# DAY-2-NIGHT IMAGE-TO-IMAGE TRANSLATION

*Rushabh Dharia, Nahush Raichura, Animesh Sagar, Anthony Tai*


## ABSTRACT

We investigate three computer vision techniques for converting scenes from day-time to night-time. The first approach automates a Photoshop method to automatically adjust the brightness, contrast, and the image curve layer of the daytime image, followed by noise addition and sky scene modification to produce the final nighttime image. The other two approaches utilize existing Deep Learning models (Cycle-GAN and Conditional Adversarial Networks) to map input images to output images. Both models follow the principles of the Generative Adversarial Network architecture with augmented loss functions to solve daytime-image-to-nighttime-image problems. We evaluate and compare the performance of the three techniques in a perceptive manner. Our experiments show that although the Photoshop technique generates good customized images, the Deep Learning approaches produce better results on generalized images that span similar context to that of the training dataset. Furthermore, we observe that Cycle-GAN is slower in terms of training process but generalizes better to unpaired data, while Conditional Adversarial Network performs better with paired image instances.


## 1. INTRODUCTION

The image-to-image translation is a computer vision technique that can be used to produce desired output images given a set of input images and specific conditions. As defined by Isola et al. [7], it is a "task of translating one possible representation of a scene into another". The "scene" can be a segmented street photo, a Black and White picture of flowers, or the outline of a handbag. The corresponding desired output images can be a recovered street scene, a colorized version of the flower image, or a handbag with specific pattern and color, as shown in Figure 1. Adobe Photoshop is a graphics editor capable of editing and composing digital images, including graphics in 3-dimensions and video. On the other hand, many Deep Learning methods, including Cycle-GANs [3] and Conditional Adversarial Networks [7], adapt Generative Adversarial Network [4] principles to capture the unique features of the source domain containing the input images and that of the target domain. This allows the prediction models to learn the association between the source and target domains during the training stage.

In this work, we examine these three different approaches for creating nighttime scenes corresponding to images captured at daytime. With the Adobe dataset for Nightfromday 3d, we show that the Photoshop method can be automated to generate nighttime scenes with convincing customized features. Similarly, we demonstrate that given enough training data, both CycleGAN and Conditional Adversarial Network models are capable of producing reasonable image-to-image translation output.

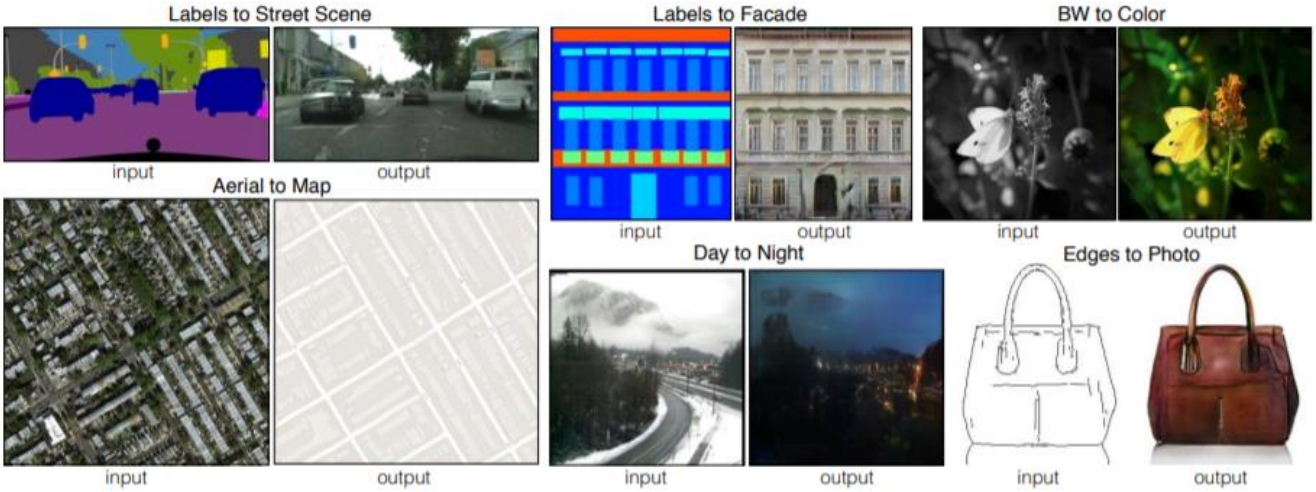


Figure 1. Examples of image-to-image translation [7]

## 2. BACKGROUND AND RELATED WORK

**Generative Adversarial Networks (GANs)** [4] - have achieved impressive results in image generation, image editing, and representation learning. Recent methods adopt the same idea for conditional image generation applications, such as text2image, future prediction etc. GANs have been successful because of adversarial loss that helps the generated images to be indistinguishable from real photos.

**Image-to-Image Translation** - Our approach builds on the "pix2pix" framework of Isola et al. [7], which uses a conditional generative adversarial network to learn a mapping from input to output images. Similar ideas have been applied to various tasks such as generating photographs from sketches or from the attribute and semantic layouts.

**Cycle-GAN** [3] is a novel approach for learning to translate an image from a source domain X to a target domain Y in the absence of paired examples. The goal is to learn a mapping G : $X \rightarrow Y$ such that the distribution of images from G(X) is indistinguishable from the distribution Y using an adversarial loss. Because this mapping is highly under-constrained, it is coupled with an inverse mapping F: $Y \rightarrow X$ and a cycle consistency loss is introduced to enforce $F(G(X)) \approx X$ (and vice versa).

## 3. METHODS

Since there is no definitive or a structured way to approach this problem, we tried to follow different approaches to solve the problem.

### 3.1. Naive Approach

As the name suggests it is the naive approach, this method aims at solving the problem with the rudimentary image processing approach. The approach is further divided into 5 steps:

1) First, we reduce the overall brightness and contrast of the image, simply by changing the alpha and the beta value of the brightness. By experimenting with different values we came to the conclusion that the alpha=0.9 and beta=-130 worked the best for most of the images.
2) The next step in this process is to apply a 3-dimensional lookup table to the image. This lookup table has been taken from Adobe [5] and has the name NightFromDay.cube. A 3d LUT basically maps each value to another RGB value. The data in this lookup table is designed in such a way that it enhances the blue channel in the image, giving it sort of a night effect. The lookup table is of the size 17x17x17 which means it has 4913 rows of data mapping each RGB value to a different RGB value.
3) The next thing we do is to detect the sky in the image. Sky detection in the image can be done using the following steps:( Referenced from [1])
    a. Convert the image to grayscale and using the Sobel operator calculate its the gradient of the image.
    b. Define an optimized Energy function J which will detect the presence of any sky region in the image.
    c. Segment the sky from the rest of the image.

We used the code given in [8] which implements the same approach and colored the rest of the landscape white and not the sky. We then stored all the sky pixels in a dictionary so that further modifications can be done in the sky region.

4) The next step was to generate a monochromatic Gaussian noise which is randomly distributed across the sky region. What this does is we randomly select pixels from the sky region and add black or white pixels to the sky region. Then we apply a blur filter of radius .25 pixels and a neon filter, to create some stars in the sky. In order to create stars of different intensity, we used a black stop and a white stop condition which clips all pixels below 175 in the pixel distribution histogram. Thus, giving a starry effect to the sky.

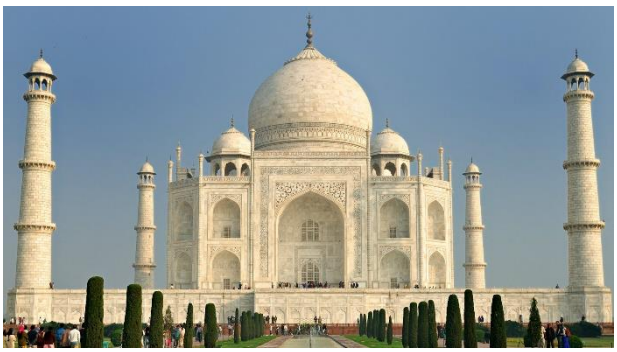

Figure 2. Input Image

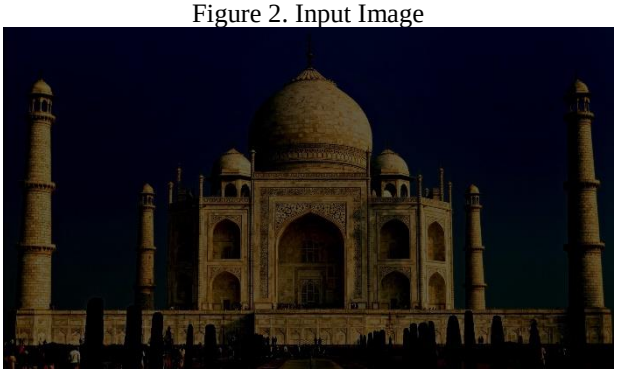

Figure 3. Image after reducing the brightness and the contrast and apply the 3dLUT

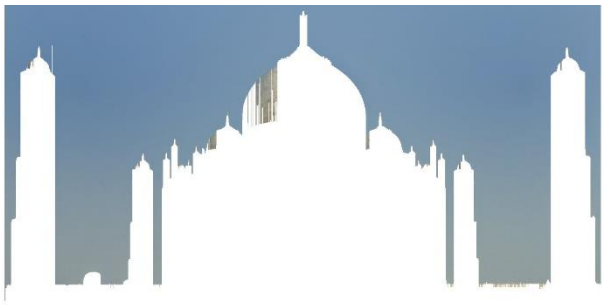

Figure 4. Sky Region detected

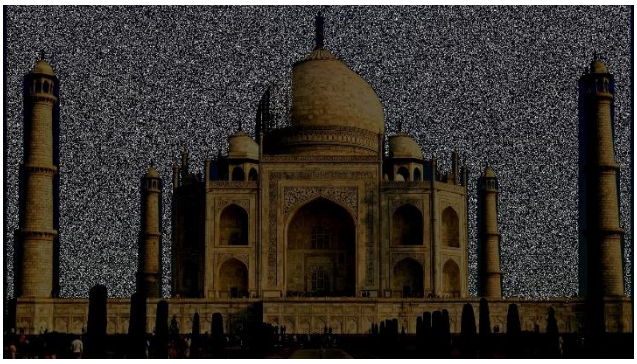

Figure 5. Monochromatic Gaussian Noise distributed equally over the sky region

Figure 6. Final output Image after applying gaussian blur and neon filter and histogram clipping

This approach is not intelligent enough to illuminate artificial light sources in the image. However, this problem can be solved using the next two deep learning approaches.

### 3.2 CycleGAN

One of the two Deep Learning approaches we propose to implement "Day to Night Image Translation" is through applying the CycleGAN technique using PyTorch, an open-source machine learning library. The procedure of CycleGAN consists of two main stages: (i) learning to map from a source domain to a target domain with minimum adversarial loss, and (ii) enforcing the inverse mapping from the target domain back to the source domain by minimizing a cycle consistency loss. With both adversarial and cycle consistency losses incorporated in its full objective function, CycelGAN trains both forward and inverse mappings at the same time to achieve unpaired image-to-image translation, resulting in and As CycleGAN learns to translate an image from a source domain to a target domain, the goal of our proposed approach is to employ the same method to map an image taken at daytime (the source domain) to the corresponding nighttime image (the target domain), thus completing the Day to Night image translation. The flow of the training data in forward and backward mapping procedures is depicted in Figure 2.
For this work, we used the PyTorch-CycleGAN code from https://github.com/aitorzip/PyTorch-CycleGAN.

To verify that the code performs without the need for further adjustments, we tested the code on a horse2zebra dataset. The training dataset contained 1,067 images of horses and 1,334 images of zebras. On the other hand, the test set was composed of 120 horse images and 140 zebra images. For a quick test, 3 epochs were run in order to capture errors in the procedure. After all modifications were put in place, we obtained a preliminary result with 260 translated images (120 horse-to-zebra and 140 zebra-to-horse). As expected, only a few resulting images were successfully translated. Some examples are given below:

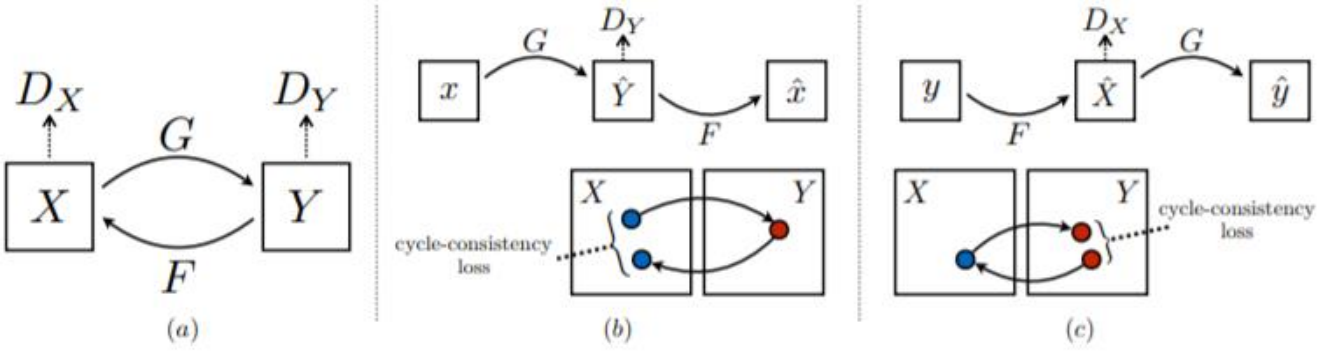


Figure 2. (a) The Cycle- GAN model consists of Forward and Backward Mapping Functions [3] G : X → Y and F : Y → X, and associated adversarial discriminators $D_Y$ and $D_X$. (b) forward cycle-consistency loss: x → G(x) → F(G(x)) ≈ x, and (c) backward cycle-consistency loss: y → F(y) → G(F(y)) ≈ $\hat{y}$

**Success:**


**Horse to Zebra**

**Failure:**


**Zebra to Incomplete Horse**

The code is then utilized to conduct Day to Night image translation on the daytime images we prepare for the proposed project shown in the results section. Encountering issues in training (mentioned in the Discussion Section) we decided to move forward with Conditional Adversarial nets

### 3.3 Conditional GANs

Conditional GANs learn a mapping from observed image x and random noise vector z, to y, G:{x,z} →y. The generator G is trained to produce outputs that cannot be distinguished from "real" images by an adversarially trained discriminator, D, which is trained to do as well as possible at detecting the generator's "fakes". The basic structure of the network is an encoder-decoder where the input passes through a series of layers which progressively downsample, until a bottleneck layer, after which the process is reversed.

Skip connections are used to circumvent bottleneck conditions and keep the desirable low-level information such as edges and other features based on the task at hand. These skip connections follow the general shape of a "U-Net".

The architecture can model high-frequencies by restricting attention to structure in the local image patches. Hence, the discriminator architecture PatchGAN only penalizes structure at the scale of patches where it tries to classify if each N x N patch in an image is real or fake. This discriminator runs convolutionally across the image averaging all responses to provide the output of D.

To optimize we use AdaGrad optimizer and found perceptively that the output was better than the Adam optimizer used in the original paper implementation. At inference time, generator net is run in exactly the same manner as during the training phase. This differs from the usual protocol in that the dropout is applied at test time and apply batch normalization using the statistics of the test batch, rather than aggregated statistics of the training batch. This approach to batch normalization, when the batch size is set to 1, has been termed "instance normalization" and has been demonstrated to be effective at image generation tasks

## 4. RESULTS

### 4.1 Naive approach results:

**Success:**

Figure 5. Left Image is the input and Right Image is the output. Successfully converted the day-time image of Eiffel Tower to night-time. Image taken from Google.

**Failure:**

Figure 6. Left Image is the input and Right Image is the output. Images which contain a lot of shadows did not perform too well in this approach as it resulted that the shadowed part turned out to be completely black. This figure is a failed test case for this approach. As their result is not very good as the parts of the images where there is no direct sunlight it gets completely darkened out.

**4.2 Cycle-GAN**
**Success:**

Figure 7. Left Image is the input and Right Image is the output.

**Failure:**

Figure 8. Left Image is the input and Right Image is the output.

**4.3 Conditional-GAN**

**Success:**

Figure 9. Left Image is the input and Right Image is the output.

**Failure:**

Figure 10. Left Image is the input and Right Image is the output.

## 5. DISCUSSION

The motivation of the Naïve approach is to automate a 15-20 step long process in Adobe Photoshop using Computer Vision techniques. The idea behind this entire process was taken from [9], and we just tried to replicate the steps. The main advantages of using this method are that the final output image produced is of the same resolution as that of the input image so there is 0-pixel distortion in the final output image. The next advantage is that this is a very fast process and hardly takes 2-3 mins to run on a Full-HD image as compared to the Cycle Gan which requires a huge amount of time required for training. This type of images are used in generating magazine images, which until now editors created using different photoshop Lookup tables and produced different artificial images.

The naive approach is not intelligent enough to illuminate artificial light sources in the image. We tried to detect light poles, windows, headlights in cars and all other sorts of light sources in the picture but it turned out that there are way too many different shapes and sizes and colors of light sources and it would be next to impossible to detect all of those, so this approach is not a very good approach for a generalized method to solve all sorts of problem but can be used effectively for images where we do not need to illuminate light sources. The deep learning methods would be better for this reason

Cycle-GANs give impressive results for unpaired dataset but need a lot of computational resources as there are two generators in it. Also learning the mapping between unpaired scenes requires much more data for training.

Conditional Adversarial Network works well for paired image dataset with a ground truth present for each input in order, to minimize the loss and disparity between these two. The training time is also comparatively lower than that of

Cycle-GAN. The architecture quickly learns the mapping from domain of day to that of night and you get results within 200 epochs. Failure cases are observed when the scene to be translated has poor visibility or any noise caused by transmission and compression artifacts.

## 6. CONCLUSION AND FUTURE WORK

In the Naive approach, there are instances where our sky detection algorithm did not work well for example in case of a horizon which separates the sea and the sky, the sky detection algorithm mistakenly marked even the sea as the sky because of its blue tint and constant gradient along with the sky. So, In future work better techniques needs to be developed to detect the sky in the image as it could be a very important aspect to a lot of concepts in computer vision.

We argue that Cycle-GANs training time can be reduced if we do not run the backwards mapping function for the first few epochs as the Generator (in Forward Mapping) may not have learned a proper mapping from x to $\hat{Y}$. So, learning a mapping from $\hat{Y}$ to $\hat{x}$ may not be beneficial.

## 7. ACKNOWLEDGEMENT

We would like to thank Professor David Crandall and Eriya Terada at Indiana University for their valuable guidance. We would also like to acknowledge the sky detection algorithm for the naive approach, which was developed by Chris Nelson and is available at https://github.com/cnelson/skydetector/blob/master/sky_detection.ipynb. In addition, we acknowledge that the CycleGAN code used in this project was developed by the authors of CycleGAN: Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Their code is located at https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix.

## 8. REFERENCES

[1] Shen, Yehu, and Qicong Wang. "Sky region detection in a single image for autonomous ground robot navigation. "International Journal of Advanced Robotic Systems 10.10 (2013): 362.

[2] Neuhausen, Marcel, et al. "Window detection in facade images for risk assessment in tunneling. "Visualization in Engineering 6.1 (2018): 1.

[3] Zhu, Jun-Yan, et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks. "Proceedings of the IEEE international conference on computer vision. 2017.

[4] Goodfellow, Ian, et al. "Generative adversarial nets. "Advances in neural information processing systems. 2014.

[5] Adobe dataset for Nightfromday 3d LUT:<https://github.com/picwellwisher12pk/Presets/blob/master/3DLUTs/NightFromDay.CUBE>

[6] Laffont, Pierre-Yves, et al. "Transient attributes for high-level understanding and editing of outdoor scenes. "ACM Transactions on Graphics (TOG)33.4 (2014): 149.

[7] Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[8] Sky Detection Code https://github.com/cnelson/skydetector/blob/master/sky_detection.ipynb

[9] Naive approach photoshop steps replicated from: http://www.designpanoply.com/blog/turn-day-to-night-using-photoshop