CartoonGAN's Application and Optimize:Transform Real-World Image to Cartoon Form

Chuanyang Qiao
1005736385

Weiheng Wang
1006032730

Pao Hua Lin
1005263046

Abstract

This paper presents an in-depth analysis and implementation of CartoonGAN, a generative adversarial network designed to transform real-world photographs into stylized cartoon-like images. We explore the model's capabilities in producing high-quality results while retaining the original images' structural information. To achieve this, we investigate GANs, image-to-image translation techniques, and CartoonGAN's architecture components. We implement the model using the PyTorch deep learning framework and curate a diverse dataset for training and evaluation. [1]

1 Introduction

In recent years, the field of computer vision and image processing has seen significant advancements, particularly in generative models. One such development is CartoonGAN, a generative adversarial network (GAN) that transforms real-world images into cartoon-like representations. This ability has garnered widespread attention due to its potential applications in various industries such as animation. The rise of social media platforms are also interested in personalized and creative image filters.
The primary objective of this project is to implement and analyze CartoonGAN, exploring its capabilities in transforming real-world photographs into cartoon-style images. By leveraging deep learning techniques and GAN architecture, we aim to achieve high-quality and visually appealing results, capturing the essence of cartoon imagery while preserving the original photographs' structural information.

2 Related Works

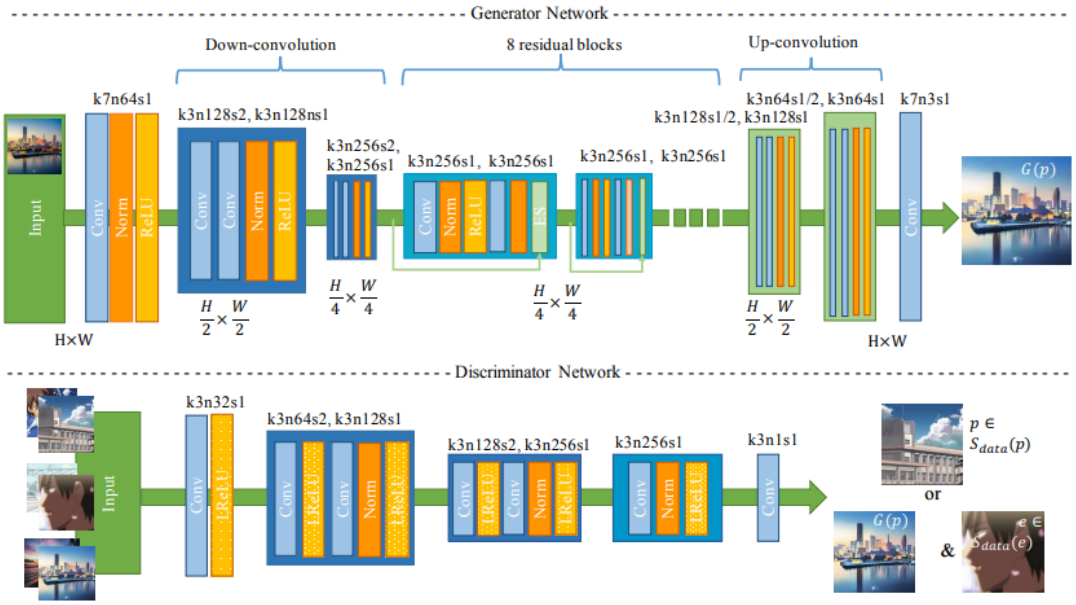
In particular, various GAN architectures have been developed for converting images into cartoon forms. One such approach is the use of $CycleGAN$[6], which can transform real-world images into cartoon-style images. However, by using CycleGAN the generated images tend to be blurry, and the training process can be time-consuming. Besides this,Wang Lei[7] used a Convolutional Neural Network (CNN) to transform images into cartoon forms. However, the computational complexity by using CNN is higher than CycleGAN, which may lead to longer processing times and require more resources.
In contrast, $CartoonGAN$[2] which was designed by Yang Chen offers a more targeted approach to generating cartoon-style images, by using a specific loss function designed to let the generator transform real-world images to Cartoon style. CartoonGAN generally requires less training time

[1] `https://github.com/Edward9292/CartoonGAN`

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

compared to CycleGAN and CNN based on the complexity of the network, making it a more efficient solution for generating cartoon-style images.

3 Algorithm And Model



Generator Network and Discriminator Network (From [1] Chen, Yang and Wang (2018)[1])

In this section, we provide a detailed analysis of the methods and algorithms employed in our project, focusing on the key components of the CartoonGAN architecture and the principles behind its operation.

Data Set

To facilitate the training and evaluation of the CartoonGAN model, we curated a diverse dataset comprising three different categories of images:

1. Cartoon Images (around 1,800): This set consists of cartoon images that conclude human, animal, environmental images, and so on.
2. Edges Smoothed Cartoon Images (around 1,800): This set contains the same images as the first category but has been processed using a Gaussian filter to smooth the edges to let the CartoonGAN model's edge-aware smoothing technique work better.
3. Real-world Photographs (around 2,000): This set mainly uses as input for the generator, after passing through the return result to the discriminator to decide if this result follows the expectation.

All three categories of images are used to train the discriminator and generator network and assess the loss during the training process.

Generator Network

The generator network in CartoonGAN is responsible for transforming input images into the desired cartoon-style output. It consists of three main stages: Encoding, Transformation, and Decoding.

1. Encoding: The input image is passed through a series of convolutional layers that extract a set of high-level feature maps.

2. Transformation: The high-level feature maps are then fed into a series of residual blocks that perform the actual style transformation.

3. Decoding: The transformed feature maps are passed through a series of transposed convolutional layers that upscale the image back to its original resolution.

Discriminator Network

The discriminator network serves as the adversarial counterpart to the generator, evaluating whether the generated images resemble the target cartoon style. It is composed of several convolutional layers that progressively downsample the input, eventually producing a scalar output.

Edge-aware Smoothing

To enhance the cartoon-like appearance of the generated images, CartoonGAN employs edge-aware smoothing. This technique promotes the formation of smooth color regions separated by well-defined edges, a characteristic often found in cartoon-style images.

Loss Functions

Here are the loss functions. Represent G as Generator, D as Discriminator, c_i is one of the cartoon image in our training set, e_j is corresponding cartoon image that had applied Gaussian noise to smoothed the edges. p_k, p_i are real-life photo image from training set:

$$\mathcal{L}(G, D) = \mathcal{L}_{adv}(G, D) + \omega \mathcal{L}_{con}(G, D)$$

$$\begin{aligned}\mathcal{L}_{adv}(G, D) = &\mathbb{E}_{c_i \sim S_{data}(c)}[\log D(c_i)] \\ &+ \mathbb{E}_{e_j \sim S_{data}(e)}[\log(1 - D(e_j))] \\ &+ \mathbb{E}_{p_k \sim S_{data}(p)}[\log(1 - D(G(p_k)))].\end{aligned}$$

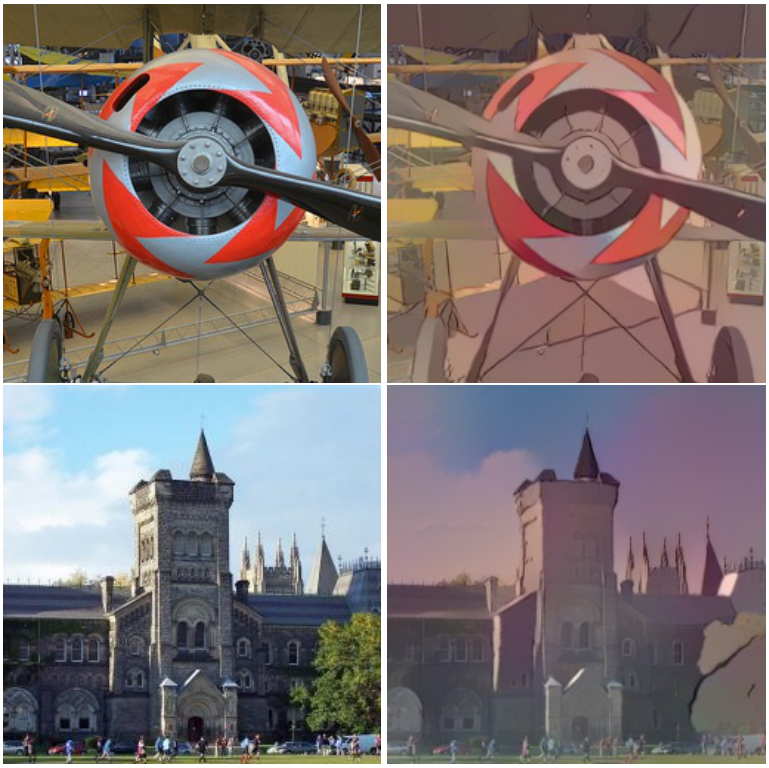
$$\begin{aligned}&\mathcal{L}_{con}(G, D) = \\ &\quad \mathbb{E}_{p_i \sim S_{data}(p)}[||VGG_l(G(p_i)) - VGG_l(p_i)||_1]\end{aligned}$$

Training Process

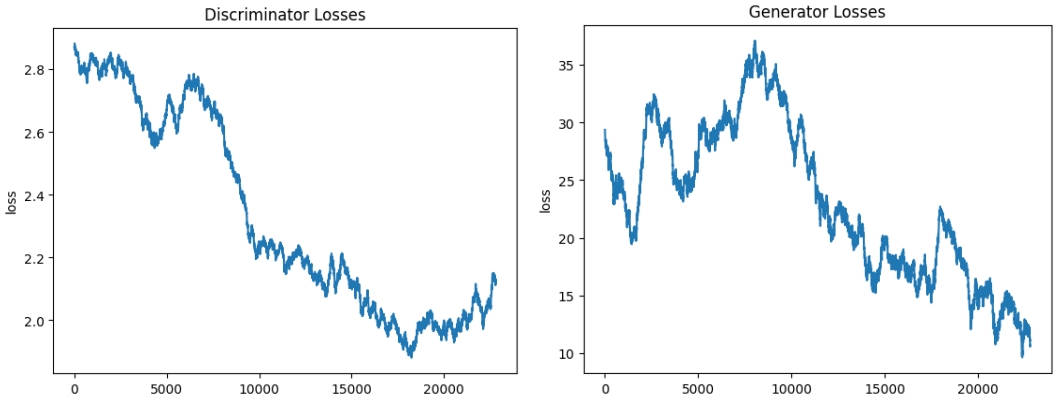
The CartoonGAN model is trained in an iterative manner, and it is updating the generator and discriminator networks alternately. The generator is trained to minimize the combined adversarial, content, and style losses, The discriminator is trained to maximize the adversarial loss. This adversarial training process goes until the networks reach an equilibrium, where the generator produces images that closely resemble the target cartoon style.

4 Result

By passing images through our trained generator model, As you can see, we successfully transformed real-life photo images into cartoon style images.



By storing the information of Discriminator loss and Generator loss in every iteration, we are able to analyze the change of the losses vs iteration as follow:



As you can see from the plots, it displays the discrimination loss on y-axis and a number of training iterations on x-axis. At the beginning of the training process, discrimination loss is relatively high as it struggles to differentiate between real images and generated cartoon images. As the training goes in process, the discriminator improves its ability to distinguish the two types of images. Similarly, the Generator loss on the y-axis and the number of iterations on the x-axis. At first, the generated image is far from a cartoon image, but as progress goes on, it is likely to be finally generated as a cartoon image.

5 Conclusion

Based on the result above, using CartoonGAN can transform the picture from real-world image to a Cartoon image easily and really stable. The special loss function defined in CartoonGAN successfully guide the model to reach our expectation.

Our study contributes to the growing body of knowledge in the field of generative models and image-to-image translation, providing valuable insights for future research and development. The successful application of CartoonGAN in this project opens up opportunities for its use in various

industries, such as animation, gaming, graphic design, and advertising, as well as personalized and creative image filters and transformations for social media platforms. We hope that our work will inspire further exploration of stylization techniques and the development of more advanced generative models in the future.

References

[1] Chen, Yang and Wang (2018). CartoonGAN: Generative Adversarial Networks for Photo Cartoonization. https://openaccess.thecvf.com/content_cvpr_2018/papers/Chen_CartoonGAN_Generative_Adversarial_CVPR_2018_paper.pdf

[2] Safebooru: Anime Image Dataset. https://www.kaggle.com/alamson/safebooru/download

[3] COCO Dataset Annotations (train/val 2017). http://images.cocodataset.org/annotations/annotations_trainval2017.zip

[4] Sunderdiek, Tobias. Cartoon-GAN. https://github.com/TobiasSunderdiek/cartoon-gan

[5] Shwerokade. Converting images to cartoon form using Cycle GAN in Keras. https://shwerokade.medium.com/converting-images-to-cartoon-form-using-cycle-gan-in-keras-540b6083b4f6

[6] Wang, Lei. Cartoon-Style Image Rendering Transfer Based on Neural Networks.