Deep Learning Project
Unpaired Day-to-Night translation using Cyclic GAN for Autonomous Driving

ID. Bhaswara - s2181738
S. Natarajan - s2122154
DH. Apriyanti - m7663513

February 13, 2019

Contents

1 Introduction

Currently, image-to-image translation using Generative Adversarial Network (GAN) has been a state-of-the-art in Deep Learning. This class of networks aims to learn a mapping between input and output images from a training set. GANs have achieved excelling results in various problems related to inpainting, image generation, image editing etc. The increasing research in GAN has also led to its applications in generating test data for evaluation of autonomous driving algorithms. The success of an autonomous driving system depends on training the neural network with varying scenes across different locations. The data generated by image transformations do not completely represent a realistic scenario. Further to this it is also difficult to generate certain scenes with snowy road conditions and trees using image transformations. Hence in order to automatically synthesize large amounts of realistic scenarios with varying weather conditions and locations unpaired translation using GAN has been explored.
The work aims to learn a translation between day-to-night scenarios for autonomous driving in the absence of paired training data.

2 Related work

Long et al. [1] proposes the use of CNN based approach to learn a parametric translation function from a dataset of input-output examples. Pix2pix method proposed by Isola et al. [2] uses Conditional GAN to learn mapping from input to output image and has achieved good performance. The above experiments are related to use cases of paired training dataset. However, in some cases, getting paired training images are difficult. This is typically a challenge that is relevant to autonomous driving. As it is difficult to collect huge amounts of data representing a wide range of test conditions. Hence, extensive research has been conducted for using GAN in unpaired image-to-image translation. Rosales et al.[3] proposed Bayesian framework that is based on Markov Random fields. Other researchers, Liu et al. [4] and Aytar et al. [5] used CoGAN and cross modal scene network respectively to learn the data across domains. In another paper, Liu et al. [6] also developed a method that is based on a combination of variational autoencoders and GAN. [4].

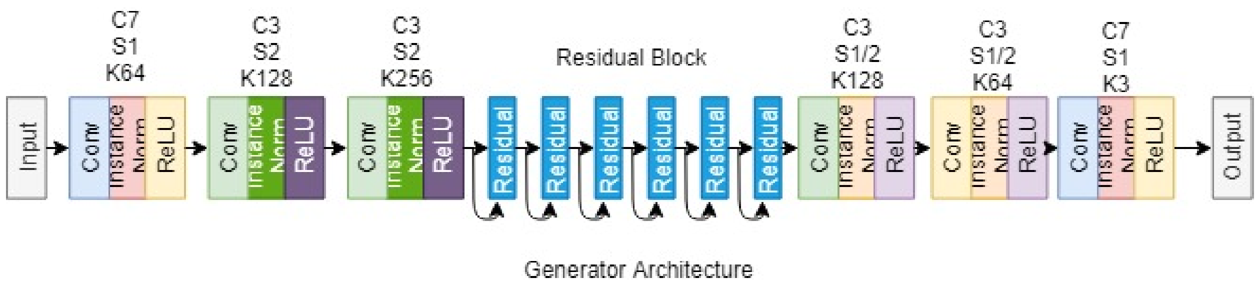
Yan-Zhu et al. [7] proposed a novel method called Cycle-GAN which is the state of art in the field of unpaired image-to-image translation. The Cycle GAN paper is a very notable contribution to this field mainly because the proposed method is not confined to specific application instead could be tailored for several other tasks. The paper presents a comparative analysis between several other architectures such as BiGAN [8, 9], CoGAN [4], SimGAN [10] and proposed CycleGAN. The CycleGAN architecture outperforms the baseline for translation in cityscapes data. Hence this has been taken as the reference architecture for analysis in the current work. From many models and variations proposed by the authors i.e. collection style transfer, object transfiguration, season transfer, photo generation from painting, photo enhancement, etc., day-to-night translation for autonomous driving has been chosen as the area of analysis.

3 Methodology

This section describes the details of the dataset chosen for analysis and the experimentation with Cycle GAN architecture adopted from [7] as shown in Figure 1.

3.1 Dataset

NEXET is the largest and most diverse dataset for road understanding and research, released by Nexar as a part of the Nexar Challenge II (`https://www.getnexar.com/challenge-2/`). The complete version of the dataset consists of 50,000 images from all over the world, recorded under varying lighting and weather conditions. The Part I dataset has been considered for analysis, it contains a total of 17000 images including day and night scenarios. The day and night images were separated based on a simple mean thresholding technique and a total of 5000 images of each day and night class were chosen for experimentation. The test data contains 150 images for day and night each chosen from the dataset other than the training data.



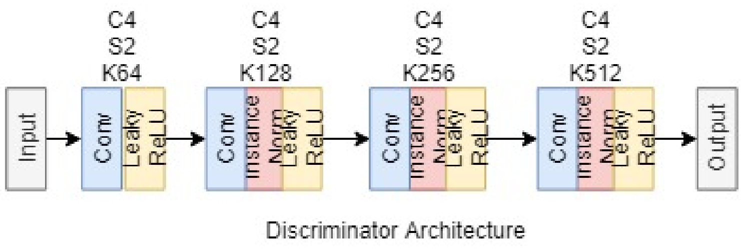


Figure 1: Cycle GAN Architecture

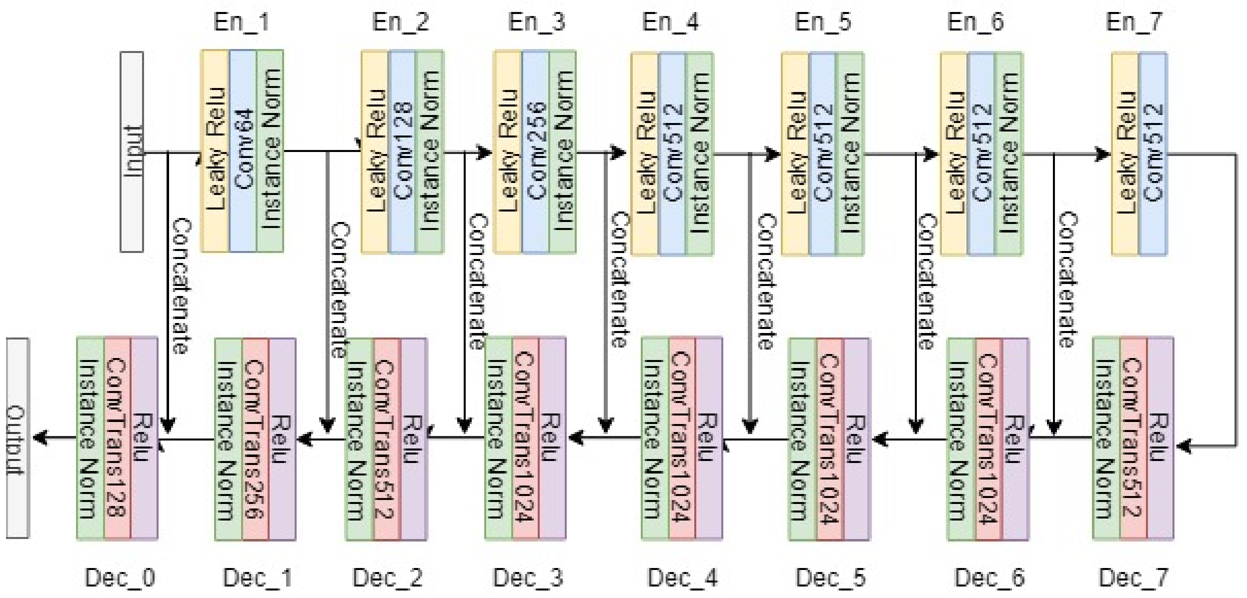


Figure 2: UNet Architecture

3.2 Implementation

The detailed architecture of the CycleGAN decribes a Generator and a Discriminator as seen in Figure 1.

Generator architecture
The generator module contains two stride-two convolutions, several residual blocks, and two fractionally-strided convolutions with stride 1/2. We use six blocks for 256x256 images and instance normalization. Let c7s1-k denote a 7x7 Convolution-InstanceNorm- ReLU layer with k filters and stride 1. dk denotes a 3 x 3 Convolution-InstanceNorm-ReLU layer with k filters and stride 2. Reflection padding was used to reduce artifacts. Rk denotes a residual block that contains two 3 x 3 convolutional layers with the same number of filters on both layer. uk denotes a 3 x 3 fractional-strided-Convolution- InstanceNorm-ReLU layer with k filters and stride 1/2 . The network with 6 residual blocks consists of: `c7s1-64,d128,d256,R256,R256,R256, R256,R256,R256,u128,u64,c7s1-3`

Discriminator architecture
For the discriminator, we use 70x70 PatchGAN. Let Ck denote a 4 x 4 Convolution-InstanceNorm-LeakyReLU layer with k filters and stride 2. After the last layer, we apply a convolution to produce a 1-dimensional output. The discriminator architectures that we used in this paper is same with the original paper [7]. We use InstanceNorm for all of the layer except the first convolutional layer and leaky ReLUs with a slope of 0.2. The discriminator architecture is: `C64-C128-C256-C512`

Training details
We used least-squares adversarial loss and λ=10. We also use the Adam solver [11] with a batch size of 1. All networks were trained from scratch with a learning rate of 0.0002. We keep the same learning rate for 100 epochs. Weights are initialized from a Gaussian distribution N (0, 0.02).

Github Link of the code

`https://github.com/bhaswara/DL_Project`

3.3 Analysis

The initial experiment with the standard CycleGAN architecture with 1000 images for training revealed certain flickering effects in test data. The section below describes the alternative experiments carried out to solve the problem of flickering in the test data.

3.3.1 Experimentation with Size of Dataset used for Training

The architecture proposed was evaluated with a varying sizes of training data to evaluate the effect of size of training data with respect to reconstruction. The training dataset was divided into sub sections of 1000, 2000, 3000, 4000, and 5000 images each for day and night scenarios. Then each of the subsets of train data were trained on the architecture described above.

3.3.2 Experimentation with Noisy Data for Training dataset

The second set of experiments were conducted by adding artificial noise to the training data. Salt and Pepper noise was added with the intuition that flickering effect would resemble this kind of noise. The noise was added to 30% of the dataset and 70% of training data without noise on a set of 1000 images.

3.3.3 Experimentation with Normalization technique

The third scenario was changing the Instance Normalization [12] into Batch Normalization [13] with the same settings as code described in Section 3.2. This scenario is used to test the effect of normalization layer on Cycle GAN with respect to the image output produced by the network. This has been trained with 1000 images training data.

3.3.4 Experimentation with UNET architecture

The fourth scenario was changing the Cycle GAN generator with UNet architecture [2] combined with Instance Normalization [12] as shown in Figure 2. This scenario is used to check whether UNet architecture can produce better result than Cycle GAN generator and handle the flickering effect. The UNet architecture has been evaluated with Batch normalization and instance normalization.The learnings from previous experiments motivated us to use instance normalization instead of batch normalization as implemented originally in [2] . This has been trained with 1000 images training data.

3.4 Suggested Method

To deal with the flickering problem, [14] proposed a unified video-to-video translation framework with an integrated global and local mechanism. This mechanism achieves stability in video and ensures consistency of the location within different frames. Usually, the main solution for the unstable videos is to add a temporal consistency loss in the optimization. But, the temporal consistency loss only ensures frame-by-frame consistency which is a kind of frame-wise mechanism. It cannot capture the video level consistency. Thus, a whole evaluation of temporal consistency of the video is needed.

Hence [14] proposes a GAN framework which uses generator and discriminator based on Recurrent Neural Network (RNN) architecture. For generator, an RNN is combined with U-Net. The purpose of this combination is to consider the relation between adjacent frames. In this part, they minimize the temporal consistency loss to achieve local consistency. Different from the traditional GAN, for discriminator, they use two-channel discriminator. These two channels have a combination of RNN and Markovian discriminator. The first channel is traditional channel to encode the spatial relation. Another channel is to encode the residual error which is the temporal consistency for the entire video. The two-channel then ensures the global temporal consistency by minimizing the loss of conditional GAN.

4 Results

The results obtained has been analyzed qualitatively and quantitatively.

4.1 Qualitative Analysis

The qualitative performance of various experiments conducted could be inferred from Figure 3. The results are arranged in a top-to-bottom order starting from original image data followed by the results of various experiments. It could be inferred that experiment with 1000 and 5000 training images gives good reconstruction. This is because the first set of experiments conducted on 1000 images were sorted manually as good quality images. In an overall comparison, UNet architecture with Instance Normalization and Standard Cycle GAN seem to give a better reconstruction than other approaches.

4.2 Quantitative Analysis

On studying the literature for means to quantify the performance of a GAN, it is inferred that Inception score and Frechet Inception Distance (FID) are the two major criteria. As stated in [15] it is advised that Inception score is not a good measure to quantify the performance of GAN and hence FID has been used. It could be inferred from Figure 4 that experiment with 1000 and 5000 training images gives good reconstruction. It is also evident from Figure 5 that UNet architecture with Instance Normalization performs well with a set of 1000 images referred to as Normal Dataset. The performance of UNet could further improve with increase in the size of training data given. The Standard CycleGAN seems to have the lowest FID score which is expected to be the best performance. This gives evidence that the choice of training data is crucial for evaluating a GAN.

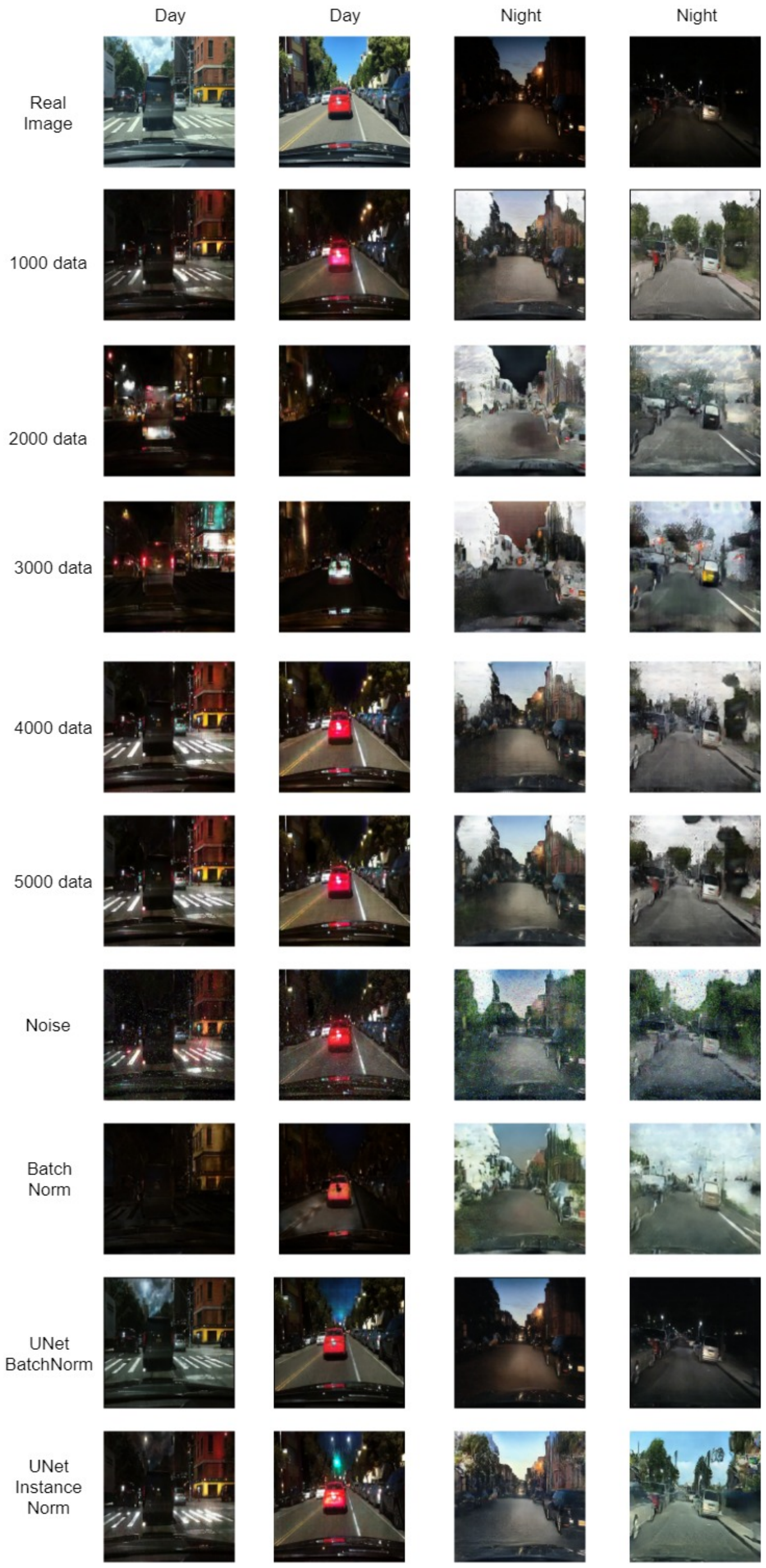


Figure 3: Result of Cycle GAN experiment

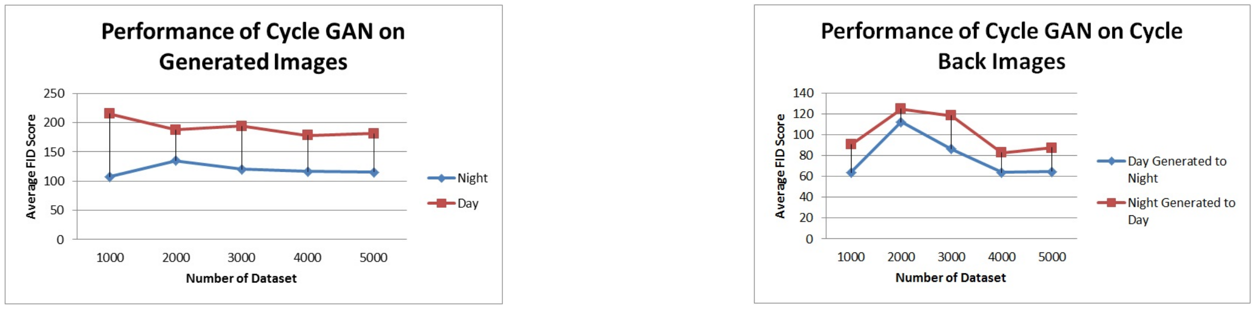


Figure 4: FID score w.r.t a) training samples b) training samples on cycle back

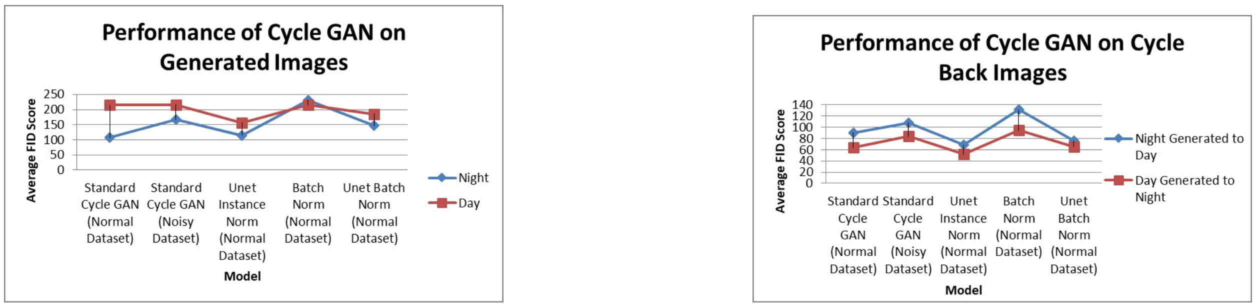


Figure 5: FID score w.r.t a) different experiments b) different experiments on cycle back

5 Discussion and Conclusion

The experiments reveal that UNet architecture with Instance normalization performs well on the given subset of training data. The results seem to be convincing but still there exists scope for improvement especially with the flickering problem that shows up in video-to-video translation.

Literature states many instances where flickering problem is common in the field of video stylization, video-to-video translation tasks, video segmentation, video super-resolution, and video colourization. This problem is caused by the movement of different frames that have different color and geography. There exists some interesting solutions for this in literature and one of them that we found interesting has been elaborated in Section 3.4.

References

[1] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” *CoRR*, vol. abs/1411.4038, 2014. [Online]. Available: http://arxiv.org/abs/1411.4038

[2] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” 2016. [Online]. Available: https://arxiv.org/abs/1611.07004

[3] Resales, Achan, and Frey, “Unsupervised image translation,” in *Proceedings Ninth IEEE International Conference on Computer Vision*, Oct 2003, pp. 472–478 vol.1.

[4] M. Liu and O. Tuzel, “Coupled generative adversarial networks,” *CoRR*, vol. abs/1606.07536, 2016. [Online]. Available: http://arxiv.org/abs/1606.07536

[5] Y. Aytar, L. Castrejón, C. Vondrick, H. Pirsiavash, and A. Torralba, “Cross-modal scene networks,” *CoRR*, vol. abs/1610.09003, 2016. [Online]. Available: http://arxiv.org/abs/1610.09003

[6] M. Liu, T. Breuel, and J. Kautz, “Unsupervised image-to-image translation networks,” *CoRR*, vol. abs/1703.00848, 2017. [Online]. Available: http://arxiv.org/abs/1703.00848

[7] J. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” *CoRR*, vol. abs/1703.10593, 2017. [Online]. Available: http://arxiv.org/abs/1703.10593

[8] J. Donahue, P. Krähenbühl, and T. Darrell, “Adversarial feature learning,” *CoRR*, vol. abs/1605.09782, 2016. [Online]. Available: http://arxiv.org/abs/1605.09782

[9] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, “Adversarially Learned Inference,” *arXiv e-prints*, p. arXiv:1606.00704, Jun. 2016.

[10] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, “Learning from simulated and unsupervised images through adversarial training,” *CoRR*, vol. abs/1612.07828, 2016. [Online]. Available: http://arxiv.org/abs/1612.07828

[11] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[12] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Instance normalization: The missing ingredient for fast stylization,” 2016. [Online]. Available: https://arxiv.org/abs/1607.08022

[13] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” 2015. [Online]. Available: https://arxiv.org/abs/1502.03167

[14] X. Wei, J. Zhu, S. Feng, and H. Su, “Video-to-video translation with global temporal consistency,” in *Proceedings of the 26th ACM International Conference on Multimedia*, ser. MM ’18. New York, NY, USA: ACM, 2018, pp. 18–25. [Online]. Available: http://doi.acm.org/10.1145/3240508.3240708

[15] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium,” *arXiv e-prints*, p. arXiv:1706.08500, Jun. 2017.