CartoonGAN: Generative Adversarial Networks for Photo Cartoonization

Yang Chen
Tsinghua University, China
chenyang15@mails.tsinghua.edu.cn

Yu-Kun Lai
Cardiff University, UK
Yukun.Lai@cs.cf.ac.uk

Yong-Jin Liu*
Tsinghua University, China
liuyongjin@tsinghua.edu.cn

Abstract

In this paper, we propose a solution to transforming photos of real-world scenes into cartoon style images, which is valuable and challenging in computer vision and computer graphics. Our solution belongs to learning based methods, which have recently become popular to stylize images in artistic forms such as painting. However, existing methods do not produce satisfactory results for cartoonization, due to the fact that (1) cartoon styles have unique characteristics with high level simplification and abstraction, and (2) cartoon images tend to have clear edges, smooth color shading and relatively simple textures, which exhibit significant challenges for texture-descriptor-based loss functions used in existing methods. In this paper, we propose CartoonGAN, a generative adversarial network (GAN) framework for cartoon stylization. Our method takes unpaired photos and cartoon images for training, which is easy to use. Two novel losses suitable for cartoonization are proposed: (1) a semantic content loss, which is formulated as a sparse regularization in the high-level feature maps of the VGG network to cope with substantial style variation between photos and cartoons, and (2) an edge-promoting adversarial loss for preserving clear edges. We further introduce an initialization phase, to improve the convergence of the network to the target manifold. Our method is also much more efficient to train than existing methods. Experimental results show that our method is able to generate high-quality cartoon images from real-world photos (i.e., following specific artists' styles and with clear edges and smooth shading) and outperforms state-of-the-art methods.

1. Introduction

Cartoons are an artistic form widely used in our daily life. In addition to artistic interests, their applications range from publication in printed media to storytelling for children's education. Like other forms of artworks, many famous cartoon images were created based on real-world scenes. Figure 1 shows a real-world scene whose corresponding cartoon image appeared in the animated film "Your Name". However, manually recreating real-world scenes in cartoon styles is very laborious and involves substantial artistic skills. To obtain high-quality cartoons, artists have to draw every single line and shade each color region of target scenes. Meanwhile, existing image editing software/algorithms with standard features cannot produce satisfactory results for cartoonization. Therefore, specially designed techniques that can automatically transform real-world photos to high-quality cartoon style images are very helpful and for artists, tremendous amount of time can be saved so that they can focus on more creative work. Such tools also provide a useful addition to photo editing software such as Instagram and Photoshop.

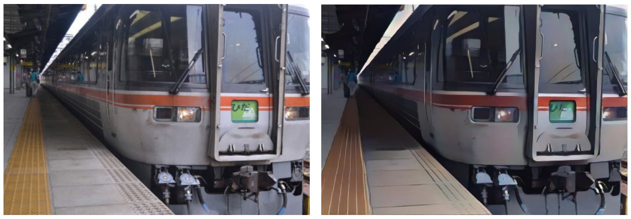

(a) Original scene (b) Our result

Figure 1. An example of cartoon stylization. (a) A real-world scene whose corresponding cartoon image appears in the animated film "Your Name". (b) Our result that transforms the photo (a) to the cartoon style. Note that our training data does not contain any picture in "Your Name".

Stylizing images in an artistic manner has been widely studied in the domain of non-photorealistic rendering [25]. Traditional approaches develop dedicated algorithms for specific styles. However, substantial efforts are required to produce fine-grained styles that mimic individual artists. Recently, learning-based style transfer methods (e.g. [6]), in which an image can be stylized based on provided examples, have drawn considerable attention. In particular, the power of Generative Adversarial Networks (GANs) [38] formulated in a cyclic manner is explored to achieve high-quality style transfer, with the distinct feature that the model is trained using unpaired photos and stylized images.

Although significant success has been achieved with

[*] Corresponding author

learning based stylization, state-of-the-art methods fail to produce cartoonized images with acceptable quality. There are two reasons. First, instead of adding textures such as brush strokes in many other styles, cartoon images are highly *simplified* and *abstracted* from real-world photos. Second, despite variation of styles among artists, cartoon images have noticeable common appearance — clear edges, smooth color shading and relatively simple textures — which is very different from other forms of artworks.

In this paper, we propose CartoonGAN, a novel GAN-based approach to photo cartoonization. Our method takes a set of photos and a set of cartoon images for training. To produce high quality results while making the training data easy to obtain, we do *not* require pairing or correspondence between two sets of images. From the perspective of computer vision algorithms, the goal of cartoon stylization is to map images in the photo manifold into the cartoon manifold while keeping the content unchanged. To achieve this goal, we propose to use a dedicated GAN-based architecture together with two simple yet effective loss functions. The main contributions of this paper are:

(1) We propose a dedicated GAN-based approach that effectively learns the mapping from real-world photos to cartoon images using unpaired image sets for training. Our method is able to generate high-quality stylized cartoons, which are substantially better than state-of-the-art methods. When cartoon images from individual artists are used for training, our method is able to reproduce their styles.

(2) We propose two simple yet effective loss functions in GAN-based architecture. In the generative network, to cope with substantial style variation between photos and cartoons, we introduce a semantic loss defined as an ℓ_1 sparse regularization in the high-level feature maps of the VGG network [30]. In the discriminator network, we propose an edge-promoting adversarial loss for preserving clear edges.

(3) We further introduce an initialization phase to improve the convergence of the network to the target manifold. Our method is much more efficient to train than existing methods.

2. Related Work

2.1. Non-photorealistic rendering (NPR)

Many NPR algorithms have been developed, either automatically or semi-automatically, to mimic specific artistic styles including cartoons [25]. Some works render 3D shapes in simple shading, which creates cartoon-like effect [28]. Such techniques called *cel shading* can save substantial amount of time for artists and have been used in the creation of games as well as cartoon videos and movies [22]. However, turning existing photos or videos into cartoons such as the problem studied in this paper is much more challenging.

A variety of methods have been developed to create images with flat shading, mimicking cartoon styles. Such methods use either image filtering [33] or formulations in optimization problems [35]. However, it is difficult to capture rich artistic styles using simple mathematical formulas. In particular, applying filtering or optimization uniformly to the entirely image does not give the high-level abstraction that an artist would normally do, such as making object boundaries clear. To improve the results, alternative methods rely on segmentation of images/videos [32], although at the cost of requiring some user interaction. Dedicated methods have also been developed for portraits [36, 26], where semantic segmentation can be derived automatically by detecting facial components. However, such methods cannot cope with general images.

2.2. Stylization with neural networks

Convolutional Neural Networks (CNNs) [17, 18] have received considerable attention for solving many computer vision problems. Instead of developing specific NPR algorithms which require substantial effort for each style, style transfer has been actively researched. Unlike traditional style transfer methods [11, 12] which require paired style/non-style images, recent studies [19, 1, 7, 8] show that the VGG network [30] trained for object recognition has good ability to extract semantic features of objects, which is very important in stylization. As a result, more powerful style transfer methods have been developed which do not require paired training images.

Given a style image and a content image, Gatys et al. [6] first proposed a neural style transfer (NST) method based on CNNs that transfers the style from the style image to the content image. They use the feature maps of a pre-trained VGG network to represent the content and optimize the result image, such that it retains the content from the content image while matching the texture information of the style image, where the texture is described using the global Gram matrix [7]. It produces nice results for transferring a variety of artistic styles automatically. However, it requires the content and style images to be reasonably similar. Furthermore, when images contain multiple objects, it may transfer styles to semantically different regions. The results for cartoon style transfer are more problematic, as they often fail to reproduce clear edges or smooth shading.

Li and Wand [20] obtained style transfer by local matching of CNN feature maps and using a Markov Random Field for fusion (CNNMRF). However, local matching can make mistakes, resulting in semantically incorrect output. Liao et al. [21] proposed a Deep Analogy method which keeps semantically meaningful dense correspondences between the content and style images while transferring the style. They also compare and blend patches in the VGG feature space. Chen et al. [3] proposed a method to improve comic style transfer by training a dedicated CNN to classify comic/non-comic images. All these methods use a single style image for a content image, and the result heavily depends on the chosen style image, as there is inevitable ambiguity regarding the separation of styles and content in the style image. In comparison, our method learns a cartoon style using two sets of images (i.e., real-world photos and cartoon images).

2.3. Image synthesis with GANs

An alternative, promising approach to image synthesis is to use Generative Adversarial Networks (GANs) [9, 34], which produce state-of-the-art results in many applications such as text to image translation [24], image inpainting [37], image super-resolution [19], etc. The key idea of a GAN model is to train two networks (i.e., a generator and a discriminator) iteratively, whereby the adversarial loss provided by the discriminator pushes the generated images towards the target manifold [37].

Several works [5, 14, 16] have provided GAN solutions to pixel-to-pixel image synthesis problems. However, these methods require paired image sets for the training process which is impractical for stylization due to the challenge of obtaining such corresponding image sets.

To address this fundamental limitation, CycleGAN [38] was recently proposed, which is a framework able to perform image translation with unpaired training data. To achieve this goal, it trains two sets of GAN models at the same time, mapping from class A to class B and from class B to class A, respectively. The loss is formulated based on the combined mapping that maps images to the same class. However, simultaneously training two GAN models often converges slowly, resulting in a time-consuming training process. This method also produces poor results for cartoon stylization due to the characteristics (i.e., high-level abstraction and clear edges) of cartoon images. As a comparison, our method utilizes a GAN model to learn the mapping between photo and cartoon manifolds using unpaired training data. Thanks to our dedicated loss functions, our method is able to synthesize high quality cartoon images, and can be trained much more efficiently.

2.4. Network architectures

Many works show that although deep neural networks can potentially improve the ability to represent complex functions, they can also be difficult to train because of the notorious vanishing gradient problem [29, 31]. The recently introduced concept of residual blocks [10] is a powerful choice to simplify the training process. It designs an "identity shortcut connection" which relieves the vanishing gradient issue while training. Models based on residual blocks have shown impressive performance in generative networks [15, 19, 38].

Another common way to ease the training of deep CNNs is batch normalization [13], which is designed to counteract the internal covariate shift and reduce the oscillations when approaching the minimum point. In addition, Leaky ReLu (LReLU) [23] is a widely used activation function in deep CNNs for efficient gradient propagation which increases the performance of networks by allowing a small, non-zero gradient when the unit is not active. We integrate these techniques in our cartoonization deep architecture.

3. CartoonGAN

A GAN framework consists of two CNNs. One is the generator G which is trained to produce output that fools the discriminator. The other is the discriminator D which classifies whether the image is from the real target manifold or synthetic. We design the generator and discriminator networks to suit the particularity of cartoon images; see Figure 2 for an overview.

We formulate the process of learning to transform real-world photos into cartoon images as a mapping function which maps the photo manifold $\mathcal{P}$ to the cartoon manifold $\mathcal{C}$. The mapping function is learned using training data $S_{data}(p) = \{p_i \mid i = 1 \ldots N\} \subset \mathcal{P}$ and $S_{data}(c) = \{c_i \mid i = 1 \ldots M\} \subset \mathcal{C}$, where N and M are the numbers of photo and cartoon images in the training set, respectively. Like other GAN frameworks, a discriminator function D is trained for pushing G to reach its goal by distinguishing images in the cartoon manifold from other images and providing the adversarial loss for G. Let $\mathcal{L}$ be the loss function, G^* and D^* be the weights of the networks. Our objective is to solve the min-max problem:

$$(G^*, D^*) = \arg \min_G \max_D \mathcal{L}(G, D) \tag{1}$$

We present the detail of our network architecture in Section 3.1 and propose two loss functions for G and D in Section 3.2. To further improve the network convergence, we propose an initialization phase and incorporate it into CartoonGAN, which is summarized in Section 3.3.

3.1. CartoonGAN architecture

Refer to Figure 2. In CartoonGAN, the generator network G is used to map input images to the cartoon manifold. Cartoon stylization is produced once the model is trained. G begins with a flat convolution stage followed by two down-convolution blocks to spatially compress and encode the images. Useful local signals are extracted in this stage for downstream transformation. Afterwards, eight residual blocks with identical layout are used to construct the content and manifold feature. We employ the residual block layout proposed in [15]. Finally, the output cartoon style images are reconstructed by two up-convolution blocks which contain fractionally strided convolutional layer with stride $1/2$ and a final convolutional layer with 7×7 kernels.

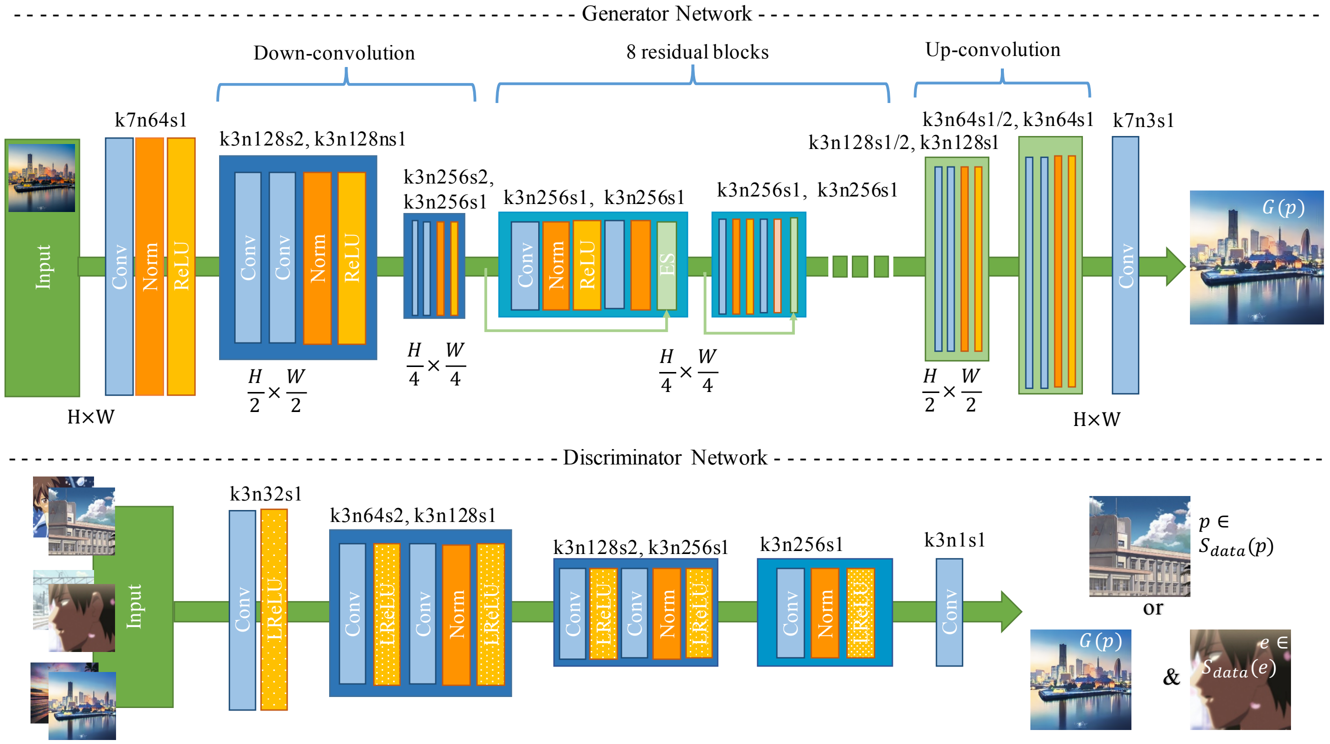


Figure 2. Architecture of the generator and discriminator networks in the proposed CartoonGAN, in which k is the kernel size, n is the number of feature maps and s is the stride in each convolutional layer, 'norm' indicates a normalization layer and 'ES' indicates elementwise sum.

Complementary to the generator network, the discriminator network D is used to judge whether the input image is a real cartoon image. Since judging whether an image is cartoon or not is a less demanding task, instead of a regular full-image discriminator, we use a simple patch-level discriminator with fewer parameters in D. Different from object classification, cartoon style discrimination relies on local features of the image. Accordingly, the network D is designed to be shallow. After the stage with flat layers, the network employs two strided convolutional blocks to reduce the resolution and encode essential local features for classification. Afterwards, a feature construction block and a 3×3 convolutional layer are used to obtain the classification response. Leaky ReLU (LReLU) [23] with $\alpha = 0.2$ is used after each normalization layer.

3.2. Loss function

The loss function $\mathcal{L}(G, D)$ in Eq.(1) consists of two parts: (1) the adversarial loss $\mathcal{L}_{adv}(G, D)$ (Section 3.2.1), which drives the generator network to achieve the desired manifold transformation, and (2) the content loss $\mathcal{L}_{con}(G, D)$ (Section 3.2.2), which preserves the image content during cartoon stylization. We use a simple additive form for the loss function:

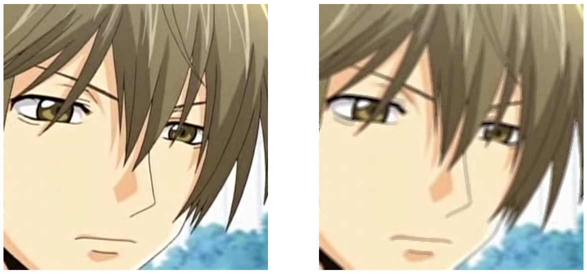
$$\mathcal{L}(G, D) = \mathcal{L}_{adv}(G, D) + \omega\mathcal{L}_{con}(G, D), \qquad (2)$$

where ω is the weight to balance the two given losses. Larger ω leads to more content information from the input photos to be retained, and therefore, results in stylized images with more detailed textures. In all our experiments, we set $\omega = 10$ which achieves a good balance of style and content preservation.

3.2.1 Adversarial loss $\mathcal{L}_{adv}(G, D)$

The adversarial loss is applied to both networks G and D, which affects the cartoon transformation process in the generator network G. Its value indicates to what extent the output image of the generator G looks like a cartoon image. In previous GAN frameworks [9, 14, 38], the task of the discriminator D is to figure out whether the input image is synthesized from the generator or from the real target manifold. However, we observe that simply training the discriminator D to separate generated and true cartoon images is not sufficient for transforming photos to cartoons. This is because the presentation of clear edges is an important characteristic of cartoon images, but the proportion of these edges is usually very small in the whole image. Therefore, an output image without clearly reproduced edges but with correct shading is likely to confuse the discriminator trained with a standard loss.

To circumvent this problem, from the training cartoon



(a) A cartoon image c_i (b) The edge-smoothed version e_i

Figure 3. By removing clear edges in a cartoon image $c_i \in S_{data}(c)$, we generate a corresponding image $e_i \in S_{data}(e)$.

images $S_{data}(c) \subset \mathcal{C}$, we automatically generate a set of images $S_{data}(e) = \{e_i \mid i = 1 \ldots M\} \subset \mathcal{E}$ by removing clear edges in $S_{data}(c)$, where $\mathcal{C}$ and $\mathcal{E}$ are the cartoon manifold and the manifold of cartoon-like images without clear edges, respectively. In more detail, for each image $c_i \in S_{data}(c)$, we apply the following three steps: (1) detect edge pixels using a standard Canny edge detector [2], (2) dilate the edge regions, and (3) apply a Gaussian smoothing in the dilated edge regions.

Figure 3 shows an example of a cartoon image and a modified version with edges smoothed out. Recall that for each photo p_k in the photo manifold $\mathcal{P}$, the generator G outputs a generated image $G(p_k)$. In CartoonGAN, the goal of training the discriminator D is to maximize the probability of assigning the correct label to $G(p_k)$, the cartoon images without clear edges (i.e., $e_j \in S_{data}(e)$) and the real cartoon images (i.e., $c_i \in S_{data}(c)$), such that the generator G can be guided correctly by transforming the input to the correct manifold. Therefore, we define the edge-promoting adversarial loss as:

$$\begin{aligned}\mathcal{L}_{adv}(G, D) = &\mathbb{E}_{c_i \sim S_{data}(c)}[\log D(c_i)] \\ &+ \mathbb{E}_{e_j \sim S_{data}(e)}[\log(1 - D(e_j))] \\ &+ \mathbb{E}_{p_k \sim S_{data}(p)}[\log(1 - D(G(p_k)))].\end{aligned} \tag{3}$$

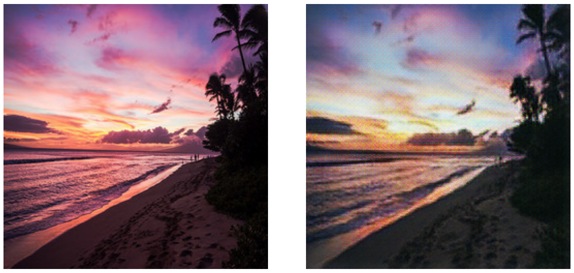
3.2.2 Content loss $\mathcal{L}_{con}(G, D)$

In addition to transformation between correct manifolds, one more important goal in cartoon stylization is to ensure the resulting cartoon images retain semantic content of the input photos. In CartoonGAN, we adopt the high-level feature maps in the VGG network [30] pre-trained by [27], which has been demonstrated to have good object preservation ability. Accordingly, we define the content loss as:

$$\begin{aligned}&\mathcal{L}_{con}(G, D) = \\ &\quad \mathbb{E}_{p_i \sim S_{data}(p)}[||VGG_l(G(p_i)) - VGG_l(p_i)||_1]\end{aligned} \tag{4}$$

where l refers to the feature maps of a specific VGG layer.

Unlike other image generation methods [6, 19], we define our semantic content loss using the ℓ_1 sparse regularization of VGG feature maps between the input photo and the generated cartoon image. This is due to the fact that cartoon images have very different characteristics (i.e., clear edges and smooth shading) from photos. We observe that even with a suitable VGG layer that intends to capture the image content, the feature maps may still be affected by the massive style difference. Such differences often concentrate on local regions where the representation and regional characteristics change dramatically. ℓ_1 sparse regularization is able to cope with such changes much better than the standard ℓ_2 norm. As we will show later, this is crucial to reproduce the cartoon style. We use the feature maps in the layer 'conv4_4' to compute our semantic content loss.



(a) Original photo (b) Image after initialization

Figure 4. For an original photo (a), the image (b) is the result after the initialization phase. See the main text for details.

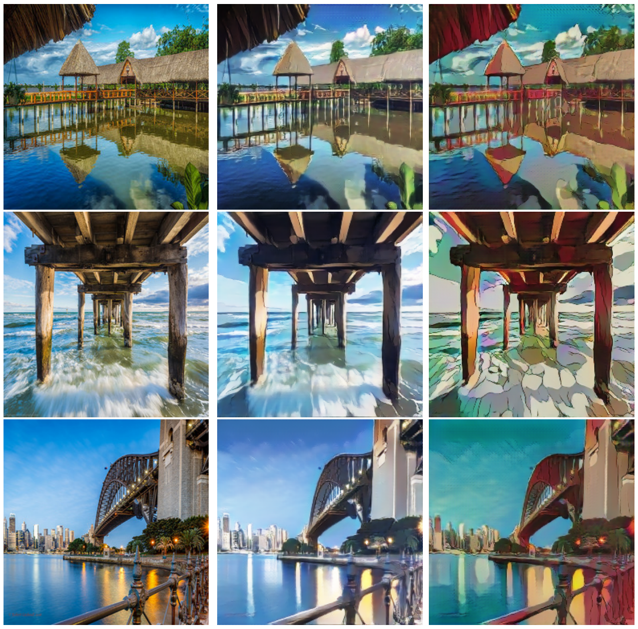
3.3. Initialization phase

Since the GAN model is highly nonlinear, with random initialization, the optimization can be easily trapped at sub-optimal local minimum. To help improve its convergence, we propose a new initialization phase. Note that the target of the generator network G is to reconstruct the input photo in a cartoon style while keeping the semantic content. We start the adversarial learning framework with a generator which only reconstructs the content of input images. For this purpose, in the initialization phase, we pre-train the generator network G with only the semantic content loss $\mathcal{L}_{con}(G, D)$. Figure 4 shows an example of the reconstructed image after 10 epochs of this initialization training phase, which already produces reasonable reconstruction. Our experimental results show that this simple initialization phase helps CartoonGAN fast converge to a good configuration, without premature convergence. Similar observation is made in [6] which uses the content image to initialize the result image to improve style transfer quality.

4. Experiments

We implemented our CartoonGAN in Torch [4] and Lua language. The trained models in our experiments are available[1] to facilitate evaluation of future methods. All experiments were performed on an NVIDIA Titan Xp GPU.

[1] http://cg.cs.tsinghua.edu.cn/people/~Yongjin/Yongjin.htm



(a) input photo (b) Shinkai style (c) Hayao style

Figure 5. Some results of different artistic styles generated by CartoonGAN. (a) Input real-world photos. (b) Makoto Shinkai style. (c) Miyazaki Hayao style.

CartoonGAN is able to produce high-quality cartoon stylization using the data of individual artists for training, which are easily obtained from cartoon videos, since our method does not require paired images. Different artists have their unique cartoon styles, which can be effectively learned by CartoonGAN. Some results of different artistic styles generated by CartoonGAN are shown in Figure 5.

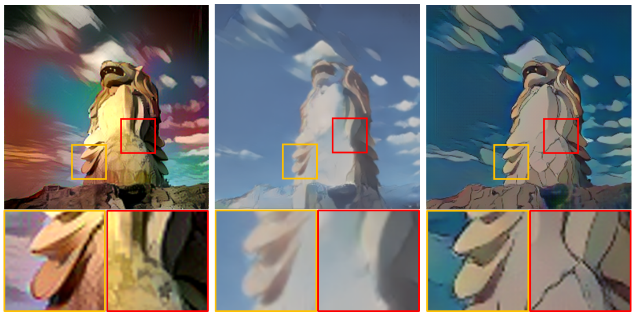
To compare CartoonGAN with state of the art, we collected the training and test data as presented in Section 4.1. In Section 4.2, we present the comparison between the proposed method and representative stylization methods. In Section 4.3, we present a further ablation experiment to analyze the effectiveness of each component in our CartoonGAN model.

4.1. Data

The training data contains real-world photos and cartoon images, and the test data only includes real-world photos. All the training images are resized and cropped to 256×256.

Photos. 6,153 photos are downloaded from Flickr, in which 5,402 photos are for training and others for testing.

Cartoon images. Different artists have different styles when creating cartoon images of real-world scenes. To obtain a set of cartoon images with the same style, we use the key frames of cartoon films drawn and directed by the same artist as the training data. In our experiments, 4,573 and 4,212 cartoon images from several short cartoon videos are used for training the Makoto Shinkai and Mamoru Hosoda style models, and 3,617 and 2,302 images from the cartoon film "Spirited Away" and "Paprika" are used for training the Miyazaki Hayao and "Paprika" style models.



(a) NST (b) CycleGAN (c) CartoonGAN

Figure 7. Details of edge generation. (a) The result of NST [6] using all the images in the training set as the style image. (b) CycleGAN [38] with the identity loss. (c) Our result.

4.2. Comparison with state of the art

We first compare CartoonGAN with two recently proposed methods in CNN-based stylization, namely NST [6] and CycleGAN [38]. Note that the original NST takes one style image I_s and one content image I_c as input, and transfers the style from I_s to I_c. For fair comparison, we apply two adaptations of NST. In the first adaptation, we manually choose a style image which has close content to the input photo. In the second adaptation, we extend NST to take all the cartoon images for training, similar to the comparative experiment in [38]. We also compare two versions of CycleGAN, i.e., without and with the identity loss $L_{identity}$. The incorporation of this loss tends to produce stylized images with better content preservation. 200 epochs were trained for both CycleGAN and our CartoonGAN.

Qualitative results are presented in Figure 6, which clearly demonstrate that NST and CycleGAN cannot deal with cartoon styles well (Figures 6b-6e). In comparison, by reproducing the necessary clear edges and smooth shading while retaining the content of the input photo, our CartoonGAN model produces hight-quality results (Figure 6f).

More specifically, NST [6] using only a style image may not be able to fully learn the style, especially for areas in the target image whose content is different from the style image (Figure 6b). When NST is extended to take more training data, rich styles can be better learned. However, the stylized images tend to have local regions stylized differently, causing inconsistency artifacts (Figure 6c).

The stylization results of CycleGAN do not capture the cartoon styles well. Without the identity loss, the output images do not preserve the content of the input photos well (Figure 6d). The identity loss is useful to avoid this problem, but the stylization results are still far from satisfactory (Figure 6e). In comparison, CartoonGAN produces high-

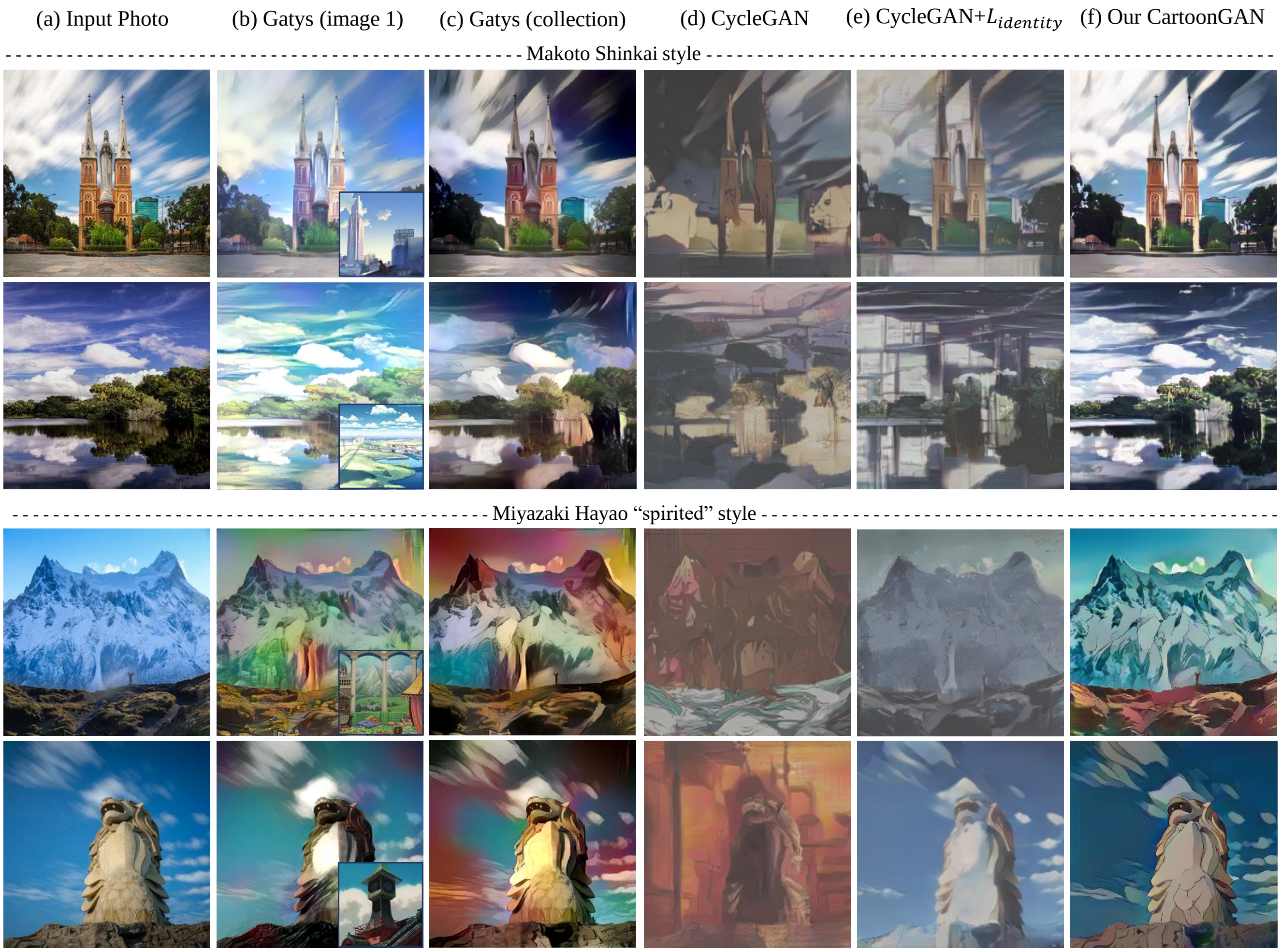


Figure 6. Comparison of CartoonGAN with NST [6] and CycleGAN [38] for Makoto Shinkai (top) and Miyazaki Hayao (bottom) styles. Gatys (image 1) and Gatys (collection) are two adaptations of NST where a cartoon image with close content to the input photo and all the cartoon images are used for training, respectively.

quality cartoonization which well follows individual artist’s style. Figure 7 shows close-up views of an example in Figure 6, demonstrating that our CartoonGAN generates the essential edges which are very important for the cartoon style.

Our CartoonGAN has the same property of not requiring paired images for training as CycleGAN. However, CartoonGAN takes much less training time. For each epoch, CycleGAN and CycleGAN with $L_{identity}$ take $2291.77s$ and $3020.31s$, respectively, whereas CartoonGAN only takes $1517.69s$, about half compared with CycleGAN + $L_{identity}$. This is because CycleGAN needs to train two GAN models for bidirectional mappings, which seriously slows down the training process. For image cartoonization, mapping back from cartoons to photos is not necessary. By using the VGG feature maps rather than a cycle architecture to restrain the content, CartoonGAN can learn cartoon stylization more efficiently.

We also compare our method with CNNMRF [20] and Deep Analogy [21], with Paprika and Mamoru Hosoda styles in Figure 8. Since both methods expect a single style image, two strategies are used to choose the style image from the training set: a manually selected cartoon image most similar in content with the input photo (image1) and a randomly picked cartoon image (image2). These methods fail to reproduce the characteristics of cartoon styles and produce results with clear artifacts, whereas our method produces high-quality stylization.

4.3. Roles of components in loss function

We perform the ablation experiment to study the role of each part in CartoonGAN. Figure 9 shows the examples of ablations of our full loss function, in which all the results are trained by Makoto Shinkai style’s data. The following results show that each component plays an important role in CartoonGAN. First, the initialization phase helps the generator G quickly converge to a reasonable manifold. As shown in Fig. 9b, without initialization, although some key features are shown, the styles are far from expectation. Sec-

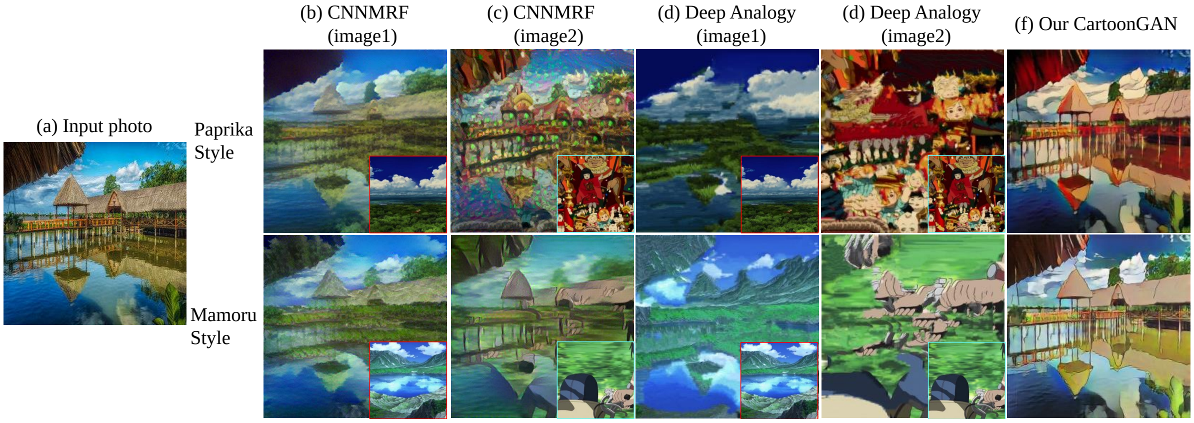


Figure 8. Cartoonization with 'Paprika' and Mamoru Hosoda styles, compared with CNNMRF [20] and Deep Analogy [21].

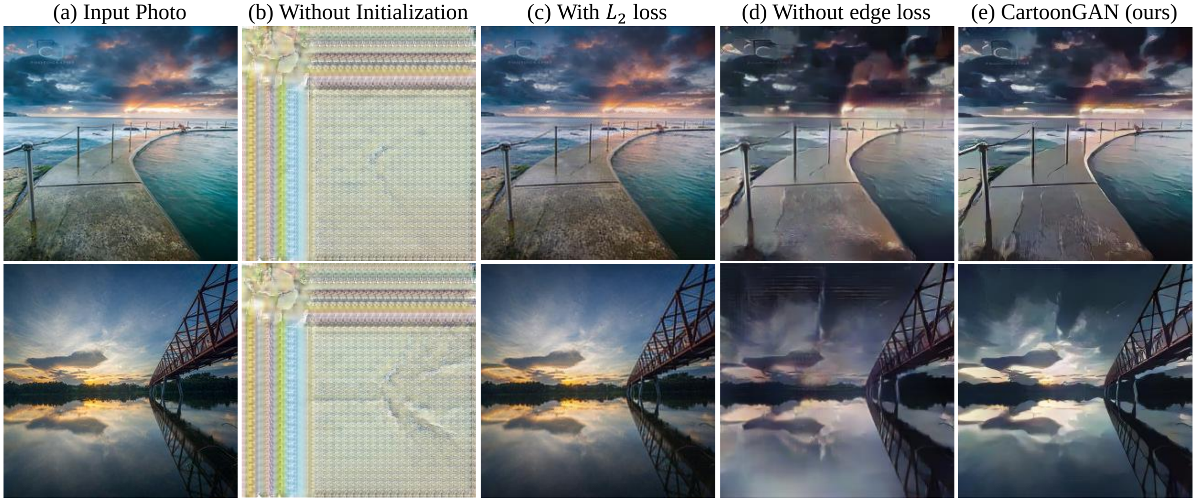


Figure 9. Results of removing/changing components in the loss function of CartoonGAN: (a) input photo, (b) without initialization process, (c) using ℓ_2 regularization for the content loss, (d) removing edge loss in the adversarial loss, (e) our CartoonGAN.

ond, even with a suitable VGG layer, large and often localized differences in feature maps of input and cartoon style images are still needed due to massive style differences. Using the ℓ_1 sparse regularization (instead of ℓ_2) of high-level VGG feature maps helps cope with substantial style differences between cartoon images and photos. Last, the elaborately designed edge loss guides the generator G to produce clear edges in results, leading to better cartoon style images.

5. Conclusion and Future Work

In this paper we proposed CartoonGAN, a Generative Adversarial Network to transform real-world photos to high-quality cartoon style images. Aiming at recreating faithful characteristics of cartoon images, we propose (1) a novel edge-promoting adversarial loss for clear edges, and (2) an ℓ_1 sparse regularization of high-level feature maps in the VGG network for content loss, which provides sufficient flexibility for reproducing smooth shading. We also propose a simple yet efficient initialization phase to help improve convergence. The experiments show that CartoonGAN is able to learn a model that transforms photos of real-world scenes to cartoon style images with high quality and high efficiency, significantly outperforming the state-of-the-art stylization methods.

In the future work, due to the importance of portrait, we would like to investigate how to exploit local facial features to improve cartoon stylization for human faces. Although we design our loss functions to tackle specific nature of cartoon stylization, similar ideas are useful for other image synthesis tasks, which we will investigate further. We also plan to add sequential constraints to the training process to extend our method to handling videos.

Acknowledgment

This work was supported by the Natural Science Foundation of China (61725204, 61521002, U1736220, 61661130156), Royal Society-Newton Advanced Fellowship (NA150431) and BNRist.

References

[1] J. Bruna, P. Sprechmann, and Y. LeCun. Super-resolution with deep convolutional sufficient statistics. In *International Conference on Learning Representations (ICLR)*, 2016.

[2] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):679–698, 1986.

[3] Y. Chen, Y.-K. Lai, and Y.-J. Liu. Transforming photos to comics using convolutional neural networks. In *International Conference on Image Processing*, 2017.

[4] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A MATLAB-like environment for machine learning. In *NIPS Workshop on BigLearn*, 2011.

[5] V. Dumoulin, I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. Courville. Adversarially learned inference. In *International Conference on Learning Representations (ICLR)*, 2017.

[6] L. Gatys, A. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2414–2423, 2016.

[7] L. A. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis and the controlled generation of natural stimuli using convolutional neural networks. *arXiv preprint arXiv:1505.07376*, 12, 2015.

[8] L. A. Gatys, A. S. Ecker, M. Bethge, A. Hertzmann, and E. Shechtman. Controlling perceptual factors in neural style transfer. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680. 2014.

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[11] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *ACM SIGGRAPH*, pages 327–340, 1998.

[12] S.-S. Huang, G.-X. Zhang, Y.-K. Lai, J. Kopf, D. Cohen-Or, and S.-M. Hu. Parametric meta-filter modeling from a single example pair. *The Visual Computer*, 30(6-8):673–684.

[13] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[14] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[15] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711, 2016.

[16] L. Karacan, Z. Akata, A. Erdem, and E. Erdem. Learning to generate images of outdoor scenes from attributes and semantic layouts. *arXiv preprint arXiv:1612.00215*, 2016.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[18] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back. Face recognition: A convolutional neural-network approach. *IEEE Transactions on Neural Networks*, 8(1):98–113, 1997.

[19] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[20] C. Li and M. Wand. Combining Markov random fields and convolutional neural networks for image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2479–2486, 2016.

[21] J. Liao, Y. Yao, L. Yuan, G. Hua, and S. B. Kang. Visual attribute transfer through deep image analogy. *ACM Transactions on Graphics*, 36(4):120, 2017.

[22] R. R. Luque. The cel shading technique. Technical report, 2012.

[23] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *International Conference on Machine Learning*, volume 30, 2013.

[24] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. In *International Conference on Machine Learning*, 2016.

[25] P. L. Rosin and J. Collomosse. *Image and Video-Based Artistic Stylisation*. Springer, 2013.

[26] P. L. Rosin and Y.-K. Lai. Non-photorealistic rendering of portraits. In *Workshop on Computational Aesthetics*, pages 159–170, 2015.

[27] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[28] T. Saito and T. Takahashi. Comprehensible rendering of 3-D shapes. In *ACM SIGGRAPH*, volume 24, pages 197–206, 1990.

[29] E. Simo-Serra, S. Iizuka, K. Sasaki, and H. Ishikawa. Learning to simplify: Fully convolutional neural networks for rough sketch cleanup. *ACM Transactions on Graphics*, 35(4):121, 2016.

[30] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[32] J. Wang, Y. Xu, H.-Y. Shum, and M. F. Cohen. Video tooning. *ACM Transactions on Graphics*, 23(3):574–583, 2004.

[33] H. Winnemöller, S. C. Olsen, and B. Gooch. Real-time video abstraction. *ACM Transactions on Graphics*, 25(3):1221–1226, 2006.

[34] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016.

[35] L. Xu, C. Lu, Y. Xu, and J. Jia. Image smoothing via L0 gradient minimization. *ACM Transactions on Graphics*, 30(6):174, 2011.

[36] M. Yang, S. Lin, P. Luo, L. Lin, and H. Chao. Semantics-driven portrait cartoon stylization. In *International Conference on Image Processing*, 2010.

[37] R. Yeh, C. Chen, T. Y. Lim, M. Hasegawa-Johnson, and M. N. Do. Semantic image inpainting with perceptual and contextual losses. *arXiv preprint arXiv:1607.07539*, 2016.

[38] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *International Conference on Computer Vision*, 2017.