CS484-585 Introduction to Computer Vision Project Paper
CartooNet

Ayhan Okuyan
Bilkent University
Ankara, Turkey
ayhan.okuyan@bilkent.edu.tr

Celal Bayraktar
Bilkent University
Ankara, Turkey
celal.bayraktar@ug.bilkent.edu.tr

Fatih Çakır
Bilkent University
Ankara, Turkey
fatih.cakir@ug.bilkent.edu.tr

Abstract

This paper proposes an attempt to improve and redesign the White-box Cartoonization framework proposed by Wang et al. *[21]. In our re-implementation, were able to point some deficiencies in the algorithm. In most of the outputs that contained detailed areas with constantly changing gradients, there were considerable amount of pixel noise. We decided that this stems from the guided filter that is used. Furthermore, we have noticed that the Frechet Inception Distance (FID) of the generated images are too close to the original photo distribution, which made the algorithm arguably too realistic. Hence, we propose two architectures in this paper that utilizes Invertible ResNet Blocks (IRBs) [3], and the use of guided filter more effectively.*
The proposed networks are compared against the original framework both quantitatively and qualitatively to validate the effectiveness of the proposed methods.

1. Introduction

Cartoon and anime is a popular form of art, widely watched by numerous people. The anime industry however, is highly dependent on the manual creation, which is a highly delicate area of work with requirements of substantial skill and talent. For content creators, it is a multi-step process that involves controlling shape, texture, color, shade and many other variables. Therefore, there exists work in literature to automate this process.
In this paper, we have worked on one of the algorithms to synthesize cartoonized images from the photographs and used the GAN-based framework proposed by Wang *et al.* [21] as baseline architecture. We have detected some parts that could be improved and opted for those improvements. We have observed some visible artifacts in the images with frequent gradient changes. To solve, we have altered the operations of the differentiable guided filter that is used in the algorithm to smooth out the images.
After observing the original FID results, we have noticed although the algorithm is the state-of-the-art, the output images FID to the real images were close and the images were too realistic. Hence, for more cartoon-like structure, we have opted for a generator design that uses Invertible ResNet blocks. This made the generator lighter in terms of memory and computational complexity, while also improving the style requirements.

2. Related Work

The idea of cartoon synthesis originates from the problem of Non-photorealistic Rendering(NPR). Basically, NPR methods represents image contents with artistic styles like pencil sketching and paints. Neural Style Transfer is one of popular NPR algorithms which uses two input images to synthesize content of one image and style of the other image. [8] Gatys et al. find out that content and style representations of image can be separable. Moreover, they can be trained jointly using convolutional neural networks(CNN).

Combining Markov Random Fields(MRF) and CNNs improves the results of facial artistic styles and photorealistic style synthesis.[15] Li and Wand achieve this by by replacing the bag-of-feature-like statistics of Gram-matrix-matching by an MRF regularizer that maintains local patterns of the “style” exemplar.

Chen et al. focuses on representing comics which typically contain minimalistic lines and shading.[5] Unlike the Gatys et al., Chen et al. initializes the optimization with the content image rather than the default white noise along with a higher weight to the style constraint, to further improve the results. Additionally, deep neural network trained for comic/photo classification, which is able to better extract comic style features.

Generative Adversarial Network (GAN) is a generative model that can generate samples according to distribution of input data by playing a minimax game with adversarial generative and discriminative networks[9]. The purpose of the generative network is to generate the data with same distribution as the training data and the purpose of the discriminative network is to discriminate whether the data is from training data or generated data. After many iterations, the generator can generate samples that cannot be distinguished by discriminator. GAN is frequently used in image generation. Image to image translations[25], image style transfer[24], super resolution techniques[14] are some applications of image generation with GAN.

Each problem may need different GAN to be solved. Conditional Generative Adversarial Networks(cGAN) generates samples according to conditions[16]. The cGAN is used for image to image translation in [12] because each image needs specific translation such as colorizing, reconstruction by edge map according to its conditions. This conditional translation calculates specific loss for each specific condition. In the loss function, it uses L1 distance because less blurring is obtained with L1 rather than L2. There is UNet based architecture for generator network and convolutional PatchGAN classifier for discriminator network[17].

CartoonGAN is one of the cartoonization method using generative adversarial network[6]. The main success of this work is to analyze unique style of training data and generating samples with this unique style. It uses two loss that are content loss and adversarial loss. Content loss is used to retain semantic content of the image. Adversarial loss is used to obtain clear edges.

3. Proposed Approach

This paper uses the White-box approach proposed in [21] as a baseline architecture and presents a two-phase improvement scheme that is based on the refined generator architecture and use of the guided filter.

3.1. Baseline Architecture

White-box architecture is designed to build a framework that is configurable. The difference from a black-box architecture is that the generator output is used in combination with known computer vision techniques to extract features, named as ”representations”. There are three of them in total, namely the structure, surface and texture representations which emphasize three important aspects of cartoon content creation.

3.1.1 Structure Representation

This represents the scheme where artist use sparse color blocks with a celluloid style to create the base for a cartoon art. To imitate this behavior, the paper proposes a workflow that uses the felzenszwalb algorithm [7], to segment the the image, then use selective search to create a more sparse segmentation map.
Then, it proposes an adaptive coloring algorithm to improve the global contrast in the resulting images. However, the author claims this approach is unstable and instead uses the simple superpixel algorithm that uses cluster means as the centroids. We have also used this approach when creating our models.

3.1.2 Texture Representation

The texture here is used as the high-frequency features of cartoon images. The author claims the easiness of learning the texture due to the color and luminance effects, and introduces a random shift algorithm that extract the grayscale representation of the image and randomly adds individual color channels to learn the underlying texture as follows.

$$\mathcal{F}_{rcs} = (1-\alpha)(\beta_1 * I_r + \beta_2 * I_g + \beta_1 * I_b) + \alpha * Y \quad (1)$$

where, in the default case, the β_i values are sampled from a uniform distribution with parameters as given below, without explicitly specifying the α value, which is the form of use in the implementation.

$$\mathcal{F}_{rcs} = \frac{(\beta_1 * I_r + \beta_2 * I_g + \beta_1 * I_b)}{\beta_1 + \beta_2 + \beta_3} \quad (2)$$

$$\beta_1 \sim U(0.199, 0.399) \quad (3)$$
$$\beta_2 \sim U(0.487, 0.687) \quad (4)$$
$$\beta_3 \sim U(0.014, 0.214) \quad (5)$$

3.1.3 Surface Representation

Surface representation imitates the cartoon style where the artist draw using coarse brushes lightly for sketching. To imitate this behavior, author proposes the use of a Differentiable Guided Filter [22], which is a fast and differentiable implementation of filter proposed by He *et al.* [10]. It is similar to the commonly used bilateral filter in the sense that it is an edge preserving filter, however the resulting images don’t contain reverse gradient artifacts as in the bilateral filter.

3.1.4 Loss Functions

With these three representations, each of them is connected to a separate discriminator to train the generator to learn

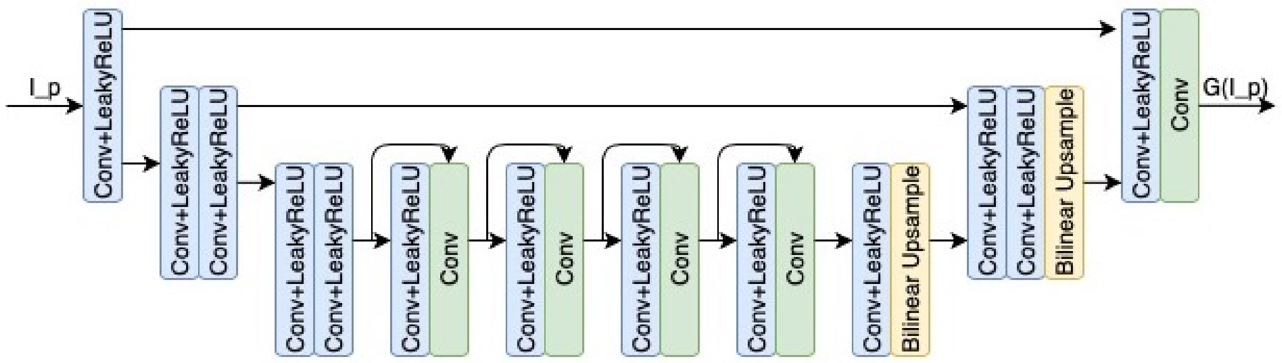

Figure 1. Generator of the White-box Architecture

them linearly independent of each other. There are in total of five loss functions that are simultaneously trained in the network, which are explained sequentially. The texture loss is defined as a standard binary logit loss as follows.

$$\begin{aligned}\mathcal{L}_{texture}(G, D_t) = &\log D_t(\mathcal{F}_{rcs}(I_c)) \\ &+ \log(1 - D_t(\mathcal{F}_{rcs}(G(I_p))))\end{aligned} \tag{6}$$

where, I_c is the cartoon image, and $G(I_p)$ is the output of the generator when input the training image I_p, trained with the discriminator D_t. The surface representation is learned similarly and trained with another discriminator D_s, as follows.

$$\begin{aligned}\mathcal{L}_{surface}(G,D_s) = &\log D_s(\mathcal{F}_{dgf}(I_c, I_c)) \\ &+ \log(1 - D_s(\mathcal{F}_{dgf}(G(I_p), G(I_p))))\end{aligned} \tag{7}$$

where, $\mathcal{F}_{dgf}$ stands for the differentiable guided filter, meaning the images are smoothed and hence, their surface representations are extracted by using the images themselves as filters.

For the structure loss $\mathcal{L}_{structure}$, the model uses a pre-trained VGG-19 [19] network. The Euclidean norm of the difference between the VGG-extracted generator output and the also the VGG-extracted structure representation is used so that the architecture learns the pixel-wise structural content. The loss is as follows.

$$\mathcal{L}_{structure} = ||VGG_n(G(I_p)) - VGG_n(\mathcal{F}_{st}(G(I_p)))|| \tag{8}$$

Furthermore, there are two other losses integrated into the architecture. One is the total variation loss [15], which is used to enforce smoothness on the output image and to decrease the salt and pepper noise. The loss ($\mathcal{L}_{tv}$) is defined as follows.

$$\mathcal{L}_{tv} = \frac{1}{H*W*C}||\nabla_x(G(I_p)) + \nabla_y(G(I_p))|| \tag{9}$$

Since this punishes high gradients in between any two pixel in the same color channel, the algorithm tries to minimize it, creating smoother outputs in the output. The other loss applied is the content loss, which ensures that the cartoonized images are semantically invariant as in Chen *et al*. [6]. The content loss is provided below.

$$\mathcal{L}_{content} = ||VGG_n(G(I_p)) - VGG_n(I_p)|| \tag{10}$$

Finally, the total loss is defined as the linear combination of the individual losses, whose weights are hyperparameters.

$$\begin{aligned}\mathcal{L}_{total} = &\lambda_1 * \mathcal{L}_{surface} + \lambda_2 * \mathcal{L}_{texture} \\ &+ \lambda_3 * \mathcal{L}_{structure} + \lambda_4 * \mathcal{L}_{content} + \lambda_5 * \mathcal{L}_{tv}\end{aligned} \tag{11}$$

The original hyperparameters that are assigned, which are kept constant for our implementation are given as follows.

$$\begin{aligned}\lambda_1 &= 1 \\ \lambda_2 &= 10 \\ \lambda_3 &= 2 * 10^3 \\ \lambda_4 &= 2 * 10^3 \\ \lambda_5 &= 10^4\end{aligned} \tag{12}$$

3.1.5 Architecture Details

The baseline architecture offers a U-Net [17] variant as the choice of generator. There are 5 **CONV-Leaky ReLU** layers with stride of 2 used for downsampling, which differs from the U-Net since it uses Max-Pooling layer for downsampling. 4 ResNet blocks with the configuration of **CONV-Leaky ReLU-CONV**, and a skip connection adding the input to the output to retain the information from the previous layers. For the upsampling, the architecture utilizes bilinear upsampling layers instead of the standard deconvolution. The baseline generator is presented in Figure 1.

The discriminator is a PatchGAN [12] discriminator, where the output is a convolutional layer. This means that each output neuron represents the binary output for an input patch.

3.2. Motivation

We have first trained the original White-box architecture to observe the model behavior, and there were two types of output behavior that we have decided to improve. First, we have observed the images that contains frequent high gradient pixels, the output images contained observable noise regions. A sample output can be observed in Figure 3.

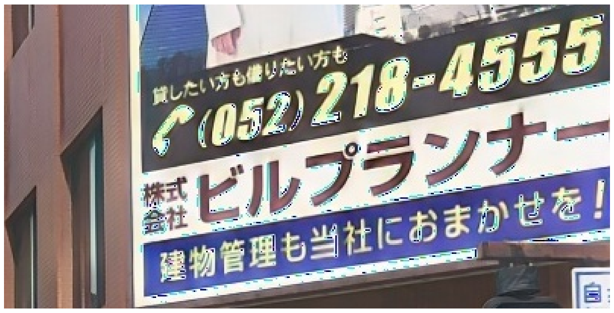

Figure 2. Observed artifacts in a sample White-box output

Furthermore, we the baseline paper is observed in terms of the Frechet Inception Distance (FID), we observed that the distance between the images and the cartoonized versions were closer than any other state-of-the-art model. With further observation, we have found that qualitatively, the images that are created, although highly relevant in terms of context, were too realistic. Therefore, our second objective was to increase the style content of the created cartoons.

3.3. Inverted ResNet Blocks

In order to increase the style content, we have introduced the Inverted ResNet Blocks (IRBs) [3] to the architecture as a novelty, which are smaller in parameter size. An IRB, is a series of convolutional block that takes advantage of separable convolutions combined with some form of normalization. We have adapted the architecture presented by Chen *et al.* in [4]. The used IRB is as follows.

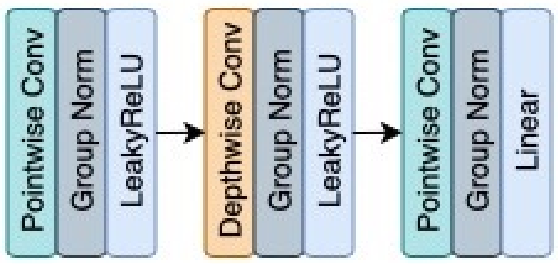

Figure 3. IRB Block

From figure, we can observe that the layer consists of three convolutional layers with fewer parameters than the original generator. Furthermore, we have used the group normalization [23] as choice of normalization instead of batch or instance normalization due to the low batch size we have trained the algorithm with to represent the data better. Note that there is no activation present in the last point-wise convolution layer.

3.4. White-box IRB

White-box IRB is the referred name of the architecture that uses IRB blocks instead of ResNet blocks, and this is the first iteration of improvement that we have proposed in this paper. Without changing the down and upsampling layers, we have placed four IRB blocks as the middle section, followed by a **CONV-Group Norm-Leaky ReLU**, which is the same with the first convolutional layer inside the IRB block. The resulting architecture is presented in Figure 4.

The first IRB block does not contain a skip connection since its feature map size doesn't change of the output and input size don't match. For that, not only we have decreased the parameter size in the network, making it lighter, we have opted for a more cartoon-ish style in the output.

3.5. Guided Filter Optimization

After further investigation and parameter tuning, we have observed that the artifacts are caused by the guided filter that is used to sharpen the output. The guided filter is used in the baseline architecture to sharpen the output as follows directly after the generator output.

$$\mathcal{I}_{interp} = \delta * \mathcal{F}_{dgf}(I_{in}, G(I_{in})) + (1 - \delta) * G(I_{in}) \quad (13)$$

Where δ is a mixing coefficient as is used originally as 1. To remove the artifacts, we have removed the differentiable guided filter from the training pipeline. While keeping the loss functions the same as before, we have removed the initial filtering.

$$\mathcal{I}_{interp} = G(I_{in}) \quad (14)$$

Furthermore, we have increased the ϵ coefficient from $5 * 10^{-3}$ to $5 * 10^{-2}$ in the guided filter in inference. This has caused some amount of loss in the sharpness, however both these changes combined resulted with the removal of the artifacts. The changes we presented here used in combination with the IRB blocks constructed the second iteration of improvements, we have made, which is referred to as the **White-box IRGF** throughout this paper. These are the novel changes that are presented in this project paper.

4. Results & Experiments

4.1. Dataset Details

For training and test, we have used the dataset that is used in the baseline architecture, which consists of four parts. There are 10000 real face and 5000 real scenery RGB

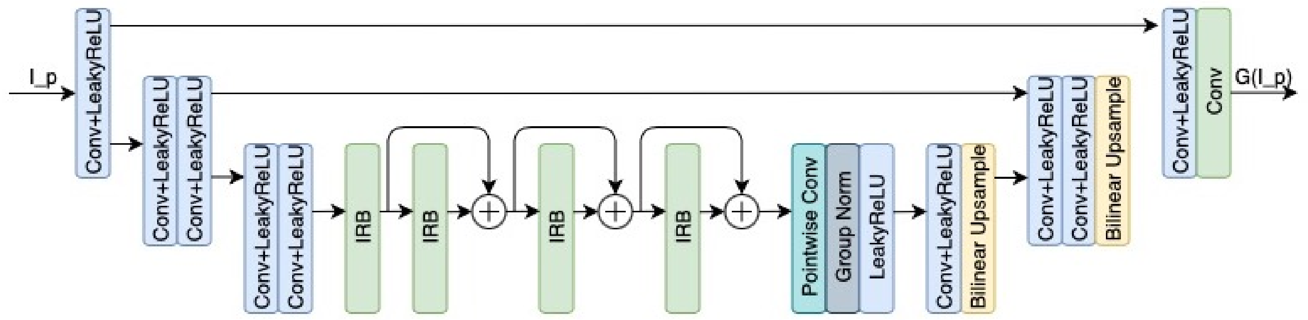

Figure 4. Generator of the White-box IRB Architecture

images collected from various sources in 256x256 resolution. For face cartoon images, there are two options, either the from PA Works or the Kyoto Studios images, which are 5000 RGB images each interpolated to a 256x256 resolution. For our implementation, we have preferred the PA Works style for faces. For the scenery cartoon images, there are three styles presented, which are Shinkai, Hayao and Mamoru styles that take their names from the artists Makoto Shinkai, Miyozaki Hayao and Mamoru Hosoda respectively. Each of them containing 5000 RGB cartoon scenes interpolated to 256x256 resolution. For this project, we preferred the Shinkai style.

4.2. Metrics

To be able to assess the trained model statistically, we adapted Fréchet Inception Distance (FID) from [11]. This distance metric mainly uses the idea from the Inception Score [18], which uses InceptionV3 network [20] for feature extraction in the obtained samples and apply an entropy calculation to find the consistency of the model. Since this score does not include any comparison between real-world samples and the generated samples, we adopted FID instead of the inception score. The formulation of FID metric is given below.

$$\begin{aligned} d_{FID}(D_{real}, D_{gen}) = ||\mu_{real} - \mu_{gen}||_2^2 + \\ Trace(cov_{real} + cov_{gen} - 2(cov_{real}cov_{gen})^{\frac{1}{2}}) \end{aligned} \quad (15)$$

In the formulation given above, D_{real} and D_{gen} represent the data distributions of real images and the generated images for the target domain. The main principle of this score is to measure the similarity of the features extracted from the generated images and the original images. In order to perform the feature extraction, a pretrained InceptionV3 [20] network has been used (for general purpose features of the images). To extract the features, the general practice is followed and the layer named `pool_3` is used. The terms μ and cov denote mean and covariance respectively, whereas the function Trace correspond to the sum of elements in the diagonal of the matrix. At the evaluation step, the better model is expected to have lower FID score. This is because if the generated cartoon images have similar features with cartoons found in the real-world, the FID metric will have a low value. Also, the FID between the generated cartoons and the original photo distribution is observed, since we would like to obtain a balance where they are not too close not too distant. One would be too realistic and as FID gets higher, the inter-domain relations would be lost.

4.3. Results

We have trained each of the network 20 epochs (batch size of 16) on an NVIDIA RTX-2080. Training overall took $\approx$ 36 hours for each model. The inference is faster in the White-box IRB and IRGF variants due to the point-wise and depth-wise convolutions. For training both generator and discriminator, we have used the Adam optimizer [13] with the learning rate 0.0002 and parameters of (β_1, β_2) being $(0.5, 0.99)$.

4.3.1 Quantitative Results

To measure how well model performed statistically, we have measured the FIDs of the two proposed models and the baseline model. The results are as presented in Table 1. It is observable that in both domains of face and scenery, the model has improved on the FID, which means, the IRB generator has reached its purpose in generating more cartoon-like outputs. These outputs conclude the success of the proposed models mathematically.

4.3.2 Qualitative Results

In order to observe the results qualitatively, we present the outputs of given samples taken from the DIV2K dataset [1, 2] in Figure 5. From the figure, we are able to observe that we have achieved the style transform that we have opted for. We see that the colors and shade regions are easily separable and cartoon-like. We also observe that there is a certain improvement in terms of colors color scheme. We see

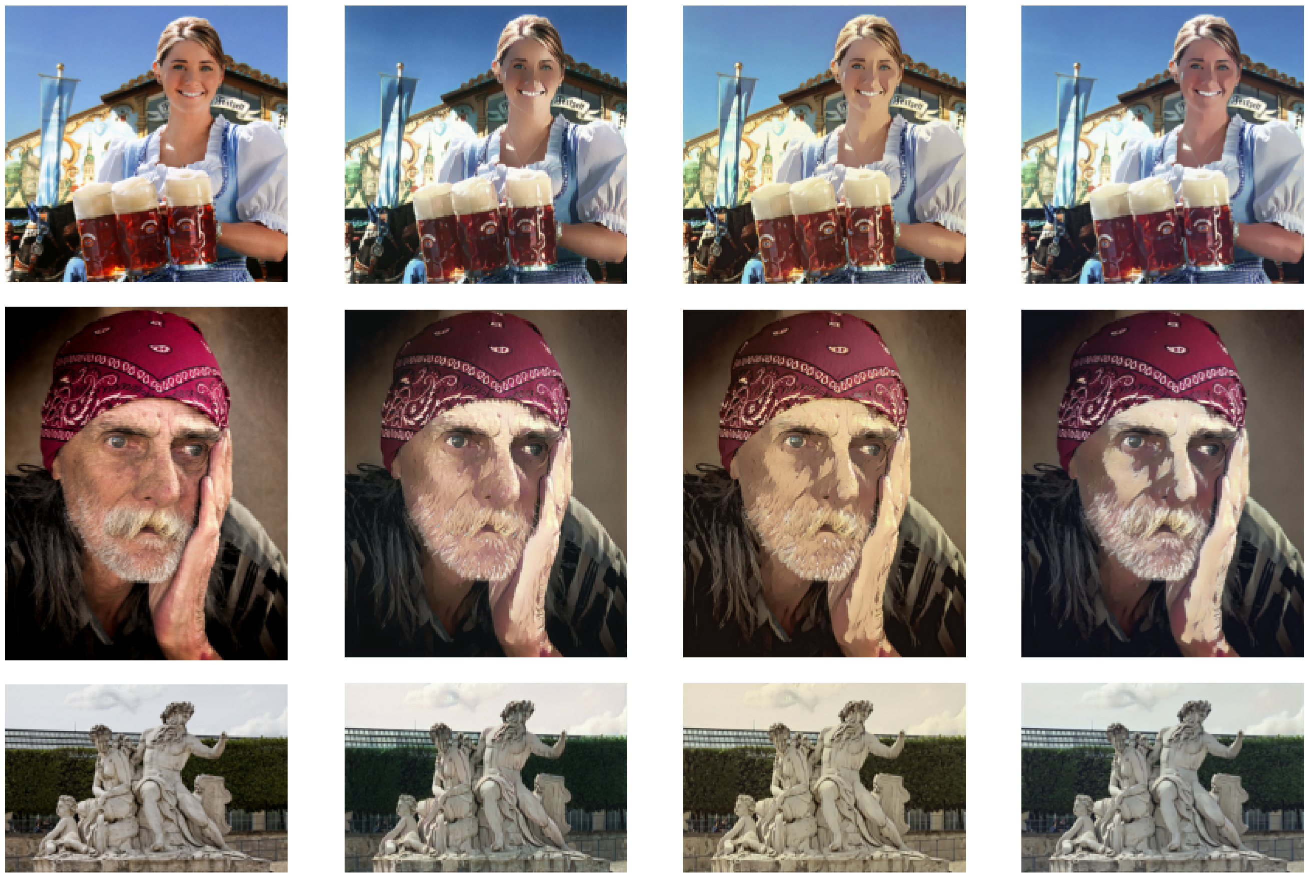
Figure 5. Sample Inputs and Outputs for the Baseline Model (White-box), White-box IRB and White-box IRGF respectively.

FID to	White-box	IRB	IRGF
Face Cartoon	153.592	**135.564**	139.880
Face Photo	49.4	63.961	59.855
Scenery Cartoon	101.149	101.704	**91.639**
Scenery Photo	21.309	24.309	54.692

Table 1. FIDs between the generated distributions to original distributions. Note that these results are valid for the model trained on PA Works face images and Shinkai style scenery.

that while the IRB model also produced successful results, there is a certain color shift that is visible. We don't observe such perturbation in the IRGF model, which mostly preserved the color components of the original image and successfully mixed with style. Although we see some loss of sharpness in the IRGF model, the overall benefit seems higher than the cost.
Another qualitative measure we should observe is the artifact component that we have observed in the original algorithm. For that, we show two sample images where we zoomed on the sections with artifacts in Figure 6. We can clearly observe that the changed operations in guided filter helped eliminate the present artifacts.

5. Conclusion

In this paper, we propose a two-stage improvement on the White-box Cartoonization framework proposed by Wang *et al.* [21], already a state-of-the-art architecture that can generate high-quality cartoonized images. We have determined two issues that could have been improved and worked on them. The first one was to eliminate the distortion artifacts observed. For that, we have removed the guided filter used for smoothing from the training and forced to work only on inference. Furthermore, we have noticed that the images were too realistic subjectively and devised a generator architecture to make the outputs more cartoon-like. We have used IRBs as choice of residual blocks and the group norm to stabilize training. We have seen performance upgrades in both of the algorithms both quantitatively and qualitatively, which validated the performance of the models. The IRB model worked better than the IRGF model in terms of distance to the face cartoon distribution, however, the IRGF was better in terms of scenery.

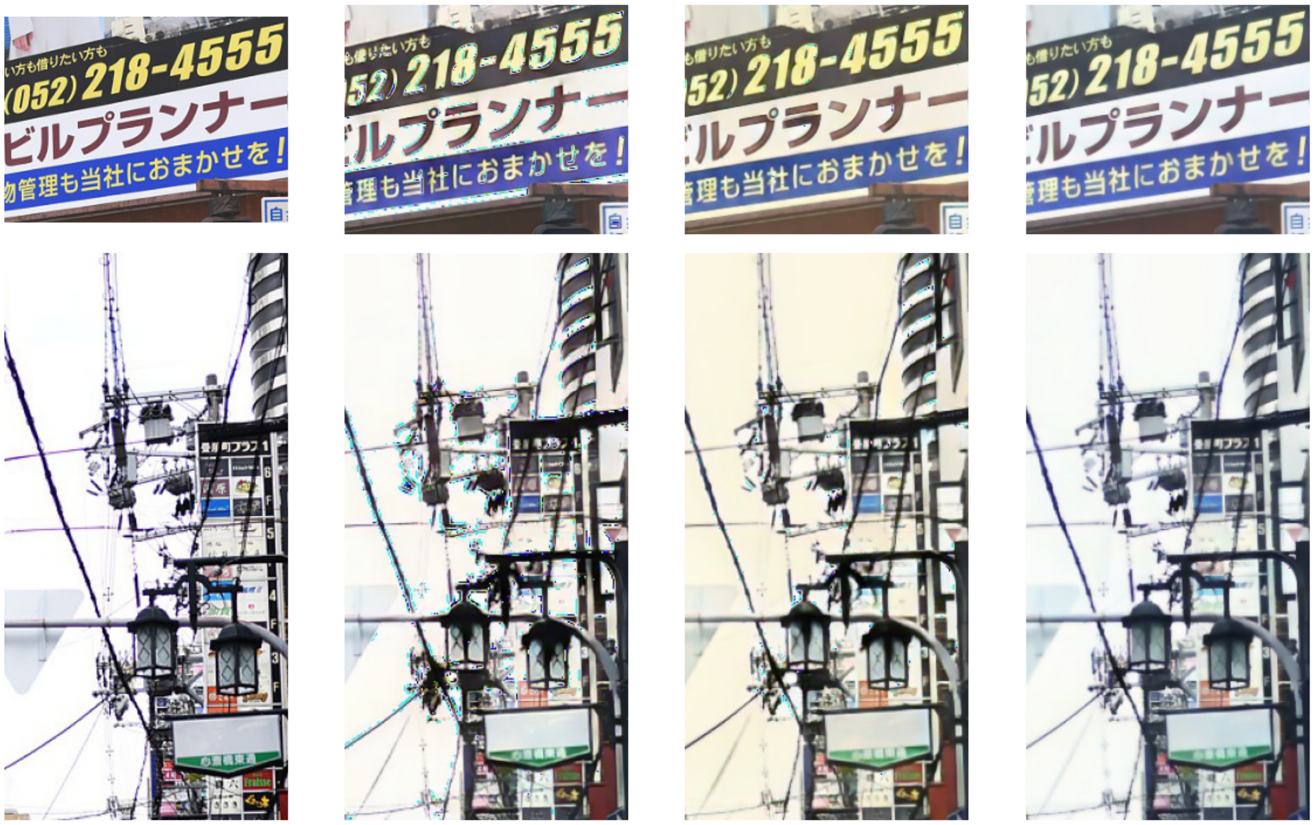

Figure 6. Sample ROIs with Visible Artifacts in the White-box Architecture (2^{nd} column). The images in a row are given as original, White-box, IRB and IRGF respectively

The code, implementation details and more sample outputs can be found in the project GitHub page [1].
Each member significantly contributed in the makings of the project. Celal and Fatih have worked on finding literature content and they have collected a list of possible improvements that could be made. Also they have taken part in the preparation of both the report and the presentation. Ayhan was responsible of converting the ideas to models and implementing those models, training the models and testing the results, creating the GitHub page, preparing the report and the presentation.

References

[1] E. Agustsson and R. Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1122–1131, Los Alamitos, CA, USA, jul 2017. IEEE Computer Society.

[2] E. Agustsson and R. Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[3] J. Behrmann, D. Duvenaud, and J. Jacobsen. Invertible residual networks. *CoRR*, abs/1811.00995, 2018.

[4] J. Chen, G. Liu, and X. Chen. Animegan: A novel lightweight gan for photo animation. In K. Li, W. Li, H. Wang, and Y. Liu, editors, *Artificial Intelligence Algorithms and Applications*, pages 242–256, Singapore, 2020. Springer Singapore.

[5] Y. Chen, Y. Lai, and Y. Liu. Transforming photos to comics using convolutional neural networks. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 2010–2014, 2017.

[6] Y. Chen, Y.-K. Lai, and Y.-J. Liu. Cartoongan: Generative adversarial networks for photo cartoonization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[7] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59:167–181, 09 2004.

[8] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style, 2015.

[9] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks, 2014.

[1]https://github.com/ayhokuyan/CartooNet

[10] K. He, J. Sun, and X. Tang. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(6):1397–1409, 2013.

[11] 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, 2018.

[12] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks, 2018.

[13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

[14] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network, 2017.

[15] C. Li and M. Wand. Combining markov random fields and convolutional neural networks for image synthesis, 2016.

[16] M. Mirza and S. Osindero. Conditional generative adversarial nets, 2014.

[17] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[18] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved Techniques for Training GANs, 2016.

[19] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[20] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the Inception Architecture for Computer Vision, 2015.

[21] X. Wang and J. Yu. Learning to cartoonize using white-box cartoon representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[22] H. Wu, S. Zheng, J. Zhang, and K. Huang. Fast end-to-end trainable guided filter, 2019.

[23] Y. Wu and K. He. Group normalization, 2018.

[24] Z. Xu, M. Wilber, C. Fang, A. Hertzmann, and H. Jin. Learning from multi-domain artistic images for arbitrary style transfer, 2019.

[25] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020.