 7: The improved CSCNet architecture

We can expect the improved architecture will lead to better results because we take into account different stride sizes, by that we train the network on more data and thus get better learning.

I trained the model on the same datasets as the paper - Waterloo Exploration and 432 images from BSD and using the exact same parameters as the paper(besides the stride size), i.e the noisy inputs were obtained by adding white Gaussian noise with constant standard deviation σ, I trained 4 models, one for different noise levels $\sigma \in \{15, 25, 50, 75\}$, for each model I learned 175 filters of size 11×11, and set $L = 12$. The model parameters were learned using the ADAM optimizer and minimize the ℓ_2 loss. The learning rate was 10^{-4} and decrease it by a factor of 0.7 every 50 epochs and iterate over 250 epochs.

Table 6 presents the results of the improved algorithm compared to regular CSCNet and other leading methods on the BSD68 dataset:

σ	BM3D	WNNM	TNRD	MLP	DnCNN	FFDNet	CSCNet	CSCNet-Imp.
15	31.07	31.37	31.42	-	31.72	31.63	31.57	31.62
25	28.57	28.83	28.92	29.96	29.22	29.19	29.11	29.17
50	25.62	25.87	25.97	26.03	26.23	26.29	26.24	26.29
75	24.21	24.40	-	24.59	24.64	24.79	24.77	24.83

Table 6: Denoising performance(PSNR) of improved architecture on BSD68 dataset

As we can see the improve architecture did gave us an slight improvement of $0.05 - 0.06$ dB compared to regular CSCNet.

Table 7 presents the results of the improved algorithm on the Set12 dataset:

σ	BM3D	WNNM	TNRD	MLP	DnCNN	FFDNet	CSCNet	CSCNet-Imp.
15	32.37	32.70	32.50	-	32.86	32.75	32.69	32.79
25	29.97	30.26	30.06	30.03	30.43	30.43	30.26	30.37
50	26.72	27.05	26.81	26.78	27.18	27.32	27.13	27.28
75	24.91	25.23	-	25.07	25.20	25.49	25.41	25.53

Table 7: Denoising performance(PSNR) of improved architecture on Set12 dataset

On the Set12 dataset we got improvement of $0.1 - 0.15$ dB compared to regular CSCNet.

References

[1] I.rey-Otero, J. Sulam, and M. Elad, "Variations on the csc model," arXiv preprint arXiv:1810.01169, 2018.

[2] Y. Liu, X. Chen, R. K. Ward, and Z. J. Wang, "Image fusion with convolutional sparse representation," IEEE Signal Processing Letters, vol. 23, no. 12, pp. 1882–1886, 2016.

[3] S. Gu, W. Zuo, Q. Xie, D. Meng, X. Feng, and L. Zhang, "Convolutional sparse coding for image super-resolution," in International Conference on Computer Vision (ICCV), pp. 1823–1831, IEEE, 2015.

[4] D. Carrera, G. Boracchi, A. Foi, and B. Wohlberg, "Sparse overcomplete denoising: aggregation versus global optimization," IEEE Signal Processing Letters, vol. 24, no. 10, pp. 1468–1472, 2017.

[5] B. Wohlberg, "Convolutional sparse coding with overlapping group norms," arXiv preprint arXiv:1708.09038, 2017.

[6] V. Papyan, J. Sulam, and M. Elad, "Working locally thinking globally: Theoretical guarantees for convolutional sparse coding," IEEE Transactions on Signal Processing, vol. 65, no. 21, pp. 5687–5701, 2017.

[7] D. Simon, J. Sulam, Y. Romano, Y. M. Lu, and M. Elad, "Mmse approximation for sparse coding algorithms using stochastic resonance," IEEE Transactions on Signal Processing, vol. 67, no. 17, pp. 4597–4610, 2019.

[8] H. Sreter and R. Giryes, "Learned convolutional sparse coding," in International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2191–2195, IEEE, 2018.

[9] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in International Conference on Machine Learning (ICML), pp. 399–406, Omnipress, 2010.