INDIAN INSTITUTE OF TECHNOLOGY ROORKEE

EEN - 400B: B.TECH PROJECT

Progress Report

Image-to-Image Translation with Generative Adversarial Networks

Project Members	Mentors
Ayush Goyal (16121007)	Dr. Vinay Pant
Naik Mayur Ravidas (16115075)	Dr. Yogesh Vijay Hote

Image-to-Image Translation with Generative Adversarial Networks

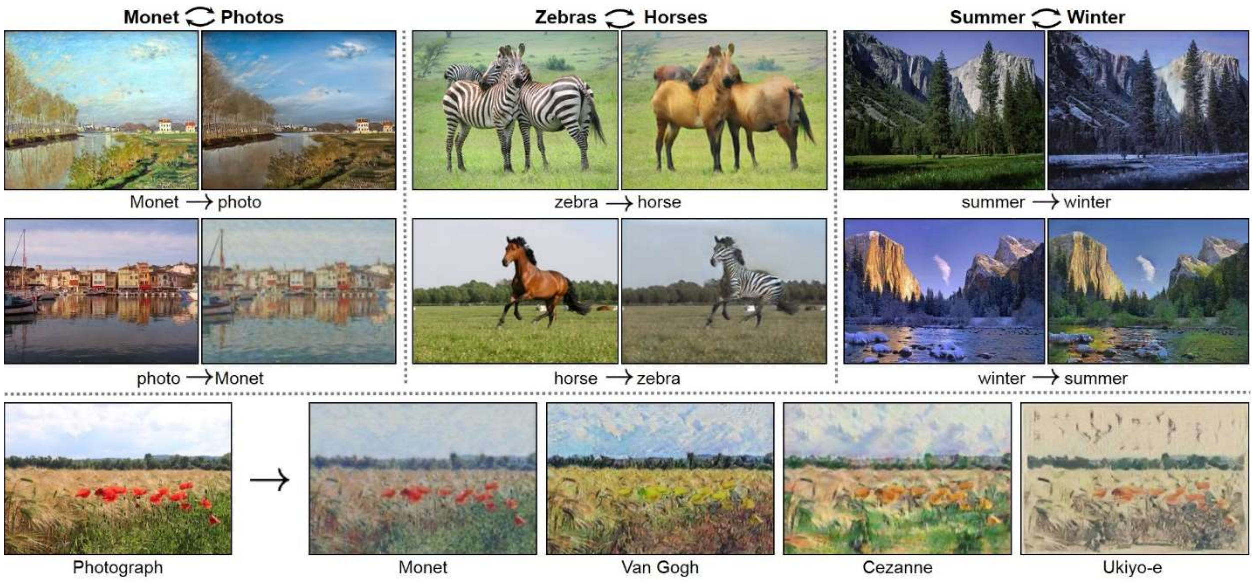

Figure 1: Image-to-Image Translation

Abstract

Image-to-Image translation is a class of vision and graphics where the aim is to create a mapping between an input image and an output image with the help of training dataset of aligned image pairs. The basic idea is the image from one domain is converted to another domain. However, for training the model, paired training datasets are necessary but unavailable. Generative Adversarial Networks of GANs are thus useful to create a general-purpose solution for this complication. The target of this project is to render an image from the source domain (say X) to a target domain (say Y) by training a mapping G:X→Y such that the distribution of images in G(x) cannot be distinguished from that of the distribution of images in the target domain Y. To achieve this, Adversarial Loss function is used. This method has a wide range of applications, most important of which is generating fake images, collection style transfer, transfiguration of objects, season transfer, enhancing a photograph, etc.

Introduction

What did Claude Monet see when he gazed upon the banks of the Seine near Argenteuil on a beautiful spring afternoon in 1873 (Figure 1, top-left)? If colored photography had been invented,

it would have documented a fresh blue sky and a shiny river reflecting it. Monet conveyed this beautiful scene that his eyes captured through wispy brush strokes and a bright color palette.
What if Monet happened to have looked upon the little harbor in Cassis on a cool summer evening (Figure 1, bottom-left)? We can imagine what Monet would have painted by taking a stroll across his paintings and rendering the scene: perhaps in pastel shades, with abrupt dabs of paint, and a somewhat flattened dynamic range.
All these scenarios can be imagined even though we have never actually seen them side by side i.e. Monet's painting next to a photograph of the scene he painted. Instead, we have the knowledge of the set of Monet paintings and the set of landscapes that we are interested in. We can realize and reason about the stylistic differences between these two sets, and thereby imagine what a scene might look like in Monet's style of painting i.e. if we were to "translate" the photograph to his style.

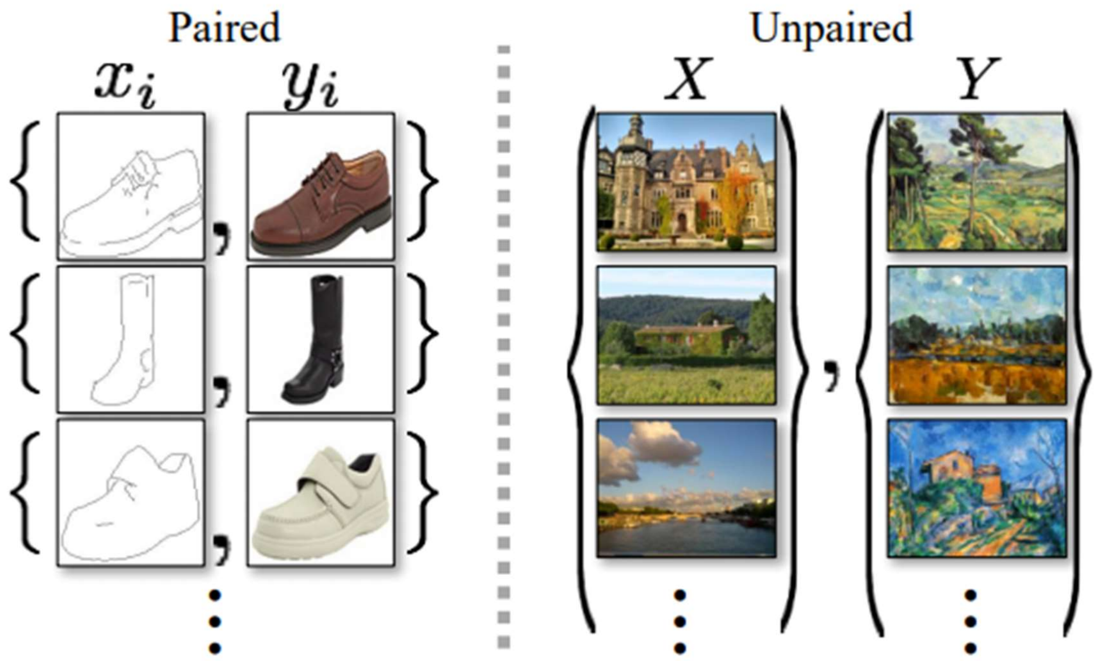

Figure 2: Paired training data (left) versus unpaired training data (right)

The aim of this project is to present a method is to learn to do the same i.e. capturing the aspects and characteristics of one image set and figuring out a way on how to translate these characteristics into another set of images where we don't know what is the target image (Absence of paired training examples).
The problem can be more broadly explained as Image-to-Image translation, i.e. converting an image from one representation of a given domain (say X), to another domain (say Y). Some examples of X and Y domains can be converting grayscale image to color, converting image to semantic labels, edge-maps to photograph. Years of research in Image processing, Computational photography, Computer vision and graphics have produced powerful translation systems in

supervised settings, where example image pairs $\{xi, yi\}^{N}_{i=1}$ are available (Figure 2, left). However, obtaining paired training dataset can be difficult and expensive. For example, we don't know what Monet saw in 1873 as we didn't have cameras to capture the image back then and thus very few datasets of tasks like semantic segmentation, and they are relatively small. Obtaining input-output pairs of graphic styling is even difficult to obtain since the desired output is highly complex, usually requiring artistic authorization. For many tasks like object transfiguration (e.g. Zebra $\leftrightarrow$ Horse, Figure 1 top-middle), the desired output is not even well-defined. We are just assuming the output based on the body structure and class of similar objects.
We therefore aim towards creating an algorithm which can translate between these domains without paired input-output examples (absence of target image; Figure 2, right). Some assumptions are made beforehand in order to work on these images that there is some underlying relationship between these domains. E.g. we assume that they are two different rendering of the same underlying scene and our aim is to seek and learn the relationship between them. Even though there is absence of paired examples, we can exploit supervision at the level of sets: we are given one set of images in domain X and a different set in domain Y. We are to train a mapping $G: X \rightarrow Y$ such that the output $y\hat{}$ $=G(x)$, $x \in X$, cannot be distinguished from images $y \in Y$ by an adversary trained to classify $y\hat{}$ apart from y. In theory, this objective can induce an output distribution over $y\hat{}$ that matches the empirical distribution P_{data} (y) (in general, this requires G to be stochastic). The optimal G thereby translates the domain X to a domain $Y\hat{}$ distributed identically to Y.

Related Works

Generative Adversarial Networks (GANs) [1] GANs have achieved impressive results in image generation, image editing and representation learning [3]. Recent methods adopt the same idea for conditional image generation applications, such as text2image, image inpainting, and future prediction, as well as to other domains like videos and 3D data. The main reason for the success of GANs is the idea of adversarial loss that forces the generated image to be, in principle, indistinguishable from the real image. The loss function is particularly a powerful tool as for image generation tasks, as this is exactly what the model is trained to reduce and thus punish the generator signal accordingly. This is the main objective that Computer graphics aims to optimize. The aim is to train a model using adversarial loss such that the translated images and the target domain images are indistinguishable. The most important use of this method is to create new data which is different from the available realistic data.

Neural Style Transfer This is another way of performing image to image translation, which creates a new image by combining features of one image with some style (typically a painting of a particular style) to another image. It is an optimization technique which takes a content image and a style image (E.g. artwork of a famous painter). It blends them together so that the output image looks like the content image but painted as the style image. This method is based on matching the Gram Matrix statistics of pre trained deep features. Our objective on the other hand, is learning the mapping between two image collections, rather than between two specific images, by trying to capture correspondences between higher-level appearance structures. Therefore, our method will be applicable to other tasks, such as painting→ photo, object transfiguration, etc. where single sample transfer methods do not perform well.

Data Collection and Pre-processing

1. **Cityscapes label ↔ Photo:**
   Training Set = Cityscapes
   Number of training Images = 2975.
   Size of the images = 128 × 128.
   We are going to use the Cityscapes val set for testing.
2. **Maps ↔ Aerial Photograph:**
   Training Set = scraped from Google Maps
   Number of training Images = 1096
   Size of the images = 256×256
   Images were sampled from in and around New York City. Data was then split into train and test about the median latitude of the sampling region (with a buffer region added to ensure that no training pixel appeared in the test set).
3. **Architectural Facades Labels ↔ Photo:**
   Number of Training Images = 400, from the CMP Facade Database.
4. **Edges → Shoes:**
   Number of Training Images = Around 50, 000, from UT Zappos50K dataset.
5. **Horse ↔ Zebra and Apple ↔ Orange:**
   Source of Images = ImageNet (using keywords wild horse, zebra, apple, and navel orange)
   Size of the images = 256×256 pixels.
   The training set size of each class: 939 (horse), 1177 (zebra), 996 (apple), and 1020 (orange).
6. **Summer ↔ Winter Yosemite:**
   Source of Images = downloaded using Flickr API with the tag yosemite and the date-taken field. Black-and-white photos were pruned.
   The images were scaled to 256 × 256 pixels.
   The training size of each class: 1273 (summer) and 854 (winter).
7. **Photo ↔ Art for Style Transfer:**
   Source of Images = Art images were downloaded from Wikiart.org. Some artworks that were sketches or too obscene were pruned by hand. The photos were downloaded from Flickr using the combination of tags landscape and landscape photography. Black-and-white photos were pruned.
   Size of the images = 256 × 256
   The training set size of each class was 1074 (Monet), 584 (Cezanne), 401 (Van Gogh), 1433 (Ukiyo-e), and 6853 (Photographs).

The Monet dataset was particularly pruned to include only landscape paintings, and the Van Gogh dataset included only his later works that represent his most recognizable artistic style.

8. **Monet's Paintings → Photos:**
   To achieve high resolution while conserving memory, we are going to use random square crops of the original images for training. To generate results, we are going to pass images of width 512 pixels with correct aspect ratio to the generator network as input.
9. **Flower Photo Enhancement:**
   Flower images taken on smartphones were downloaded from Flickr by searching for the photos taken by Apple iPhone 5, 5s, or 6, with search text flower. DSLR images with shallow DoF (Depth of Field) were also downloaded from Flickr by search tag flower, dof. The images were scaled to 360 pixels by width. The training set size of the smartphone and DSLR dataset were 1813 and 3326, respectively.

Applications

There are various applications of this method. Some of them are mentioned below

1. **Collection Style Transfer**

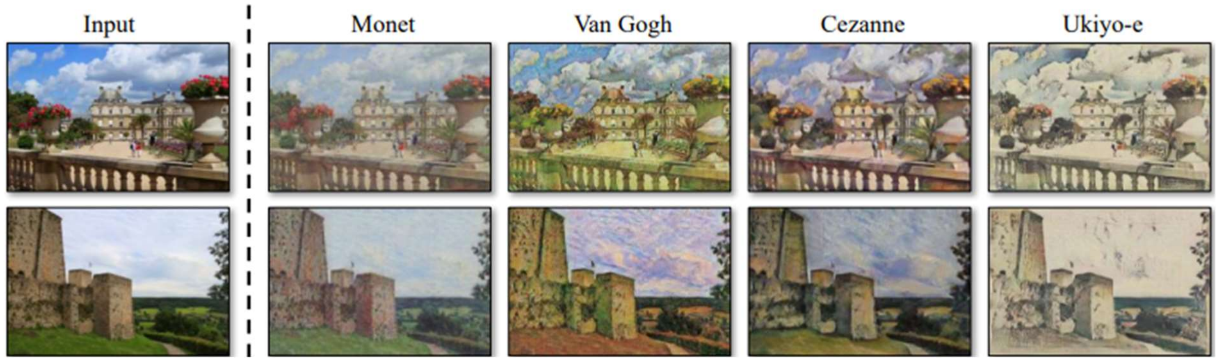

Figure 3: Collection Style Transfer

Collection Style Transfer refers to learning the artistic styles from one domain of paintings and applying it to another domain of paintings or real time photographs. Unlike recent work on "Neural Style Transfer", our aim is to mimic an entire collection of artworks, rather than transferring the styles of a single selected piece of art.

2. **Object Transfiguration**

Horse ↔ Zebra and Apple ↔ Orange are two typical examples of this method. The basic idea is to translate one domain to another domain having visually similar categories. E.g.

Horse cannot be transfigured to a rabbit and vice-versa. Similarly, Orange cannot be translated to a papaya and vice-versa.

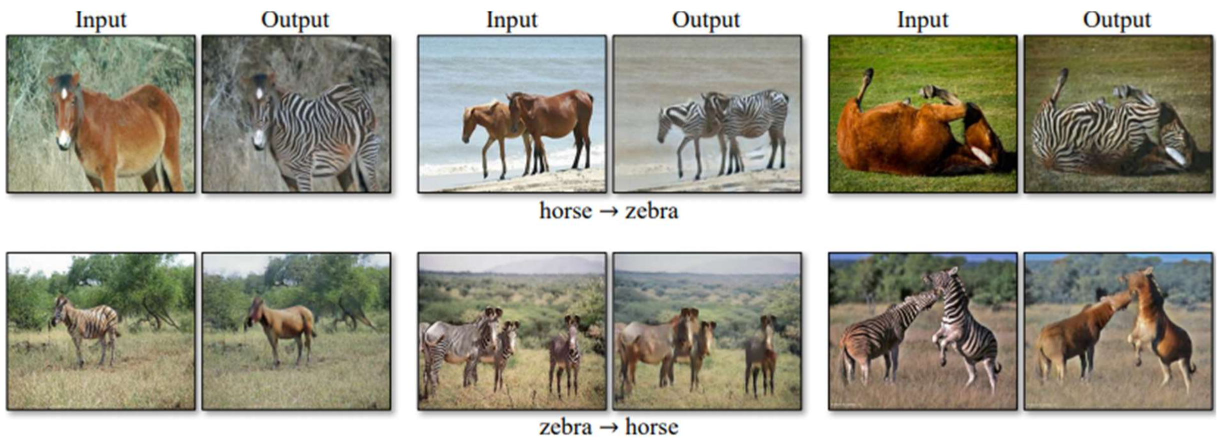

Figure 4: Object Transfiguration

3. **Season Transfer**

Season transfer basically means translation of photographs taken in one season to another and vice-versa. E.g. translation of a photograph taken in winter to a photograph in summer and vice-versa.

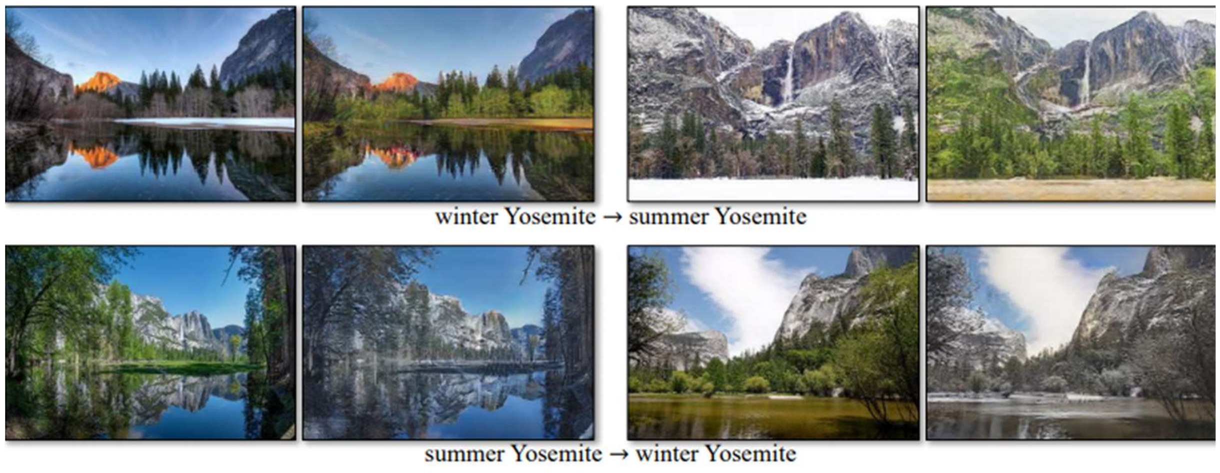
winter Yosemite → summer Yosemite
summer Yosemite → winter Yosemite

Figure 5: Season Transfer

4. **Photo generation from paintings**

Generally, all the famous painters' drawings of nature are pretty much non-existent as in the present. Thus, generation of photographs, as its name suggests, is generation of photo realistic images from a given painting or generating multiple styles of paintings of a particular scenery. Figure 6 shows some of the results obtained after using our technique for image-to-image translation.

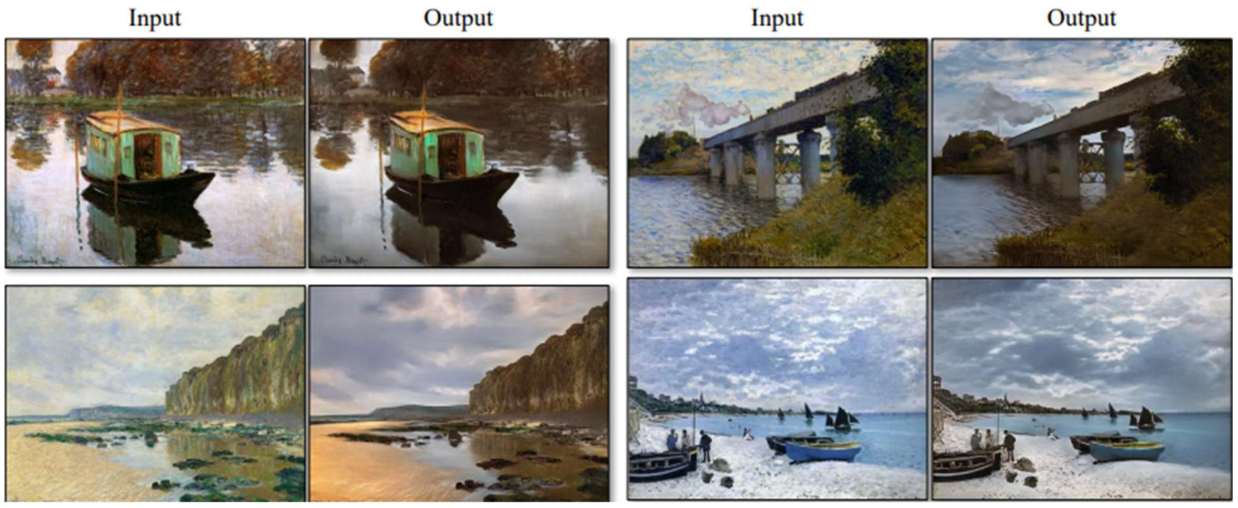

Figure 6: Photo generation from paintings

5. Photo Enhancement

Photo enhancement as its name suggests is improvement in the quality of the image in some or the other way. The main aspect of enhancing the image that we are interested in is improvement of the depth field (E.g. giving a macro effect) on close-up photographs of different flowers. Our method can successfully obtain the high resolution images for the existing blur photographs too.

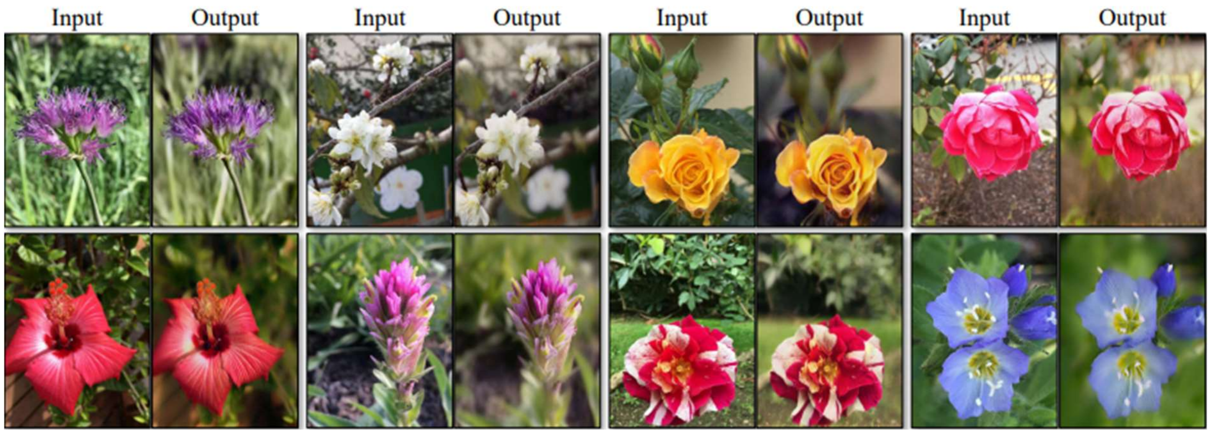

Figure 7: Photo Enhancement

Formulation

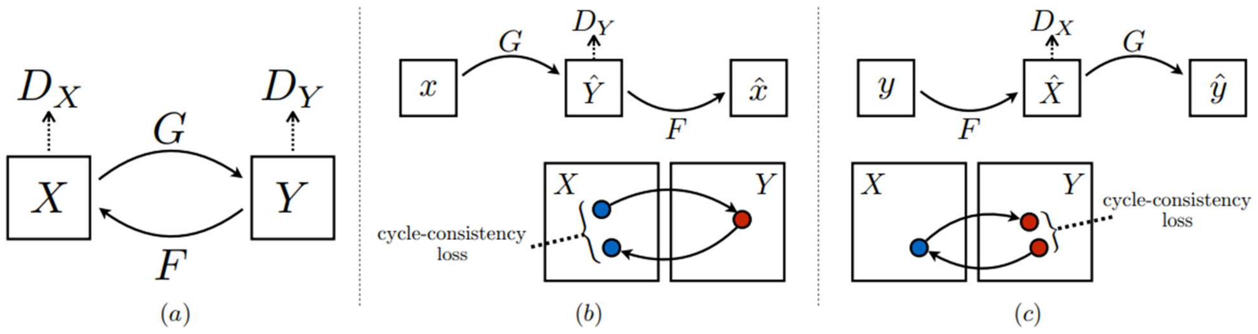

Figure 8: (a) We are using two generators in opposing directions namely G: X → Y and F: Y → X, along with discriminator networks D_Y and D_X. D_Y forces G to generate fake images from images belonging to domain X such that generated images are identical to images in domain Y, and vice versa for D_X and F. In order to constrain our problem, cycle consistency losses are being introduced in both the directions to control the quality of reconstructed image: (b) forward cycle consistency loss: x → G(x) → F(G(x)) ≈ x, and (c) backward cycle consistency loss: y → F(y) → G(F(y)) ≈ y.

As mentioned earlier, our aim is to train a mapping between two domains (say X and Y) given training samples $\{x_i\}^N_{i=1}$ where $x_i \in X$ and $\{y_j\}^M_{j=1}$ where $y_j \in Y$. The distribution of data for the two domains might be given as $x \sim p_{data}(x)$ and $y \sim p_{data}(y)$. We can see in Figure 8 (a) that the network aims to learn two kinds of relations in the opposite directions given by G: X → Y and F: Y → X where G and F are basically generators. Also, we can notice two discriminator networks which are essentially patchGANs [5] which we will talk later as D_X and D_Y, where D_X tries to separate the real images belonging to domain X from translated fake images obtained using generator network F; similarly, D_Y tries to separate the real images belonging to domain Y from translated fake images obtained using generator network G. The objective function has two parts: **adversarial losses** to minimize the difference between generated images and the original images; and **cycle consistency losses** so that G and F are not under constrained.

Adversarial Loss

For the generator G: X → Y along with its discriminator D_Y, the adversarial loss is given as follows:

$$\begin{aligned}\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) = &\ \mathbb{E}_{y\sim p_{\text{data}}(y)}[\log D_Y(y)] \\ &+ \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x))]\end{aligned}$$

Here the generator network G is encouraged to produce fake images G(x) which are identical to the images belonging to domain Y using the discriminator D_Y. At the same time discriminator network D_Y tries to discriminate between generated images G(x) and original images y. D tries to max it against the adversary G that aims to min the function given by,

$$\min_G \max_{D_Y} \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y)$$

Similarly, we can use another adversarial loss for learning generator network F: Y → X along with its discriminator D_X given by,

$$\min_F \max_{D_X} \mathcal{L}_{\text{GAN}}(F, D_X, Y, X)$$

Cycle Consistency Loss

Theoretically, we want to the learn the generators G and F which generate the output images identical to those belonging to target domains Y and X respectively. If the networks are large enough then, the input images may be mapped to any random permutation of images belonging to the target domain, where the distribution of generated images will match the targeted image distribution. That's why the adversarial losses can't guarantee that the mappings can translate the images correctly. In order to constraint our problem we want our network to be cycle-consistent: as shown in Figure 8 (b), for each image belonging to domain X, the translation cycle must be able to bring x back to the original image given by, x → G(x) → F(G(x)) ≈ x. We have named this forward cycle consistency. Similarly, as shown in Figure 8 (c), backward cycle consistency: y → F(y) → G(F(y)) ≈ y. For this we are introducing the cycle consistency loss:

$$\begin{aligned}\mathcal{L}_{\text{cyc}}(G, F) &= \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] \\ &+ \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1]\end{aligned}$$

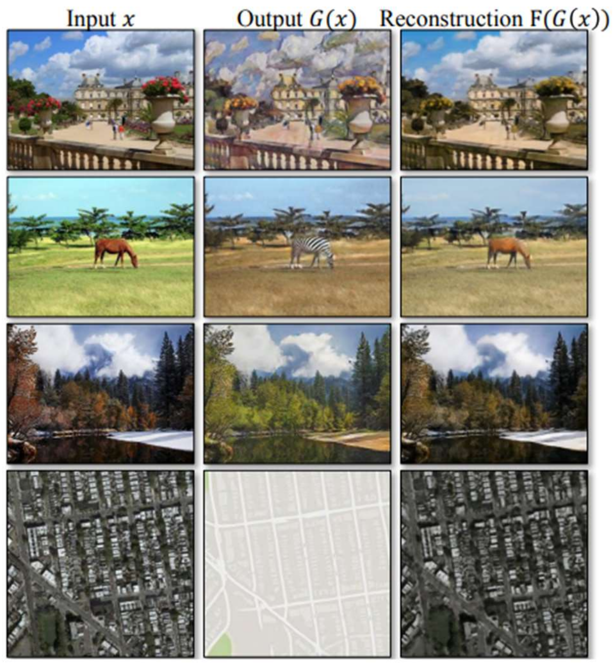

Figure 9: x denotes the real images, generated images denoted by G(x) and the reconstructed images are denoted by F(G(x)). Top to bottom: photo ↔ Cezanne, horses ↔ zebras, winter → summer Yosemite, aerial photos ↔ Google maps.

While performing various experiments, we also used L2 norm in place of the L1 norm along with the adversarial losses between F(G(x)) and x as well as between G(F(y)) and y, but no significant changes in the performance of the networks were observed.

Figure 9 shows the effects of the cycle consistency loss along with the adversarial loss. We can see that the reconstructed image F(G(x)) is identical to the input image x which verifies the use of this loss.

Full Objective

The full objective function of our suggested technique is given as follows:

$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) = &\mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\ &+ \lambda\mathcal{L}_{\text{cyc}}(G, F),\end{aligned}$$

Here, how much importance must be given to which objective whether adversarial or cycle-consistency is decided using the regularization factor lambda. Now we want to solve the following for G and F generators:

$$G^*, F^* = \arg\min_{G,F}\max_{D_x, D_Y} \mathcal{L}(G, F, D_X, D_Y)$$

We can see from the above equations that our network can be considered as using two variational "autoencoders" as shown in [8]: One in direction X → X and the other in direction Y → Y. We can also call it adversarial autoencoder shown in [9], which primarily use an adversarial loss as the name suggests in order to train the intermediate layer to mimic the target domain distribution.

Implementation

Network Architecture The architecture used by us for the generator to generate fake images is taken from [10] which shows good results for image styling and high-resolution image generation. The architecture contains two conv layers of stride 2, a few blocks from ResNet [11], and 2 fraction-stride conv layers having stride of 0.5. We use 5 residual blocks for images of resolution 128 × 128 and for images of 256 × 256 and high res train images we use 9 ResNet blocks. Similar to [10], we are using instance-norm in certain layers as shown in [12]. We are using PatchGANs [5] of size 70 x 70 as our networks for discriminator, in order to classify if 70 × 70 overlap patches of image are fake or real. We have used this patchGAN [5] approach for the discriminator because it reduces the number of parameters during training in comparison to a full resolution image discriminator network and hence making the training faster and thereby reducing overall training time of the network.

Training details For a training loss $L_{GAN}(G, D, X, Y)$, we are training the Generator to min $E_{x\sim pdata(x)}[(D(G(x)) - 1)^2]$ and training the Discriminator to min $E_{y\sim pdata(y)}[(D(y) - 1)^2] + E_{x\sim pdata(x)}[D(G(x))^2]$.

In order to limit the oscillations or the noise in the model [1], we are trying the strategy followed in [14] and using a buffer of fake images generated to update the parameters of the discriminator instead of the latest fake images generated. We are using the image history of 50 images.

We have used regularization factor (λ) as 10 for all our experiments we have performed. We have used Adam as our optimizer function as shown in [15] and also, we have used batch size of 1 due to gpu limitations. We have used the learning rate (alpha) = 2×10^{-4}. We have trained all networks from starting. The learning rate is kept constant for 100 epochs initially and after that we linearly reduce the alpha to 0 over remaining 100 epochs of a total of 200 epochs.

Results

We have achieved a per-pixel accuracy of 0.85 and a per-class accuracy of 0.40 along with class IoU (Intersection over Union) of 0.32 for Cityscapes (labels → photos) dataset.

The following is the plot of Generator, Discriminator and Cycle losses recorded during training after each epoch for 200 epochs.

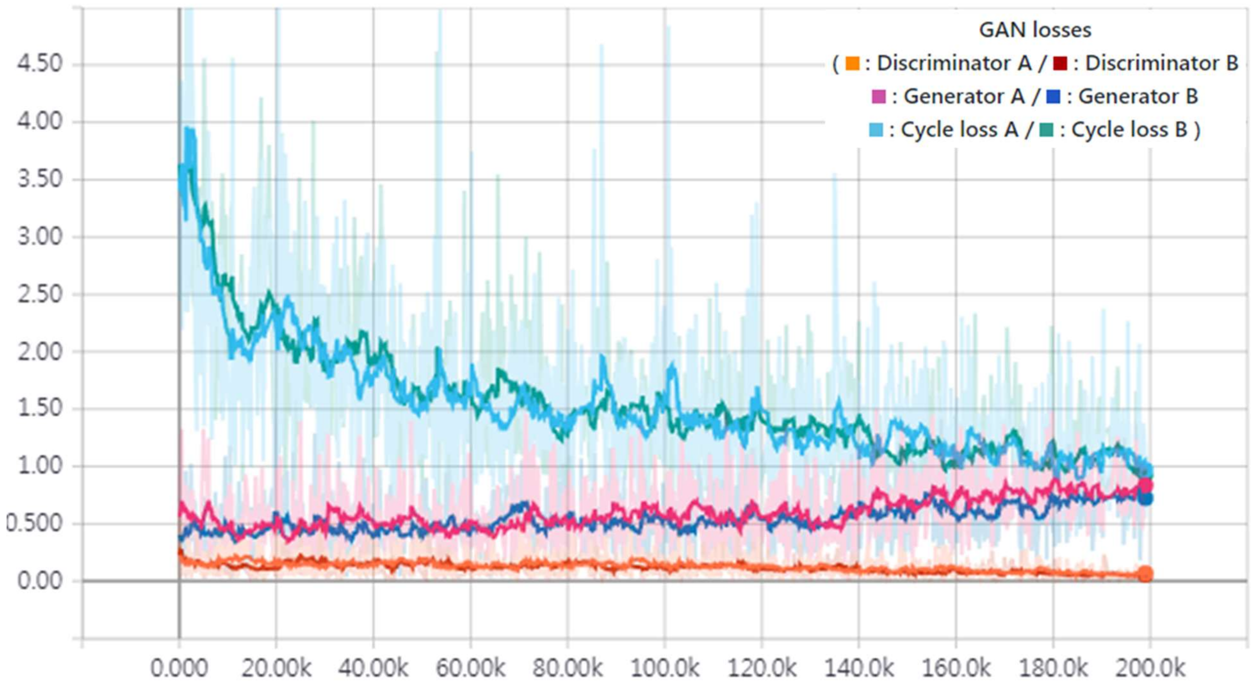

Figure 10: Plot of Generator, Discriminator and Cycle losses during training.

In the above plot, we can observe that the cycle losses are very high initially because the reconstructed image is nowhere near the original image and also the discriminators can easily identify the fake images from the real ones, but with time the network starts to produce decent image translations, thus helping in bringing down the cycle losses and making it difficult for the discriminators to identify the fake images.

Conclusion

The results achieved so far using our method/technique for translation suggests that introducing cycle loss along with adversarial loss is a promising approach for many image-to-image translation tasks, especially those involving highly complex graphical outputs. The suggested network learns a adapted loss function to the task and data available at hand, which makes this applicable for a variety of scenarios.

References

1. Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." In *Advances in neural information processing systems*, pp. 2672-2680. 2014.
2. Mirza, Mehdi, and Simon Osindero. "Conditional generative adversarial nets." *arXiv preprint arXiv:1411.1784* (2014).
3. Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." *arXiv preprint arXiv:1511.06434* (2015).
4. Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. "Image-to-image translation with conditional adversarial networks." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1125-1134. 2017.
5. Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks." In *Proceedings of the IEEE international conference on computer vision*, pp. 2223-2232. 2017.
6. Choi, Yunjey, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8789-8797. 2018.
7. Tang, Hao, Dan Xu, Nicu Sebe, and Yan Yan. "Attention-Guided Generative Adversarial Networks for Unsupervised Image-to-Image Translation." *arXiv preprint arXiv:1903.12296* (2019).
8. Hinton, Geoffrey E., and Ruslan R. Salakhutdinov. "Reducing the dimensionality of data with neural networks." *science* 313.5786 (2006): 504-507.
9. Makhzani, Alireza, et al. "Adversarial autoencoders." *arXiv preprint arXiv:1511.05644* (2015).
10. J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In ECCV, 2016.
11. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In CVPR, 2016.
12. D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022, 2016.
13. X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In CVPR. IEEE, 2017.
14. A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. In CVPR, 2017.
15. D. Kingma and J. Ba. Adam: A method for stochastic optimization. In ICLR, 2015.