Low-Light Image Quality Enhancement using Deep Convolutional Generative Adversarial Network

Leo Adlakha[1], Prateek Bhardwaj[1] and Abhijeet Singh Varun[1]
[1]Department Computer Engineering, Netaji Subhas University of Technology, Delhi, India

Abstract - We present a novel approach to adjust various image properties of low-light images to yield an enhanced image with better contrast, brightness etc. The problem is to enhance the quality of images taken from various mobile devices like iPhone, Samsung etc. We propose a deep learning-based approach involving Deep Convolutional Generative Adversarial Networks (GANs), that can be trained with a pair of images of low quality as a noise matrix and the output of the generator is used to compare the results when this output and high quality image (groundtruth) are fed into the Discriminator to obtain various losses involving color, texture etc. It works very well on real life test images taken from the publicly available datasets like Samsung-S7 and MIT-Adobe 5K. We have also hosted our model using Heroku here[https://iqe-os.herokuapp.com/]. You can also refer to the Github Repository [https://github.com/Leo-Adlakha/IQE] for more information.

Introduction

Image quality is of fundamental importance in any imaging system, especially DSLR and smartphone cameras. Modern smartphones for example, provide multiple camera sensors and sophisticated image signal processor (ISP) pipelines to produce images with good contrast, detail, colours, and dynamic range while at the same time mitigating against degradations. However, despite great advances in imaging hardware and ISP pipelines, there remains substantial room for improvement. Even professional photographers using DSLR will spend sig- nificant time using photo editing software to produce a quality digital photograph. Images in low-quality are affected by low contrast, bad visibility and high noise. These are the challenges of image quality enhancement problems we have as humans prefer high quality image which has good visibility. Many deep learning-based approaches are already introduced in this field including the benchmark model CURL.

We introduce a Deep Convolutional Generative Adversarial Network (GANs) based architecture to produce High Quality images from Low Quality images. DCGANs is a class of machine learning algorithms where two neural networks usually called Generator and Discriminator compete with each other to present better results. In DCGANs usually we have a convolutional network as the Generator and same for a discriminator, but we introduce residual blocks in our network of generators. The Residual Blocks usually have better performance as compared to the usual CNN because ResNets tend to learn the identity function quickly as compared to when required in the CNN layer. Our model also uses various loss functions involving various aspects of the images which help in improving that particular feature and also the overall quality of the image. Using the training data from the publicly available dataset and then after preprocessing it, it learns to generate new results with the same statistics as training. Here we have multiple pairs of low and high quality images which act as the training data. Low light images are fed into the generator which outputs an enhanced image and then the output of the generator and high-quality image goes to the discriminator to find the various losses and accordingly update weights of the various layers of generator and discriminator.. We trained our network on the (100, 100, 3) cropped images of the training images. Fig. 1 shows an example of the results of our model with the low quality image of the same scene. The technologies we used in this are open source and we have described it in the next section..

(a) (b)

Fig. 1 (a) It shows a Low-Quality Image captured by an iPhone, (b) It shows the result of our model when the image in (a) is fed into our generator.

Open Source Technologies Used

- **Tensorflow 2 -** We used tensorflow for constructing our DCGAN model. It contains various functions to construct layers for the model like conv2D, BatchNormalization, activation functions like relu, softmax, etc. Apart from this, it also has various classes to handle image matrices named tensors and various useful operations related to it.

- **Django Framework -** It is a high-level Python Web framework that encourages rapid development and clean, pragmatic design.. We used Django for the rapid development of our website, so that our work is available to others and help them to enhance their low-light images to enhanced ones without having the knowledge of various softwares like Adobe Lightroom etc.

- **Python -** We used python as our base programming language to develop the model and our website.

- **Numpy, Matplotlib, Pillow, Scipy -** We used the above mentioned libraries to handle various tensors using numpy arrays, matplotlib to check the performance of our model by plotting results of the generator, pillow for opening images of various formats into numpy arrays, Scipy for various mathematical operations.

- **Heroku, Gunicorn -** It was used to host our website on the Heroku Servers. Gunicorn was used to host WSGI based Django applications. Gunicorn is a WSGI HTTP server for UNIX, fast clients and sleepy applications.

- **psycopg2 -** We used it to migrate the default databases in Django Framework. It is an adapter to the PostgreSQL Database using Python Programming Language.

- **Whitenoise -** We used whitenoise as a middleware in our application to handle and serve all the static files related to the website like scripts, images, css files, etc.

- **Git -** One of the important open-source technologies used is git to manage the contributions by various members of the group. It is a version control system which helps many people to collaborate on a project, along with maintaining a history of various commits.

Related Work

1. Deep Bilateral Learning for Real-Time Image Enhancement

It presents a machine learning approach where the effect of a reference filter, pipeline, or even subjective manual photo adjustment is learned by a deep network (CNN) that can be evaluated quickly and with cost independent of the reference's complexity. It performs most predictions in a low-resolution bilateral grid. It introduces a new node for deep learning that performs a data-dependent operation. This allows the alleged slicing operation that reconstructs an output image at full image resolution from the 3D bilateral grid by considering every pixel's input color in addition to its x, y location. Architecture is constructed to learn a local affine color transformation that will be applied to the input through a new multiplicative node. The loss function used during training is evaluated at full resolution.

2. EnlightenGAN: Deep Light Enhancement without Paired Supervision

This paper explores the low light image enhancement problem. It proposes a generative adversarial network (GAN), EnlightenGAN, that can be trained without low/normal-light image pairs. It uses unsupervised image-to-image translation, adopts GANs to build an unpaired mapping between low and normal light image spaces without relying on exactly paired images. It uses dual-discriminator to balance the low-light enhancement and self-regularized perceptual loss to constrain the feature distance between low-light input image and its enhanced image. It also involves a self-regularized attention map at each level of deep features to exploit the illumination information of the low-light input image.

Dataset

1. Samsung – S7 Dataset

This is a dataset of pairs of RAW and JPEG images captured using the Samsung S7 rear camera. For each scene, both normally lit and low light are captured (low light is simulated by shorter exposure). A total of 110 scenes are captured in full resolution (12M pixel). The dataset is used and described in the paper "DeepISP: Learning End-to-End Image Processing Pipeline". The above dataset was used by dividing each frame into multiple frames of size (100, 100, 3), where first and second dimension tell about the height and width of the cropped frame and third refers to the channels RGB.

2. MIT-Adobe 5K Dataset

It contains 5000 photographs taken by a SLR camera with the help of some photographers. They are all in RAW format; that is, all the information recorded by the camera sensor is preserved. The images cover a broad range of scenes, subjects, and lighting conditions. This dataset was then made by hiring five photography students in an art school to adjust the tone of the photos. Each of them retouched all the 5,000 photos using a software dedicated to photo adjustment (Adobe Lightroom) on which they were extensively trained. The retouchers were asked to achieve visually pleasing renditions, akin to a postcard. The retouchers were compensated for their work. We took the work done by Expert C as it was the set of images favoured by most of the people. Then again they were cropped in frames of (100, 100, 3) to help train our model.

Method

It consists of two models –

1. Generator : It consists of 16 layers (excluding Activation Layers) of convolution and batch normalization. The Architecture has the same padding and stride of 1 across all the relevant dimensions. The Architecture starts with a Convolution Layer of 64 filters with filter size of 9 x 9. After which follows four sets of Residual Blocks. These Residual Blocks consist of Conv2D layers and batch normalized layers with filter size of 3 x 3 with 64 filters each. After the residual blocks, we have another set of Conv2D layers, 2 with Activation functions as ReLU and 64 3 x 3 filters, whereas 1 with tanh as its activation function and 64 9 x 9 filters.

2. Discriminator : It consists of 11 layers. The first being a Conv2D layer with 48 11 x 11 filters with the same padding and stride as four along the channel matrix of the image. Then, we have a set of Conv2D followed by Batch Normalization Layer

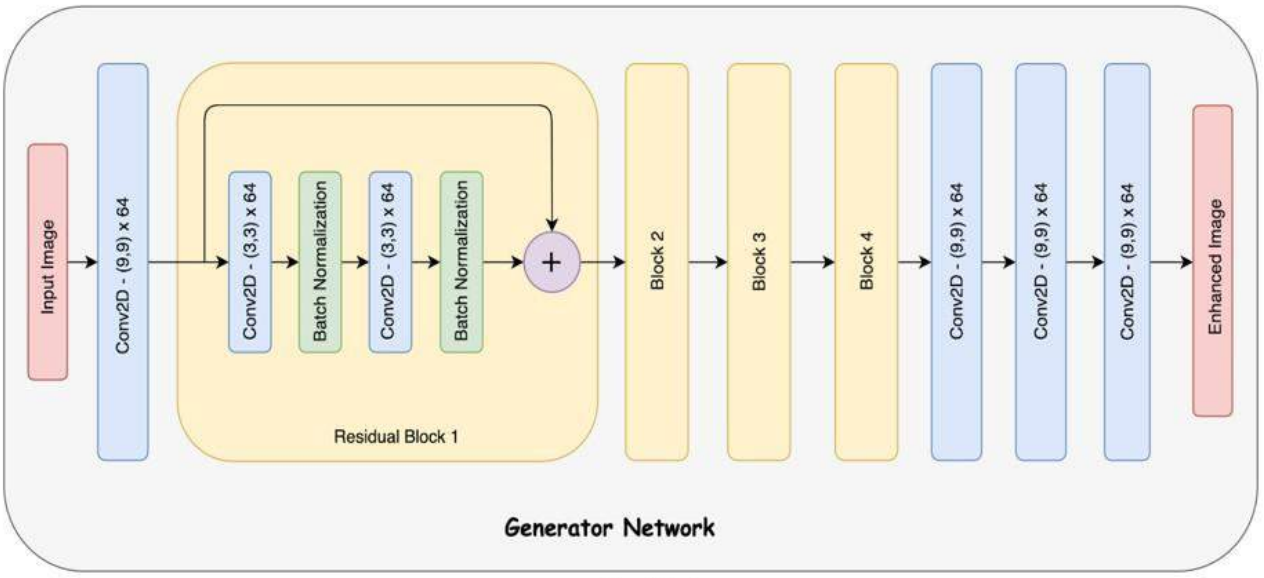


Fig. 2(a)

with Leaky ReLU with negative x slope / alpha as 0.2, as their activation function of the following configuration –

a. Conv2D – 128 filters of size 5 x 5 with the same padding and stride as 2.
b. Conv2D – 192 filters of size 3 x 3 with the same padding and stride as 1.
c. Conv2D – 192 filters of size 3 x 3 with the same padding and stride as 1.
d. Conv2D – 128 filters of size 3 x 3 with the same padding and stride as 2.

After this we flatten our image to process it through the Dense / Fully Connected Layer resulting in dimension [BATCH_SIZE, 128 * 7 * 7]. Now, we pass it through Dense Layers with 1024 neurons / units in the hidden layer. Then the output is processed with Leaky ReLU Activation function and then again the Dense Layer to get the result / measure of fakeness of an image with Softmax Activation Function.

The network was trained on a Nvidia Tesla GPU for 20K iterations using a batch size of 50. The images were processed by taking a crop of 100 x 100 across all the images in the dataset. The parameters of the network were optimized using Adam [11] modification of stochastic gradient descent with a learning rate of 5e-4.
Figure 2 shows the DCGAN network used by us.

Loss Function

The main issue of the image quality enhancement task is that input and target photos can't be matched densely i.e., pixel-to-pixel, completely different optics and sensors cause non-linear distortions and aberrations, resulting in a non-constant shift of pixels. Therefore, the standard per-pixel losses are not applicable in our case. We build our loss function under the assumption that the overall image quality can be decomposed into three independent qualities, color, texture and content.
We now define loss functions for each component, and ensure invariance to local shifts by design.

1.Color Loss

To measure the color difference between the enhanced and target images, we propose applying a Gaussian blur and computing Euclidean distance between the obtained representations. Color loss is defined as:

$$\mathcal{L}_{\text{color}}(X, Y) = \|X_b - Y_b\|_2^2,$$

where X_b and Y_b are the blurred images of X and Y,

$$X_b(i, j) = \sum_{k,l} X(i + k, j + l) \cdot G(k, l),$$

and the 2D Gaussian blur operator is given by

$$G(k, l) = A \exp\left(-\frac{(k - \mu_x)^2}{2\sigma_x} - \frac{(l - \mu_y)^2}{2\sigma_y}\right)$$

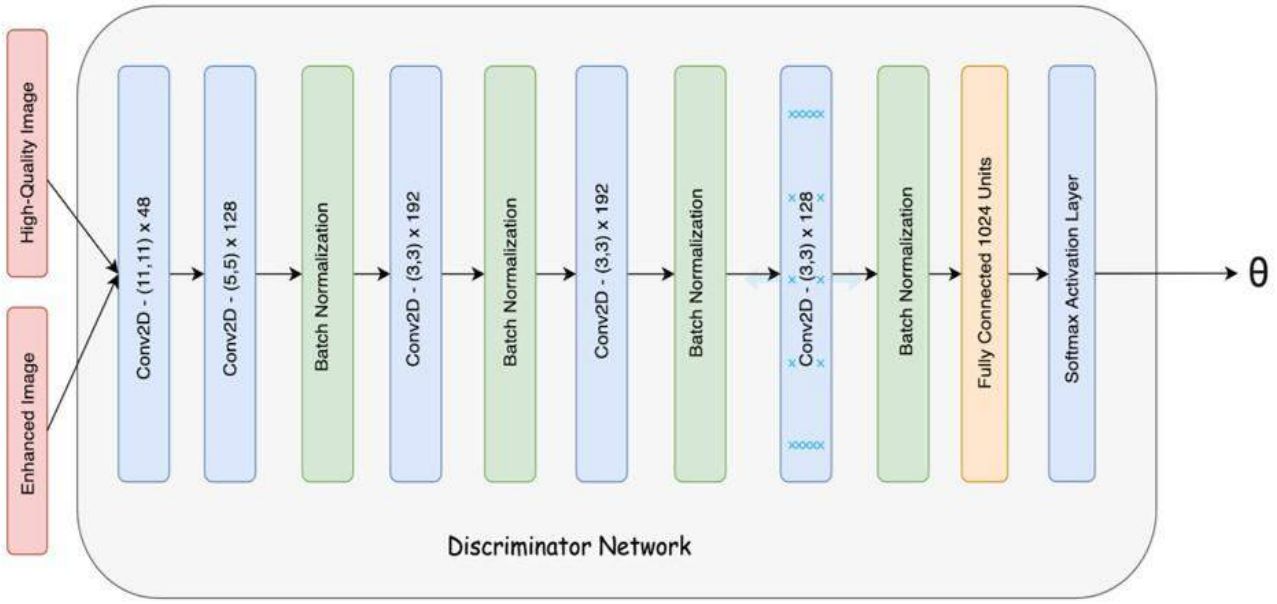


Fig. 2 (b)

where we defined A = 0.053, μx,y = 0 and σx,y = 3. The idea is to calculate the difference in brightness, contrast and major colors between the images while not considering texture and content comparison. For this, we define a constant σ by visual inspection, the smallest value that ensures that texture and content are not considered. The important property of this loss is its unchanginess to very small distortions. As a result, color loss manages the improved images to have the identical color distribution as the target one, while being invariant to small mismatches.

2.Texture loss

It observes both improved and target images, and its goal is to predict whether or not the input image is real or not. It is trained to minimise the cross-entropy loss function possible. It is written as:

$$\mathcal{L}_{\text{texture}} = -\sum \log D\left(F_{\mathbf{W}}(I_s), I_t\right)$$

where Fw and D denote the generator and discriminator networks, respectively.

The discriminator is pre-trained on the {phone, DSLR} image pairs, and then trained jointly with the proposed network as is conventional for GANs. This loss is shift-invariant thus no alignment is needed.

3.Content Loss

We outlined it based on the activation maps produced by the ReLU layers of the pretrained VGG-19 network. Rather than measuring per-pixel difference between the images, this loss encourages them to own similar feature illustrations that contain numerous aspects of their content and perceptual quality. In our case, it is used to preserve image semantics since other losses don't consider it. It is defined as Euclidean distance between features of the enhanced and target images. It is written as:

$$\mathcal{L}_{\text{content}} = \frac{1}{C_j H_j W_j} \left\| \psi_j\left(F_{\mathbf{W}}(I_s)\right) - \psi_j\left(I_t\right) \right\|$$

where Cj, Hj and Wj denote the number, height and width of the feature maps, Fw(Is) the enhanced image and ψj() be the feature map obtained after the jth convolutional layer of the VGG-19 CNN.

4.Total Variational Loss

It enforce spatial smoothness of the improved images:

$$\mathcal{L}_{\text{tv}} = \frac{1}{CHW} \left\| \nabla_x F_{\mathbf{W}}(I_s) + \nabla_y F_{\mathbf{W}}(I_s) \right\|$$

where C, H and W are the dimensions of the generated image Fw(Is).

5.Total Loss

Final loss is defined as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{content}} + 0.4 \cdot \mathcal{L}_{\text{texture}} + 0.1 \cdot \mathcal{L}_{\text{color}} + 400 \cdot \mathcal{L}_{\text{tv}}$$

The coefficients were chosen based on preliminary experiments on the DPED training data.

Result and Discussion

In our work we mainly focus on the light enhancement problem of images and cover it well. Our output is generally accurate and, even when it differs from the ground-truth, it remains plausible. The quality of image is a subjective concept but the resulting images of our work looks pretty good. Image quality can be assessed using two methods: subjective and objective. Subjective methods are based on the individual assessment while objective methods are based on computational models that can predict image quality. Here we consider Peak signal to noise ratio (PSNR) and structural index similarity (SSIM). They are two measuring tools that are widely used in image quality assessment.

We compare our work with the previous methods as mentioned in table 1.

Table 1

Method	**PSNR**	**SSIM**
CURL	24.2	0.88
Our Work	**23.04**	**0.9437**
DPE	22.15	0.850
HDRNet	21.96	0.866
Distort-and-recover	20.97	0.841
White - Box	18.57	0.701

1.PSNR

It is used as a quality measurement between the original and a reconstructed image. The mean-squared error (MSE) and the PSNR are used to compare image compression quality. The MSE represents the cumulative squared error between the compressed and the original image, whereas PSNR represents a measure of the peak error. The lower the value of MSE, the lower the error. MSE is calculated as:

$$MSE = \frac{\sum_{M,N} [I_1(m,n) - I_2(m,n)]^2}{M * N}$$

where m,n are numbers of rows and columns in images. PSNR is calculated as:

$$PSNR = 10 \log_{10} \left(\frac{R^2}{MSE} \right)$$

where R is the maximum fluctuation in input image data type (if the input image has a double-precision floating-point data type, then R is 1. If it has an 8-bit unsigned integer data type, R is 255). For color images, the definition of PSNR is the same except the MSE is the sum over all squared value differences divided by image size and by three.

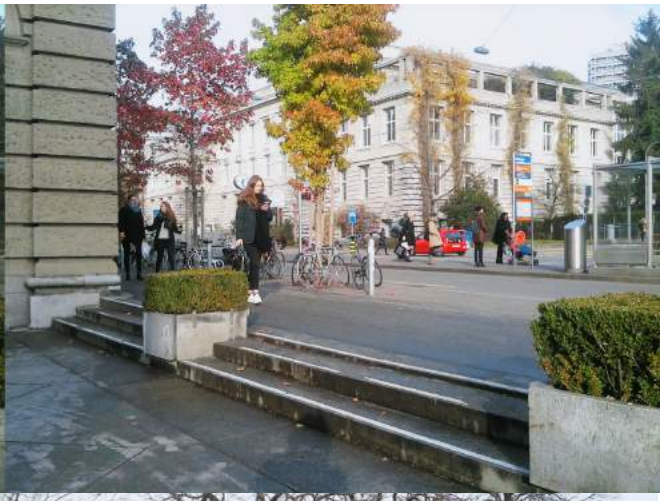
The higher the PSNR, the better the quality of the reconstructed image. We can see in the table our model has PSNR value 23.04 which is pretty well as it beat many previous methods but less than CURL (Best method till now).

2. SSIM

It is a metric that quantifies image quality degradation caused by processing like data compression or by losses in data transmission.It is a method for measuring the similarity between two images. The SSIM values range between 0 to 1, 1 means it perfectly matches the reconstructed image with the original one. Generally SSIM values 0.97, 0.98, 0.99 for good quality reconstruction techniques.

Our work has been quite satisfactory in this measure also.

Looking overall we have got very good results in both parameters SSIM and PSNR.





Conclusion

In this paper, we introduced a Deep Convolutional Generative Adversarial Network architecture (DCGANs) with multiple layers of convolution, residual blocks, and fully connected layers, that can perform low-light image enhancement on full-resolution images. Our model has been trained using pairs of low and high quality images which help the network to learn the enhancement property for the images like contrast, brightness etc. in a much easier fashion. The model has been tested on various real life test images and it performs well in comparison to the benchmark CURL model. The loss function we consider is dependent on the image qualities in terms of color, texture and content and are independent (Assumption). Other networks were also used (i.e. VGG-19, which classifies among 1000 classes) in order to account for the content loss. In future work, we try to build the model if the image qualities (color, texture and content) are dependent on each other and consider underwater images to enhance the area for this model to work.

References

1. Eli Schwartz, Raja Giryes, and Alex M. Bronstein. "DeepISP: Learning End-to-End Image Processing Pipeline." arXiv preprint arXiv:1801.06724 (2018).

2. Vladimir Bychkovsky, Sylvain Paris, Eric Chan, & Frédo Durand (2011). Learning Photographic Global Tonal Adjustment with a Database of Input / Output Image Pairs. In The Twenty-Fourth IEEE Conference on Computer Vision and Pattern Recognition.

3. Loh, Y., & Chan, C. (2019). Getting to Know Low-light Images with The Exclusively Dark DatasetComputer Vision and Image Understanding, 178, 30-42.

4. C. Li, C. Guo, W. Ren, R. Cong, J. Hou, S. Kwong, D. Tao, “An Underwater Image Enhancement

Benchmark Dataset and Beyond," IEEE Trans. Image Process., vol. 29, pp.4376-4389, 2019.

5. Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, & Yoshua Bengio. (2014). Generative Adversarial Networks.

6. Alec Radford, Luke Metz, & Soumith Chintala. (2016). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks.

7. Karen Simonyan, & Andrew Zisserman. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition.

8. Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. (2015). Deep Residual Learning for Image Recognition.

9. Fernando A. Fardo, Victor H. Conforto, Francisco C. de Oliveira, & Paulo S. Rodrigues. (2016). A Formal Evaluation of PSNR as Quality Measurement Parameter for Image Segmentation Algorithms.

10. 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, Apr. 2004.

11. Mart´ın Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghe- mawat, Geoffrey Irving, Michael Isard, et al. TensorFlow: a system for large-scale machine learning. In OSDI, vol- ume 16, pages 265–283, 2016.

12. Yu-ShengChen, Yu-ChingWang, Man-HsinKao, andYung- Yu Chuang. Deep photo enhancer: Unpaired learning for im- age enhancement from photographs with GANs. In CVPR, 2018.

13. M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. In arXiv preprint arXiv:1701.07875, 2017.

14. M. Gharbi, J. Chen, J. T. Barron, S. W. Hasinoff, and F. Du- rand. Deep bilateral learning for real-time image enhance- ment. ACM Transactions on Graphics, 36(4):118, 2017.

15.P.Isola,J.Y.Zhu,T.Zhou,andA.A.Efros.Image-to-im age translation with conditional adversarial networks. In Pro- ceedings of IEEE Conference on Computer Vision and Pat- tern Recognition (CVPR), July 2017.

16.N. Kodali, J. Abernethy, J. Hays, and Z. Kira. On convergence and stability of GANs. In arXiv preprint arXiv:1705.07215, 2017.

17. Raman Maini, & Himanshu Aggarwal. (2010). A Comprehensive Review of Image Enhancement Techniques.

18. Yifan Jiang, Xinyu Gong, Ding Liu, Yu Cheng, Chen Fang, Xiaohui Shen, Jianchao Yang, Pan Zhou, & Zhangyang Wang. (2021). EnlightenGAN: Deep Light Enhancement without Paired Supervision.

19. O'Shea, K., & Nash, R. (2015). An Introduction to Convolutional Neural Networks. ArXiv e-prints.

20. Diederik P. Kingma, & Jimmy Ba. (2017). Adam: A Method for Stochastic Optimization.