COMP.SGN.300 Project Report: Learning Deep Convolutional Networks for Demosaicing

Minh Tran

April 2023

Contents

1 Introduction

1.1 Project requirements

This is the report for the project work in the course COMP.SGN.300: Advanced Image Processing. The project requirement is to study a research work in the field of image processing, compare the results of the work with other related studies, and carry out experiments to reproduce the results. The paper chosen in this report is *Learning Deep Convolutional Networks for Demosaicing* [1].

1.2 Demosaicing

Modern camera sensors record the image in the form of color filter arrays (CFAs), which means each pixel of the sensor is sensitive to only a certain color. This results in the output of the camera sensor being a mosaic of pixels with different colors. Post processing is necessary to recover the missing color information and obtain the real image.

This process of recovering the real image from the mosaic output of the camera is called *demosaicing*. This process can be mathematics based with different transformations, or learning based with deep neural networks. The paper investigates the usage of deep neural networks for demosaicing.

2 The content of the paper

This section discusses the main points of the paper, including the network architecture used in the paper and the notable comparison to other demosaicing methods listed in the paper.

2.1 Proposed network architectures for demosaicing

The paper proposes 2 architectures for demosaicing. All the architectures are convolutional neural networks. The first architecture (DMCNN) is shallower with 3 convolutional blocks, and the second architecture (DMCNN-VD) is deeper with 20 convolutional blocks. The input of both networks are a 33×33 image patch extracted from the mosaic image. Add table for the architecture from the paper.

2.1.1 Convolutional network for demosaicing

DMCNN contains 3 convolutional blocks. The blocks have 128 convolutional filters of size 9×9, 64 convolutional filters of size 1×1, and 3 convolutional filters of size 5×5, respectively. The blocks also include ReLU activation function and batch normalization.

The model is optimized using the MSE loss between the output of the network and the target true image.

2.1.2 Very deep convolutional network for demosaicing

DMCNN contains 20 convolutional blocks. The first 19 blocks have 64 convolutional filters of size 3×3. The last block have 3 convolutional filters of size 3×3 to reconstruct the 3 color channels. The output of the convolutional kernels are zero padded to maintain the original image size. The blocks have SELU activation function and batch normalization. There is a residual connection between the input image and the output image: the input mosaic image is filled by bilinear interpolation on the individual color channels, and the network estimates the difference between the interpolation result with the target image.

The model is optimized using the MSE loss between the residual output of the network and the target real image.

2.2 Results of the method from the paper and comparison with other proposed demosaicing methods

The paper did a very thorough analysis of the performance of their networks (DMCNN, DMCNN-VD) with other demosaicing methods. The reason for choosing these different demosaicing systems are unclear. Nevertheless, the authors have successfully proven the superiority of DMCNN-VD.

2.2.1 Performance of DMCNN-VD against other methods on demosaicing Bayer CFA

The authors evaluate the performance of the demosaicing methods using peak signal-to-noise ratio (PSNR) between the output image of the model and the target real image.

The Kodak dataset and MacMaster dataset [2] are used for testing the performance of the models. The PSNR values are calculated for each individual color channel and for the whole image. The models are evaluated 3 times: on each dataset individually and on the combination of 2 datasets.

TABLE II in [1] shows the result of DMCNN and DMCNN-VD against other 10 approaches: SA [3], SSD [4], NLS [5], CS [6], ECC [7], RI [8], MLRI [9], ARI [10], PAMD [11], and AICC [12]. The result shows that DMCNN-VD tops all other methods in all criteria with higher PSNR values. In most cases, the PSNR values from DMCNN-VD is above 40. DMCNN performs worse since it is a shallower network, but the result is very comparable to other methods.

Table III in [1] shows the performance of DMCNN-VD against 2 other deep demosaicing methods: SIGGRAPH Asia [13] and ICME [14]. DMCNN-VD has a slightly better performance comparing to the other systems, but the difference is very minimal. The convolutional kernel size also has effect on the performance of DMCNN-VD.

2.2.2 Performance of DMCNN-VD against other methods on linear space

In reality, the raw images from camera sensor are captured in the linear space of radiance, and the demosaicing process happens on that space inside the camera [1]. TABLE V in [1] shows the result of DMCNN-VD with ARI [10] and RTF [15] on clean data in the linear space. DMCNN-VD outperforms other methods by 2 db.

TABLE VI shows the result of the different demosaicing methods on noise data in linear space. Here the authors fine-tune the DMCNN-VD architecture to yield DMCNN-VD-Tr to work with noisy data. DMCNN-VD already outperforms other methods in noisy data as well, and DMCNN-VD-Tr beats DMCNN-VD in every criterion.

2.2.3 Experiment with other mosaic patterns

TABLE VII in [1] shows the performance of DMCNN-VD with different types of mosaic CFA: Bayer, Diagonal stripe, CYGM, and Hirakawa. DMCNN-VD proves its flexibility by achieving consistent results on every tested CFA pattern. Moreover, with some modifications to the model, DMCNN-VD can construct a new mosaic pattern that works best for the data. This model is called DMCNN-VD-Pa.

TABLE VIII in [1] shows the result of SVEC demosaicing of AP, DMCNN, and DMCNN-VD on 50 HDR images. DMCNN and DMCNN-VD outperforms AP in SVEC demosaicing, and DMCNN-VD has a significant improvement compared to DMCNN.

3 Experiment with DMCNN and DMCNN-VD

This section discuss my practical analysis on the paper. The work includes comparing the models from the paper with other suggested demosaicing topics, and testing out the architectures from the paper.

3.1 Comparison of DMCNN and DMCNN-VD with other demosaicing methods mentioned in the project

In the project list, the other demosaicing methods are [16], [3], [17], [18], [19], and [20]. Since the original paper does not compare the performance of DMCNN and DMCNN-VD against these works (except for [3]), I did such analysis and the result is presented in Table 1. The comparison includes as much testing statistics on Kodak and MacMaster datasets as possible.

In the list of demosaicing works given in the project, [3], [17], [18], [20] provide testing PSNR on the Kodak and MacMaster datasets. In [16], the performance is

Table 1: Comparison of DMCNN, DMCNN-VD and the other proposed demosaicing methds from the project.

Method	Kodak dataset				MacMaster dataset			
	RPSNR	GPSNR	BPSNR	CPSNR	RPSNR	GPSNR	BPSNR	CPSNR
[3]	39.80	43.31	39.50	40.54	32.73	34.73	32.1	32.98
[17]	40.22	43.07	39.62		36.67	40.17	35.41	
[18]	40.50	38.72	38.60					
[20]	42.20	45.47	41.29	42.57	39.29	42.44	37.49	39.17
DMCNN	39.86	42.97	39.18	40.37	36.50	39.34	35.21	36.62
DMCNN-VD	**43.28**	**46.10**	**41.99**	**43.45**	**39.69**	**42.53**	**37.76**	**39.45**

evaluated by the incresae in PSNR comparing to bilinear interpoloation. Instead No statistics for demosaicing is provided in [19].

DMCNN-VD has the best performance on every criterion. The GAN-based approach in [20] has very close results to DMCNN-VD: the differences are at most 1 db compared to DMCNN-VD. This implies that deep learning approaches tend to be very effective in demosaicing.

3.2 Re-implementing DMCNN and DMCNN-VD

In the paper, the authors create a new trainng set for the models. However, since the data and code is not provided, I need to create and adapt the datasets and models myself. [1]

3.2.1 Data

The training, validation and test set are created from the McM dataset. For each sample in the McM dataset, patches of size 33×33 are extracted. The patches are then augmented by rotating with 0, 90, 180 and 270 degrees. The mosaic patterns of the true images are created by only sampling either the red, green or blue values of the pixels. The training, validation and test set are divided from the processed image patch with the ratio of $60 : 20 : 20$.

I have tried the same data processing steps with the Kodak dataset, but since the resolution is very high (3072×2048), the result is too much data for the GPU to handle. Memory overflow constantly happens, so I decided to experiment only with the McM dataset. One possible solution to this is to downsize the Kodak dataset to 768×512, which can be a further improvement idea for the project.

3.2.2 The adaptation of DMCNN and DMCNN-VD

During the experiment, there are many difficulties in optimzing a very deep neural network like DMCNN-VD. The model is more prone to overfitting, and

[1] The code for the implementation is available at my GitHub repo

Table 2: The results of my implementation of DMCNN and DMCNN-VD.

Network	RPSNR	GPSNR	BPSNR	CPSNR
DMCNN	36.07	38.21	36.41	36.14
DMCNN-VD	32.62	38.53	31.80	33.07

the training process happens less stably. In the hidden convolutional layers of DMCNN-VD, dropout of 0.25 to avoid overfitting. Due to the time constraint, instead of initializing DMCNN-VD similar to the paper, I used He initialization instead. The rest of DMCNN-VD is similar to the architecture in the paper. The architecture of DMCNN is the same as described in the paper.

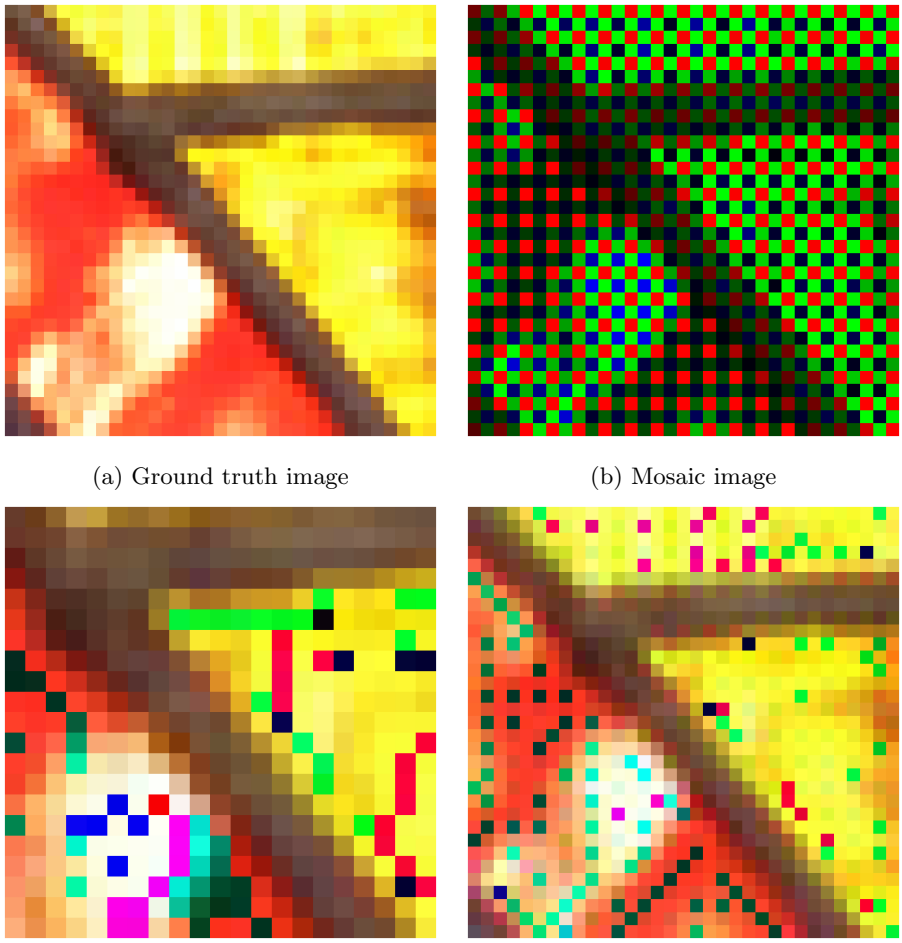
The models are trained using Adam [21] as the optimizer, with the learning rate of 0.001. Early stopping is used on the loss of the validation process, with the patience of 30. The implementation are developed using PyTorch with CUDA support. The hardware used for training is NVIDIA Tesla V100 from TUNI's Narvi cluster.

3.2.3 Results

Table 2 shows the result of my implementation of DMCNN and DMCNN-VD. The performance of DMCNN matches the figures presented in the paper. Disappointingly, the result of DMCNN-VD is much worse. DMCNN-VD witnesses a 3 db decrease comparing to DMCNN, and 10 db decrease comparing to the numbers in the paper.

Figure 1 illustrates one example of demosaicing result of DCMNN and DMCNN-VD from my implementation. This image seems to be hard for both DMCNN and DMCNN-VD with many colors and a lot of change in brightness. Both DMCNN and DMCNN-VD leave a lot of artifacts on the demosaiced image. Interestingly, the artifacts tend to follow the edges in the image. This phenomenon can also be observed in Figure 2, where artifacts from the output of DMCNN-VD concentrate around the curvy detail of the image. This time the image is much easier, with consistent shade of green and darkness. Two models perform much better this time, where DMCNN has no artifact and DMCNN-VD has much less artifact comparing to Figure 1.

Figure 3 shows the demosaicing results of DMCNN and DMCNN-VD on an image of Tampere by Visit Tampere The output of DMCNN looks strange since the demosaiced image is obtained by assembling the output patches together, and some information is lost because DMCNN outputs a patch of size 21×21. Both of these images have artifacts; however, the artifacts on the demosaiced image by DMCNN tends to be concentrated around an area, while the artifacts on the demosaiced image by DMCNN-VD spread evenly throughout the image. The output image by DMCNN-VD also has the problem of visible borders around the edges of the patches.



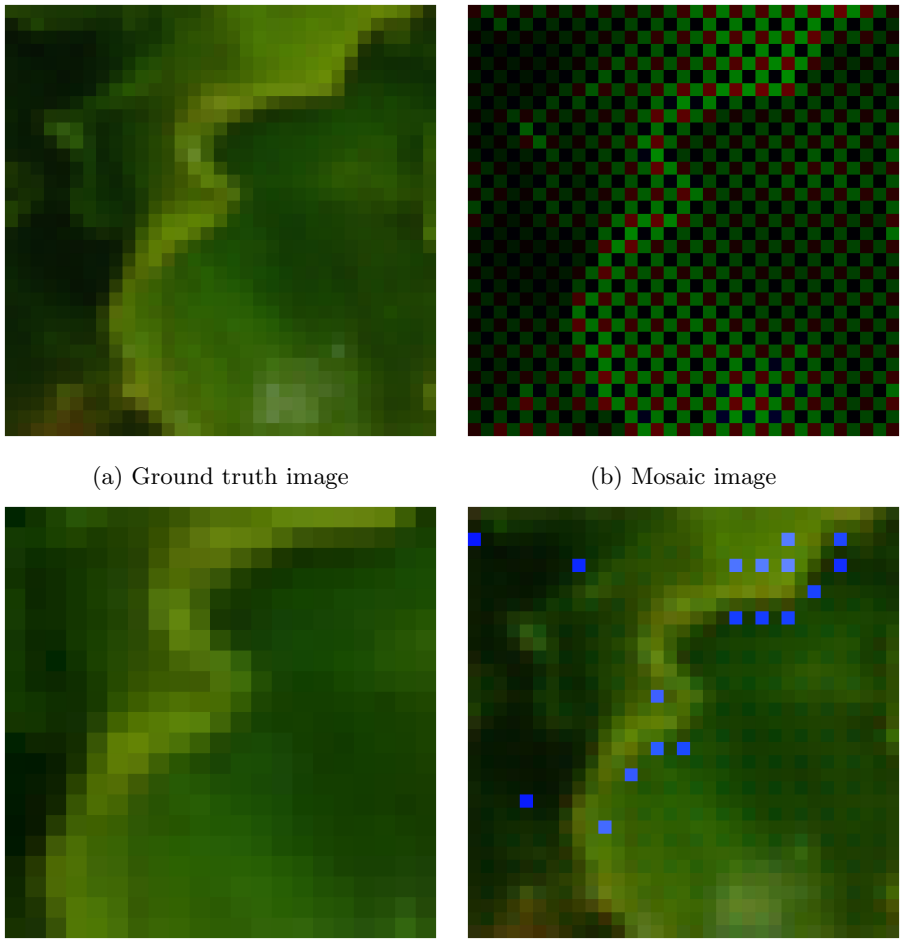
(a) Ground truth image

(b) Mosaic image

(c) Prediction from DMCNN

(d) Prediction from DMCNN-VD

Figure 1: Predictions of first test sample



(a) Ground truth image

(b) Mosaic image

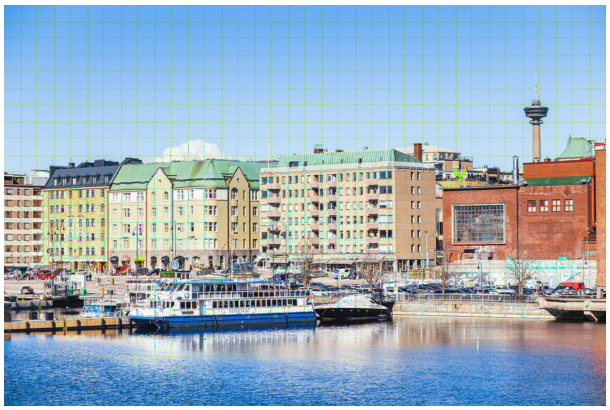
(c) Prediction from DMCNN

(d) Prediction from DMCNN-VD

Figure 2: Predictions of second test sample

(a) Ground truth image

(b) Output image by DMCNN



(c) Output image by DMCNN-VD

Figure 3: Demosaicing on a new image, both DMCNN and DMCNN-VD

3.2.4 Discussion

There are multiple possible reasons for lower results of DMCNN-VD. First of all, it might be more difficult to optimize a very deep neural network like DMCNN-VD. DMCNN is much shallower, therefore it is less sensitive to changes in optimization conditions and easier to replicate the results. Another problem is the dissimilar in some of the design choices. My implementation of DCMNN-VD does not have the similar weights initialization method, and since I am not too familiar with residual connection in neural networks, the bilinear interpolation and residual connection might have been implemented incorrectly. The different dataset used for training, validation and testing might also have a negative impact on trying to achieve the paper's scores.

All of this can be the possible further improvements to the project. The accuracy of the implementation of bilinear interpolation and residual connection in DMCNN-VD needs to be verified. More research into MSRA is needed to have the correct initialization process. Downsampling and extracting patches from Kodak dataset to collect more data is also another possible idea to improve the demosaicing system.

4 Conclusion

The paper *Learning Deep Convolutional Networks for Demosaicing* is an interesting and noteworthy work in the field of demosaicing. The deep convolutional network architecture proposed in the system outperforms numerous other works in the field. My own comparison with other demosaicing systems in the course project consolidates the demosaicing performance of DMCNN-VD.

Unfortunately, the implementation of DMCNN-VD is not successful. The problems might be lack of knowledge and skills leads to inaccuracy of the code script, the difficulty in optimizing a very deep neural network, and/or lack of official datasets. Further works to improve the system are also proposed according to these potential failures.

Overall, the project has been enjoyable and beneficial for myself. The project is a good chance to look into the problem of demosaicing, which happens very frequently in our handheld devices. It is very nice to learn multiple methods for demosaicing, from traditional mathematics to more recent deep learning approaches. The project is also an opportunity to explore and develop a new deep learning system from scratch. I got to practice image data creation and augmentation, creating the models, training and testing the models, and optimizing the system to try to achieve good results. Many skills of mine are improved after this project: problem solving and decision making, deep learning and image processing knowledge, and using Python and PyTorch for deep learning.

References

[1] N.-S. Syu, Y.-S. Chen, and Y.-Y. Chuang, *Learning deep convolutional networks for demosaicing*, 2018.

[2] L. Zhang, X. Wu, A. Buades, and X. Li, "Color demosaicking by local directional interpolation and nonlocal adaptive thresholding," *Journal of Electronic Imaging*, vol. 20, no. 2, p. 023 016, 2011.

[3] X. Li, "Demosaicing by successive approximation," *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 370–379, 2005.

[4] A. Buades, B. Coll, J.-M. Morel, and C. Sbert, "Self-similarity driven color demosaicking," *IEEE Transactions on Image Processing*, vol. 18, no. 6, pp. 1192–1202, 2009.

[5] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 2272–2279.

[6] P. Getreuer, "Image Demosaicking with Contour Stencils," *Image Processing On Line*, vol. 2, pp. 22–34, 2012.

[7] S. P. Jaiswal, O. C. Au, V. Jakhetiya, Y. Yuan, and H. Yang, "Exploitation of inter-color correlation for color image demosaicking," in *2014 IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 1812–1816.

[8] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, "Residual interpolation for color image demosaicking," in *2013 IEEE International Conference on Image Processing*, 2013, pp. 2304–2308.

[9] D. Kiku, Y. Monno, M. Tanaka, and M. Okutomi, "Minimized-Laplacian residual interpolation for color image demosaicking," in *Digital Photography X*, N. Sampat, R. Tezaur, S. Battiato, and B. A. Fowler, Eds., International Society for Optics and Photonics, vol. 9023, SPIE, 2014, p. 90230L.

[10] Y. Monno, D. Kiku, M. Tanaka, and M. Okutomi, "Adaptive residual interpolation for color and multispectral image demosaicking," *Sensors*, vol. 17, no. 12, 2017, ISSN: 1424-8220.

[11] M. Zhang and L. Tao, "A patch aware multiple dictionary framework for demosaicing," in *Computer Vision – ACCV 2014*, D. Cremers, I. Reid, H. Saito, and M.-H. Yang, Eds., Cham: Springer International Publishing, 2015, pp. 236–251, ISBN: 978-3-319-16811-1.

[12] J. Duran and A. Buades, "A Demosaicking Algorithm with Adaptive Inter-Channel Correlation," *Image Processing On Line*, vol. 5, pp. 311–327, 2015.

[13] M. Gharbi, G. Chaurasia, S. Paris, and F. Durand, "Deep joint demosaicking and denoising," *ACM Trans. Graph.*, vol. 35, no. 6, Dec. 2016, ISSN: 0730-0301.

[14] R. Tan, K. Zhang, W. Zuo, and L. Zhang, "Color image demosaicking via deep residual learning," in *2017 IEEE International Conference on Multimedia and Expo (ICME)*, IEEE, 2017, pp. 793–798.

[15] D. Khashabi, S. Nowozin, J. Jancsary, and A. W. Fitzgibbon, "Joint demosaicing and denoising via learned nonparametric random fields," *IEEE Transactions on Image Processing*, vol. 23, no. 12, pp. 4968–4981, 2014.

[16] H. Malvar, L.-w. He, and R. Cutler, "High-quality linear interpolation for demosaicing of bayer-patterned color images," in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, 2004, pp. iii–485.

[17] J. Liang, J. Li, Z. Shen, and X. Zhang, "Wavelet frame based color image demosaicing," *Inverse Problems and Imaging*, vol. 3, Aug. 2013.

[18] L. Shao and A. U. Rehman, "Image demosaicing using content and colour-correlation analysis," *Signal Processing*, vol. 103, pp. 84–91, 2014, Image Restoration and Enhancement: Recent Advances and Applications, ISSN: 0165-1684.

[19] S. Arjomand Bigdeli, M. Zwicker, P. Favaro, and M. Jin, "Deep mean-shift priors for image restoration," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017.

[20] F. Wu, T. Huang, W. Dong, G. Shi, Z. Zheng, and X. Li, "Toward blind joint demosaicing and denoising of raw color filter array data," *Neurocomputing*, vol. 453, pp. 369–382, 2021, ISSN: 0925-2312.

[21] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017.