A SIMPLE IMAGE INPAINTING ALGORITHM WITH GENERATIVE ADVERSARIAL NETWORKS

Clayton F. S. Leite
Department of Electrical Engineering
Aalto University
clayton.souzaleite@aalto.fi

Aziza Zhanabatyrova
Department of Electrical Engineering
Aalto University
zhanabatyrova.aziza@aalto.fi

May 19, 2019

The code of this work is available at: https://github.com/claytonfk/simple_inpainting_GANs

1 Introduction

Image inpainting or completion consists in filling regions of missing pixels of an image with coherent synthetic information [1]. Some of its applications revolves around repairing damaged parts of aged photos, deleting distracting objects or people from the scenery or completing occluded regions [2].

Image inpainting is still a very open and challenging research topic. Current methods can be divided into two categories: traditional-based and learning-based approaches [1]. In the former, pixel information from neighboring spots are diffused to fill the missing region or patches of similar areas are copied to perform the image inpainting. In the latter category of approaches, deep neural networks are used to predict pixels for the missing regions.

Early studies attempted to solve the problem by copying the patches from the existing background into the missing holes [3]. These approaches are used in practical application and can deliver good results when the pictures present repetitive structures to copy from and are not highly structured.

Rapid development of deep learning methods inspired works that are categorized as learning-based methods [4]. These methods are able to generate new contents in highly structured pictures, such as pictures of faces, using convolutional neural networks (CNN) trained on large datasets. Yi et al. [5] proposed the use of Wasserstein generative adversarial networks (GANs) having a CNN with multiple columns, each of which with their own filter size, as the generator. Wasserstein GANs are also used in the work of Yu et al. [1] with a global discriminator - evaluating the consistency of the reconstruction with the entire image - and a local discriminator - focusing on the local consistency of the reconstruction.

The goal of this work is to create a simple algorithm based on GANs for the image inpainting problem and investigate the difference in performance between this method and the more complex state-of-the-art solutions. In Section 2, we define mathematically the method of this work. In Section 3, we detail the setup of the experiments. In Section 4, the evaluation of the method is performed. Finally, in Section 5, ending comments close this work.

2 Methods

The method presented in this work consists of two deep convolutional neural networks: a generator G and a discriminator D. These networks pit against each other in a minimax game [6]. Given a corrupted image I_c, the job of the generator is that of filling the missing pixels with visually realistic and consistent information, thus outputting a reconstruction $G(I_c) = I_r$. As for the discriminator, given an image, its purpose is to spot whether or not it is a reconstruction I_r or an original image I_o. Mathematically, the discriminator is trained to minimize the following loss function $\mathcal{L}_D$.

$$\mathcal{L}_D = \underset{I_o \sim p_{I_o}}{-\mathbb{E}} [\log(D(I_o))] - \underset{I_c \sim p_{I_c}}{\mathbb{E}} [\log(1 - D(G(I_c)))] \tag{1}$$

The generator not only aims at maximizing the second term of $\mathcal{L}_D$,but also to minimize the mean squared error between an image in its corrupted I_c and original I_o form. These two goals can be defined as a cost function $\mathcal{L}_G$ that the generator is trained to be minimize.

$$\mathcal{L}_G = \mathop{\mathbb{E}}_{I_c \sim p_{I_c}} \lambda_1 \text{MSE}(I_t, G(I_c)) + \mathop{\mathbb{E}}_{I_c \sim p_{I_c}} \lambda_2 [\log(1 - D(G(I_c)))] \tag{2}$$

where λ_1 and λ_2 are real-valued positive scaling factors between the two goals of minimization and MSE$(\cdot)$ is the mean square error function.

3 Experimental Setup

The methods were implemented on Python 3.6 using the Pytorch 1.0 framework and the code was run using 4 NVidia Tesla V100 GPU's for both training and testing.

Composed of 15000 street images of Paris extracted from Google's mapping service, the Paris StreetView [7] was used as dataset. The testing set contains 100 of these images, whereas the training set is formed by the remaining 14900 images. Before being input to the network, the images are normalized by dividing every individual pixel by 127.5 and then subtracting 1.

During the training, crops of size 256x256 pixels are randomly extracted from the training images - which have originally a resolution of 936x537 pixels. The images used for the testing phase have a resolution of 227x227 pixels and these are not cropped in smaller parts. Also, the corrupted images are obtained by drawing in the center of the images a white square of sides equal to 100 pixels. Following previous works in image inpainting [1, 7, 2, 5, 8, 9, 10], we utilize the peak signal to noise ratio (PSNR) and the structural similarity index (SSIM). The choices of resolution, hole size and metrics make it possible to easily compare our work with related ones.

The network architectures for the generator and discriminator are depicted in Figure 1 and Figure 2, respectively, as well as detailed in Table 1 and Table 2. The networks were trained for 5 epochs with a batch size equal to 50. The Adam algorithm was used to minimize the loss functions with learning rate of 0.00025 for the generator and 0.0001 for the discriminator. The parameters λ_1 and λ_2 were chosen to be 1 and 0.01, respectively. These choices of values were made by trial and error in several preliminary experiments.

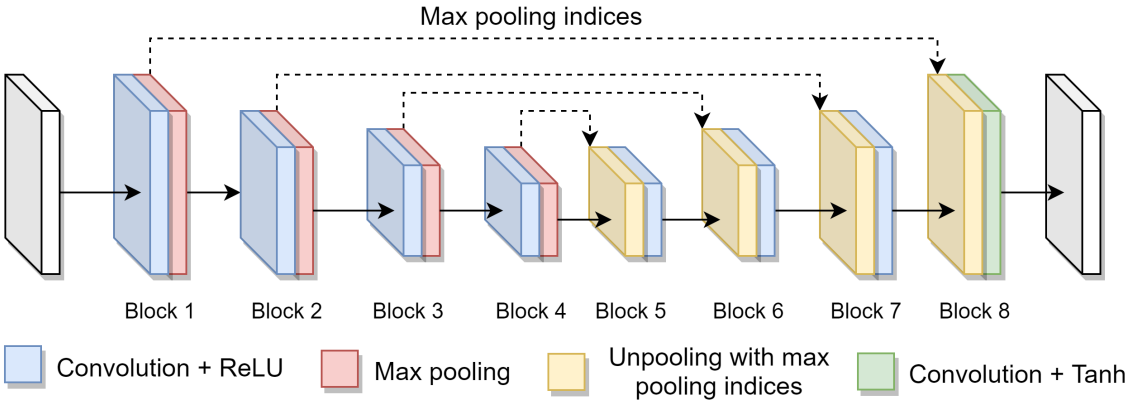

Figure 1: Network architecture of the generator inspired by the paper of Shelhamer et al. [11].

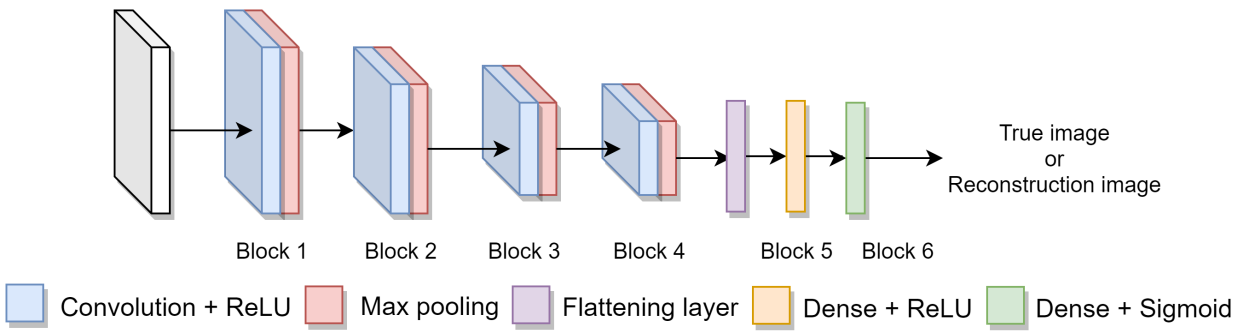

Figure 2: Network architecture of the discriminator.

Block	Layer	Input shape [N x C x H x W]	Output shape [N x C x H x W]	Details
#1	Convolution ReLU Max pooling	50 x 3 x 256 x 256	50 x 64 x 128 x 128	Conv. kernel: 4x4, padding 2, stride 1 Pooling kernel: 2x2, stride 2
#2		50 x 64 x 128 x 128	50 x 128 x 64 x 64	
#3		50 x 128 x 64 x 64	50 x 256 x 32 x 32	
#4		50 x 256 x 32 x 32	50 x 512 x 16 x 16	
#5	Unpooling Convolution ReLU	50 x 512 x 16 x 16	50 x 256 x 32 x 32	Conv. kernel: 4x4, padding 2, stride 1 Unpooling kernel: 2x2, stride 2
#6		50 x 256 x 32 x 32	50 x 128 x 64 x 64	
#7		50 x 128 x 64 x 64	50 x 64 x 128 x 128	
#8	Unpooling Convolution Tanh	50 x 64 x 128 x 128	50 x 3 x 256 x 256	

Table 1: Details of the architecture of the generator.

Block	Layer	Input shape [N x C x H x W]	Output shape [N x C x H x W]	Details
#1	Convolution ReLU Max pooling	50 x 3 x 256 x 256	50 x 8 x 128 x 128	Conv. kernel: 4x4, padding 2, stride 1 Pooling kernel: 2x2, stride 2
#2		50 x 8 x 128 x 128	50 x 16 x 64 x 64	
#3		50 x 16 x 64 x 64	50 x 16 x 32 x 32	
#4		50 x 16 x 32 x 32	50 x 16 x 16 x 16	
#5	Dense + ReLU	50 x 4096	50 x 256	-
#6	Dense + Sigmoid	50 x 256	50 x 1	-

Table 2: Details of the architecture of the discriminator.

4 Evaluation

The quantitative results of our work, compared with others, are listed in Table 3. Figure 3 and Figure 4 show the qualitative results of our method.

Method	SSIM	PSNR
CE [7]	**0.87**	23.49
MSNPS [8]	0.84	**24.44**
CA [1]	0.85	23.78
Multicolumn CNNs [5]	0.86	24.65
GLCIC [9]	0.84	24.07
EdgeConnect [10]	0.86	24.23
Ours	0.81	16.12

Table 3: Comparison of the results with related works.

Figure 3: Visual comparison of the results. From left to right: input, Context Encoder [7], pix2pix [12], MSNPS [8], ShiftNet [2], ours.

Figure 4: Other visual results of our model. From left to right: reconstruction, ground truth and input.

The quantitative results by themselves do not transmit much information on the limitations of our work, however, with the help of the qualitative observations, one can observe that the simple GANs model performs bad at reproducing the color information of the region surrounding the holes, as well as it outputs a low resolution and blurred reconstruction. Also, it shows a lack of good understanding of the continuity of the structures (e.g. door, people, poles, trees) in the reconstructed information.

5 Conclusions

In this work, we have approached the image inpainting problem with a simple algorithm based on GANs. The qualitative and quantitative results have shown a good deal of difference with respect to the state-of-the-art results. We could infer that the cost for the complexity of the state-of-the-art models is compensated with more satisfactory results. For future work, we would like to address solutions that would lead the model to better reproduce color and structural information in the holes of the images.

References

[1] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. *arXiv preprint arXiv:1801.07892*, 2018.

[2] Zhaoyi Yan, Xiaoming Li, Mu Li, Wangmeng Zuo, and Shiguang Shan. Shift-net: Image inpainting via deep feature rearrangement. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 3–19, Cham, 2018. Springer International Publishing.

[3] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. In *ACM Transactions on Graphics (ToG)*, volume 28, page 24. ACM, 2009.

[4] Yijun Li, Sifei Liu, Jimei Yang, and Ming-Hsuan Yang. Generative face completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3911–3919, 2017.

[5] Yi Wang, Xin Tao, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. Image inpainting via generative multi-column convolutional neural networks. *CoRR*, abs/1810.08771, 2018.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[7] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. *CoRR*, abs/1604.07379, 2016.

[8] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. High-resolution image inpainting using multi-scale neural patch synthesis. *CoRR*, abs/1611.09969, 2016.

[9] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Trans. Graph.*, 36(4):107:1–107:14, July 2017.

[10] Kamyar Nazeri, Eric Ng, Tony Joseph, Faisal Z. Qureshi, and Mehran Ebrahimi. Edgeconnect: Generative image inpainting with adversarial edge learning. *CoRR*, abs/1901.00212, 2019.

[11] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651, 2017.

[12] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. *CoRR*, abs/1611.07004, 2016.