CartoonGAN: Generative Adversarial Networks for Photo Cartoonization

CSCI 1470 Final Project

Shiyu Liu, Sibo Zhou, Junhui Huang
Brown University

May 2, 2025

Introduction

The goal of our project is to transform real-world photographs into stylized cartoon images using deep learning. Inspired by the popularity of cartoon-style outputs from models like ChatGPT, we built our own cartoonization model based on the CartoonGAN architecture. Cartoonization involves simplifying complex visual content into clean lines, bold colors, and stylized features—a task that is difficult to achieve manually but well-suited for deep learning.

Our model learns this transformation in an unsupervised manner using unpaired image datasets, allowing for effective training without the need for one-to-one photo-cartoon mappings. We trained on 4000 real-world photos from the COCO dataset and 4000 cartoon-style images from the Safebooru anime dataset on Kaggle. This setup not only supports creative image stylization but also highlights the challenge of balancing content preservation with stylistic abstraction.

Methodology

The CartoonGAN architecture consists of a generator and a discriminator designed specifically for photo-to-cartoon translation. The generator transforms real-world photos into cartoon-style images using a deep convolutional neural network with three stages: initial convolution, downsampling through two strided convolution blocks, and eight residual blocks for feature transformation, followed by upsampling layers that reconstruct the image at its original resolution. To preserve both content and cartoon characteristics, the generator uses normalization and ReLU activations, finishing with a 7×7 convolution to produce the final output. The discriminator, on the other hand, is a shallow patch-based CNN that distinguishes between real cartoon images, generated outputs, and edge-smoothed cartoons. It employs Leaky ReLU activations and focuses on local style features like sharp edges and smooth shading. This design allows CartoonGAN to efficiently learn cartoon stylization from unpaired datasets while preserving content and generating high-quality, stylistically consistent outputs.

We train the CartoonGAN model using an adversarial learning framework with unpaired photo and cartoon datasets. The generator is first pretrained using only the content loss, which compares high-level feature maps between the input photo and the generated image via a network to preserve semantic content. After this initialization phase, both the generator and discriminator are trained jointly. The discriminator learns to distinguish real cartoon images from both edge-smoothed cartoons and generated outputs, using an edge-promoting adversarial loss that emphasizes cartoon-like characteristics such as sharp edges. The generator is updated to fool the discriminator while also minimizing content loss, striking a balance between stylistic transformation and content preservation. This training setup allows the model to learn cartoonization effectively without requiring paired image data.

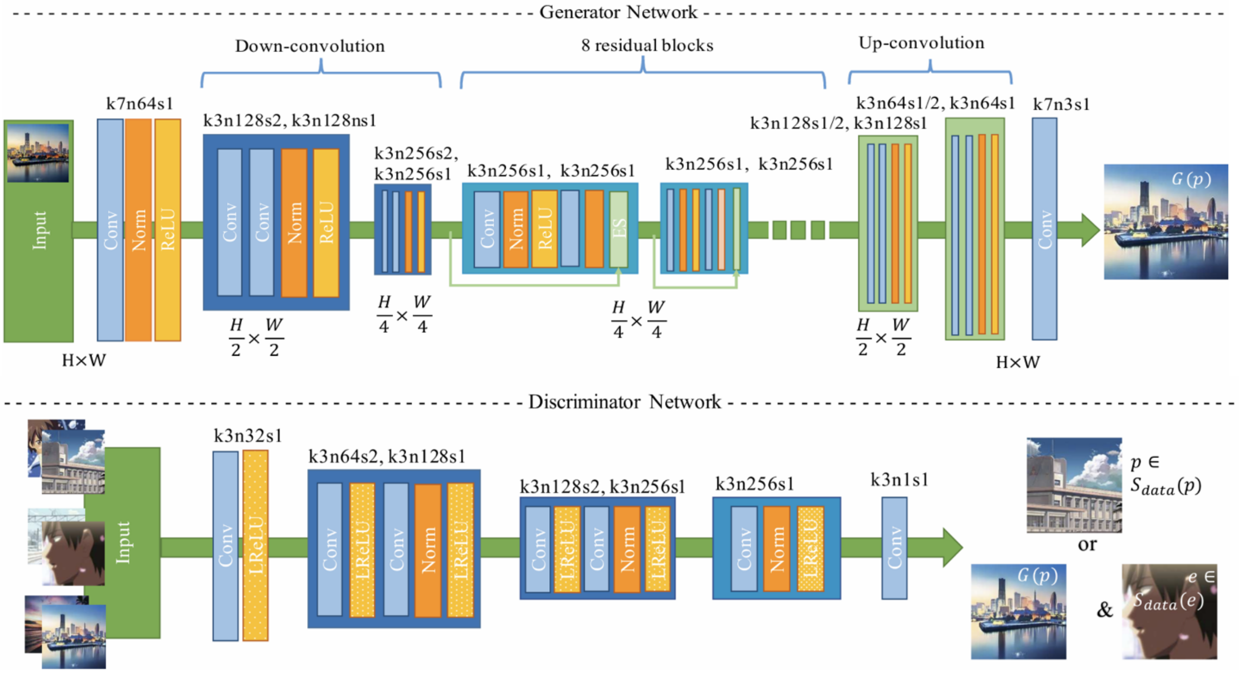


Figure 1. Model Structure Overview (Chen et. al 2018).

To faithfully reproduce the style of cartoons while preserving the underlying scene, we optimize a weighted sum of three losses each time we update the generator *G*:

- **Edge-Promoting Adversarial Loss.**

$$\mathcal{L}_{\text{adv}} = \log D(y) + \log(1 - D(G(x))) + \log(1 - D(y_s))$$

Following Chen et al. (2018), the discriminator *D* is trained on three inputs—real cartoons, edge-smoothed cartoons, and *G*'s outputs. Real cartoons are labeled 1, while both edge-smoothed and generated cartoons are labeled 0. This pushes *G* to generate images whose edge statistics match genuine cartoons and avoids trivial solutions where edges disappear.

- **Content Loss.**

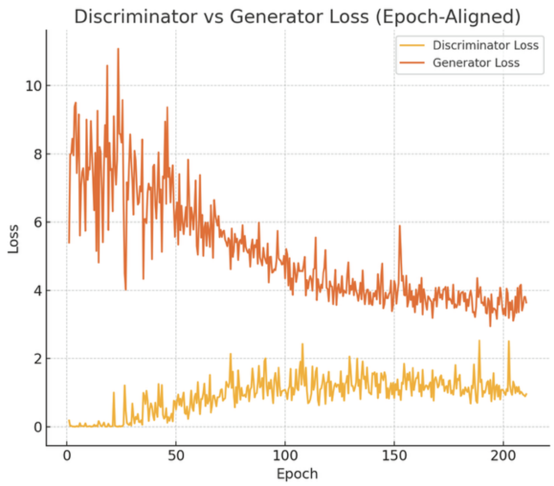
$$\mathcal{L}_{\text{content}} = \|\phi(G(x)) - \phi(x)\|_1$$

We pass the input photo *x* and its cartoonized counterpart *G(x)* through VGG (fixed weights) and compute an *L1* distance between their feature maps at layers *relu3_3* and *relu4_3*. This encourages high-level semantic structure (shapes, object layout) to remain unchanged after stylization.

Results

The results showcase visual comparisons between real-world photographs and their cartoonized outputs after 210 training epochs. The examples illustrate the model's ability to perform complex visual abstraction:

- Structural Consistency: Key objects such as people, animals, and backgrounds are preserved in position and form, indicating the model successfully maintains content.
- Stylistic Transformation: The outputs display characteristic cartoon features, including smoothed textures, simplified color palettes, and enhanced edges. This demonstrates that the model learns to suppress unnecessary photo details while emphasizing high-level shapes and contours.
- Artistic Coherence: The cartoonized images reflect a cohesive visual style, suggesting that the GAN model not only learns local features (like edges) but also global artistic patterns typical of cartoons.



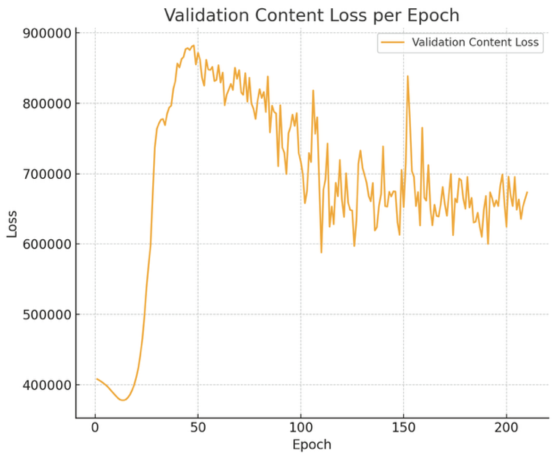


Figure 2. Discriminator vs Generator Loss. Figure 3. Validation Content Loss per Epoch.

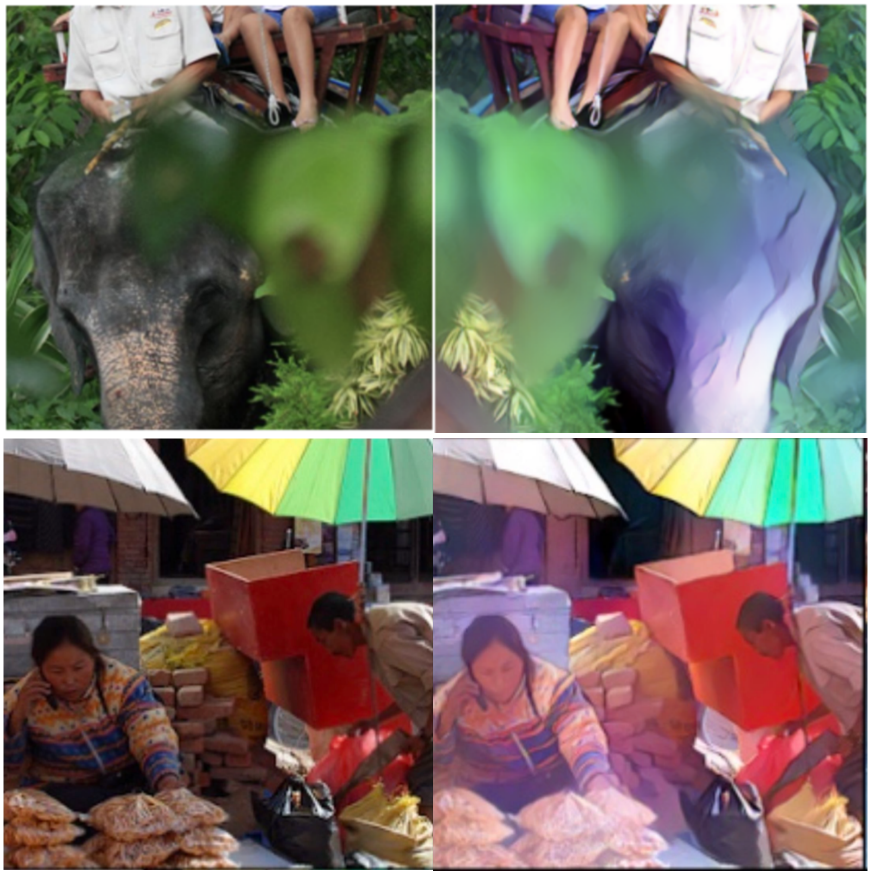


Figure 4. Sample after 210 training epochs (left:original photo, right: cartoonized images)

These qualitative results validate the effectiveness of the adapted CartoonGAN architecture, even in the face of challenges like limited data and training stability. The outputs confirm that the model achieves its goal of converting photos into artistically compelling cartoon images while preserving the essence of the original content.

Challenges

Throughout the development of this GAN model, we encountered several significant challenges. Debugging and stabilizing the training process required substantial effort, particularly in addressing issues like vanishing gradients, poor network convergence, and the inherent instability of adversarial training. Preparing a balanced dataset also proved complex—carefully curating 4,000 high-quality real-world images and 4,000 stylized cartoon images required additional preprocessing to ensure consistency and diversity across domains. One of the most persistent obstacles was achieving the right visual balance in the outputs: while we aimed for clear, bold outlines characteristic of cartoons, overly aggressive stylization often resulted in artifacts or unnatural transitions, requiring careful experimentation with loss functions and training parameters.

Hyperparameter tuning posed another major bottleneck. Training each model configuration took a long time, which made it difficult to systematically experiment with different settings such as loss weights and warmup durations. Limited GPU resources further constrained the scale of our tuning process. Additionally, evaluating our model's performance was a challenge in itself—since the output is a stylized image rather than a traditional classification or regression result, there was no clear or objective way to quantify accuracy. The original CartoonGAN paper does not specify a standardized evaluation metric, leaving us without a reference benchmark for visual quality assessment. Despite these challenges, our iterative experimentation led to visually compelling results that demonstrate the viability of our approach.

Reflection

We're satisfied with the outcome of our project, especially considering the scope and constraints. We successfully reached our base goal: we replicated the core CartoonGAN architecture and confirmed that it could generate visually convincing cartoon-style images on simpler datasets. The outputs demonstrated clear stylistic transformation and preserved essential content, which we validated through visual inspection. However, we were not able to meet our target or stretch goals due to time and resource limitations.

The model generally worked as expected, though there were challenges. As anticipated with GANs, we encountered instability during training and had to carefully tune hyperparameters and loss functions to promote convergence and balance between content preservation and stylization. Despite these hurdles, the outputs were visually convincing and aligned with our expectations of what cartoonized images should look like. One pleasant surprise was how well the model generalized to diverse photo inputs, even though we trained with limited cartoon training data. In short, the model's performance was in line with our expectations.

Knowing what we know now, particularly about the time cost of training and tuning, we would probably structure our experimentation phase differently. If we could start over, we'd allocate more upfront time specifically for setting up a more systematic way to explore hyperparameters, perhaps even trying to find or develop some rough quantitative measures (even if imperfect) to guide us earlier on. Relying solely on visual inspection after long training runs made the tuning process slow. Having more computational resources or a more structured tuning plan from the beginning might have helped us explore different configurations more efficiently.

With additional time, we see a few clear paths for improvement. We would definitely dive deeper into hyperparameter optimization – systematically testing different loss weights, maybe trying different optimizer settings or learning rate schedules – to push the visual quality further and potentially reduce some of the artifacts we encountered. Expanding the dataset, especially the cartoon image set, could also help the model learn more robustly or capture different stylistic nuances. Finally, exploring more advanced GAN stabilization techniques or minor architectural tweaks could potentially improve convergence and the overall consistency of the results, helping us reach the target or stretch goals we initially set.

This project gave us a very real appreciation for the challenges involved in working with GANs. We learned firsthand that training GANs requires patience and persistence due to issues like convergence problems and sensitivity to hyperparameters. Also, achieving the desired output involves a constant balancing act – in our case, between preserving the original photo's content and applying the cartoon style effectively without introducing undesirable artifacts. In terms of metrics, evaluating the "success" of a generative model like this is inherently subjective, especially without established quantitative benchmarks, making iterative improvement challenging. The quality and consistency of the training data are crucial, and preparing suitable datasets requires careful effort.

References

Chen, Y., Lai, Y.-K., & Liu, Y.-J. (2018). CartoonGAN: Generative Adversarial Networks for Photo Cartoonization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

Lamson, A. (2018). Safebooru Anime Image Dataset [Data set]. Kaggle. https://www.kaggle.com/alamson/safebooru