GAN-based Remastering of Video Recordings of StarCraft Game

Sandeep Kumar Ramani
Department of Electrical and Computer Engineering
University of Wisconsin, Madison
`ramani4@wisc.edu`

Abstract

In this work, I explore the possibility of using Generative Adversarial Networks (GAN) [1] to do an image-to-image translation between the old graphics original game videos and the remastered cartoon game videos in the game of StarCraft. A modified Pix2Pix architecture is used here to remaster the old game videos and it was trained over 12,000 manually collected paired image dataset . I observed that the modified Pix2Pix model suffered from high color loss specific to each player in the game and thus I explore the solution to this problem using a two stage approach. In the first stage, the model generates a grayscale cartoon image given a grayscale original game image and the output is then merged with colorful original game image which is given as an input to the second stage to generate colorful cartoon image. In order to modify the existing Pix2Pix architecture for game specific setup, I experimented by fine-tuning various hyperparameters and architectures whose results show that the modified architecture model is suited for remastering the video game. In the future, I hope to complete the second stage of the pipeline and build an end-to-end remastering GAN model. The source code for this project can be found at here.

1 Introduction

StarCraft is one of the most selling military science fiction media franchise game created by Chris Metzen and James Phinney and owned by Blizzard Entertainment [2]. The game debuted in the year 1998 with old graphics and later in the year 2017 they launched a remastered version containing high-definition graphics and cartoon versions of the game.Given an old StarCraft video game recording is it very challenging to learn a mapping function to remaster into it a cartoon video game as it has different game structures, new objects,and color distribution.

To tackle this problem, Generative Adversarial Networks has shown superior results in the image to image translation tasks. In this study, I aim to use GAN model under a supervised setting to remaster an old graphics video game. This project was motivated to help the gamer to revisit their old video game recording and remaster it into a new cartoon game allowing them to enjoy viewing in the modern cartoon viewing experience.

2 Related work

The Generative Adversarial Network has shown an remarkable progress in recent years to generate high quality samples in various domain applications such as super resolution [3], text-to-image generation [4], image-to-image translation [5], anomaly detection[6] and image inpainting tasks [7]. Some of the popular GAN models are Pix2Pix which is widely used for paired image-to-image translation [5], CycleGAN is used for unpaired image-to-image translation [8], StarGAN is used for multiple domain image-to-image translations problems [9] and BigGAN which based on class-conditional image generation is used to generate high-resolution and high-quality images [10].

The work from Satoshi Iizuka and Edgar Simo-Serra [11] showed that using temporal convolutional neural networks with attention mechanisms acted as a single framework to tackle the entire remastering task and work by VideoGorillas [12] which uses a combination of unique recurrent neural network for each project to learns the characteristics of titles created during the same era of the old video and then a GAN model to generate a new image synthesis and up-scaled image with reduction in noise and artifacts to remaster the old video.

In this work, I modified the existing Pix2Pix architecture to remaster the old graphics StarCraft game video into a cartoon version of the game. The motivation behind using Pix2Pix as a baseliner is that it has shown to work well on various paired image datasets for image-to-image translation tasks.

3 Methods

3.1 Dataset

For this work, since there is no open-sourced dataset for the StarCraft game we manually collected 12,000 paired image dataset containing the old graphics and the cartoon version, an example image is shown in Figure 1. The color images are of size 256 x 256 after preprocessing. The data was collected from more than 50 game video recording from [13] which were equally distributed among the three players Zerg,Terrans,and Protoss. I automated the data collection process using a python script which randomly clicks on the color spots on the game console map, which indicates the user activity, which goes to that required location to a screenshot of both the version. The images are then combined together and passed as an input to the Pix2Pix model.

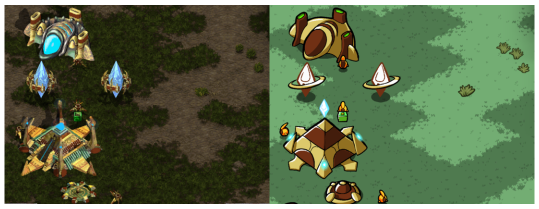


Figure 1: Paired ground-truth image

3.2 Problem Formulation

Given a set of input-output pairs $\{x_i, c_i\}_i^N$, for each image $x_i \in \mathrm{R}^{WXHXC}$, $c_i \in \mathrm{R}^{WXHXC}$ is its corresponding cartoon image pair that encodes the same location. Our goal is to do an image to image translation using the conditional GAN framework, that learns a mapping function G such that G :$\{x, z\}$-> c. The generator network G is trained to fool the discriminator D in order to produce fake image in cartoon domain that cannot be distinguished from the real image by the discriminator which is also trained in parallel to classify the real and fake pair images.

Our objective function can we expressed as a sum of adversarial loss which aims the generator to minimize the loss against the discriminator that aims to maximize it and an L1 loss term which forces to learn low-frequency details to produce realistic looking fake images, thus the overall objective can we formulated as

G = $\arg\max_G \max_D L_{cGAN}(G, D)$ + $\lambda L_{L1}(G)$

where,

$L_{L1}(G) = E_{x,y}[||y - G(x)||_1]$

$L_{cGAN}(G, D) = E_{x,y}[logD(x, y)] + E_x[log(1 - D(x, G(x))]$

where λ is the regularization hyperparameter, D(x,y) represents the discriminator's estimate of the probability that real data instance x is real given y and G(x) is the generated image.

3.3 Network Architecture

3.3.1 Generator Architecture

I modified the Pix2Pix generator architecture to include a higher number of filters in multiples of 256 instead of default 64 and used Resnet 9 block generator network instead of UNet model since it resulted in better performance. Let Ck donates a Convolution-BatchNorm-ReLU layer with k filters and a convolution kernels of size 4x4 with a stride of 2. The convolutions layers are upsampled by a factor of 2 in the decoder and downsampled by a factor of 2 in the encoder and the discriminator. The generator architecture is defined as below,

Encoder:
C256-C512-C1024-C2048-C2048-C2048-C2048-C2048
Decoder:
C2048-C2048-C2048-C2048-C1024-C512-C256

At the last layer of the decoder, the convolution is applied to map to the number of output channels (here its one channel) followed by a Tanh function.As an exception to the above notation, is that BatchNorm is not applied to the first C64 layer in the encoder and leaky ReLUs with slope of 0.2 is used in the encoder while the decoder uses ReLUs instead and this setup is the similar to that of the original Pix2Pix network with exception in the architecture defined above.

3.3.2 Discriminator Architecture

The discriminator architecture is a 70x70 patch GAN which classifies if each 70x70 patch in an image as real or fake and the architectures is defined below,

C64-C128-C256-C512

where batch normalization is not applied to the first C64 layer and a leaky relu with slope 0.2 was used in all the layers. The final layer is convoluted give to a 1-dimensional value which is followed by a binary cross-entropy with logits function to classify if the pair of images are real or fake.

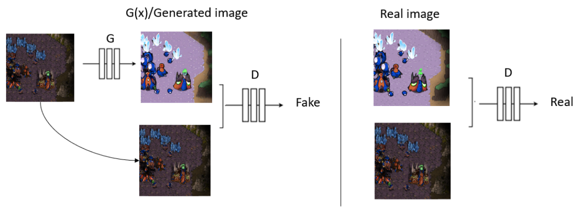


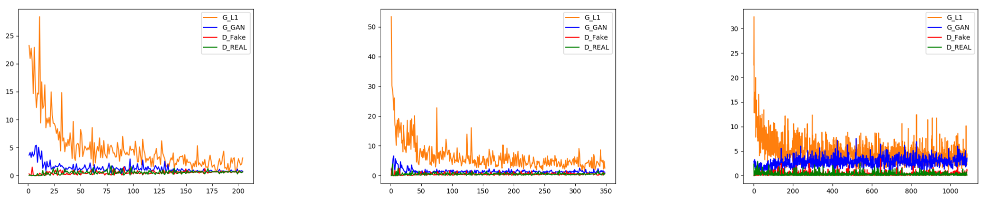
Figure 2: Overview of the Pix2Pix GAN networks which learns mapping from old graphics to cartoon version.

4 Experiments

Using the paired image dataset, I trained the modified Pix2Pix model to generate a realistic cartoon image given an old original game image. I found the image augmentation did not improve the performance as it introduced random flipping in the generated image which did not preserve the game specifics features, thus no augmentation were used during training stage. At first, when we trained over the 2000 image dataset, we observed that the generated image lacked minute details and had a higher L1 loss which is the metric we used to evaluate the model performance. I observed from the loss graph that discriminator was too strong thus I tried to weaken the discriminator by using a single convolutional layer instead of four. Even though it was performing well in the training set and the model was overfitting and did not generalize well on the hold out set.

In order to validate whether the Pix2Pix architectures is able to perform well in this game specific setting, I trained the model by gradually increasing the training set in multiples of 10 starting from 10 to 1000 image dataset in order to see if the model is able to fit the training data well. I fine-tuned various hyper-parameters such as using learning rate, batch size, number of filters in the generator network, cosine learning rate scheduler and tested various architectures to get convergences as shown in Figure 3. I found that the batch size of 1 resulted in lower L1 loss compared to other batch sizes and also for image generation tasks batch size of 1 was shown to be more effective [14]. Increasing the number of generator filters in all the layers helped the model to learn complex features which further reduced the loss as shown in Table 1. I tested both UNet and Resnet variants as the generator network architecture and found that UNet did not converge and resulted in very high loss compared to Resnet.

We observed that the authors of Pix2Pix used dropout as the noise input to the generator to generalize the model well for various image generations tasks. As we are concerned only about StarCraft game we found that by removing dropout we observed a significant reduction in the loss. In addition to that, I observed that by increasing by momentum parameter β_1 to 0.8 there is a reduction in the L1 loss as shown in Table 1 and also it helped to reduce the training loss when trained on larger dataset.



(a) 356x356 Input image (b) Higher no. of generator filter (c) Without drop

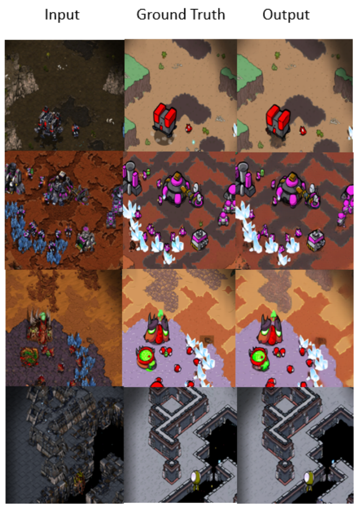
Figure 3: Train loss curves with Resnet 9 block generator trained using cosine learning rate scheduler on 1000 image dataset.

Table 1: Experiments on 1000 image dataset with Resnet 9 block generator trained using cosine learning rate scheduler and various hyper-parameters except the first experiment

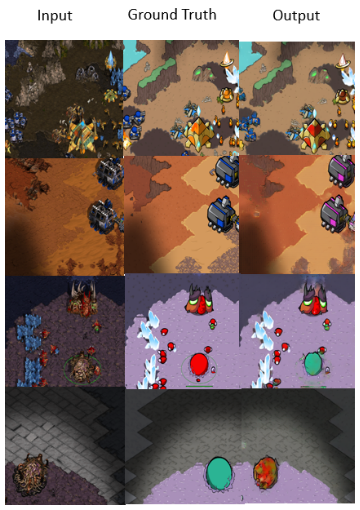
Experiment	Avg. L1 train loss	Avg. L1 test loss
UNet Generator	12.0	18.6
With Dropout	10.69	13.36
Without Dropout	6.107	10.01
Momentum β_1 of 0.8	5.8	9.2
Higher no. of generator filter + β_1 of 0.8	5.3	8.99
356x356 input image + β_1 of 0.8	2.65	8.03

In order to generalize well on unseen data, I increased the training set size to 7500 paired images with the test size of 4500 images, containing diverse color and structure information. I observed that even though the structure in the cartoon images were almost correct, the object colors were mismatched as shown in Figure 4 b). Since, each player in the game as a unique color, preserving color was a challenging problem to solve. In order to solve the color issue, we divided the process into two stage pipeline.

First, the color original game image is preprocessed to a grayscale image of size 256x256 and which is passed as an input to the stage one modified Pix2Pix model. The model generates a grayscale cartoon game image which is merged with a colorful original game image to form a 4 channel input which is passed as an input to the second stage GAN network to generate the colorful cartoon image. The motivation behind this two stage approach is that, since the structure and object information is preserved in the grayscale cartoon image and color information can be extracted from the colorful original game image helping the generative model to learn the color information and thus help solve the color issue. Currently, stage one model uses Resnet 9 block generator with filters in multiple of 256 with a momentum parameter β_1 of 0.8.The model is trained using cosine learning rate scheduler with Adam optimizer. The model gave a average L1 train loss of 6.2 and L1 test loss of 15.5. In the future, I would like to further optimize the model by fine-tuning the hyperparameter and complete the stage two and build an end to end model. Few examples of the generates gray cartoon image and failure cases are shown in Figure 5.

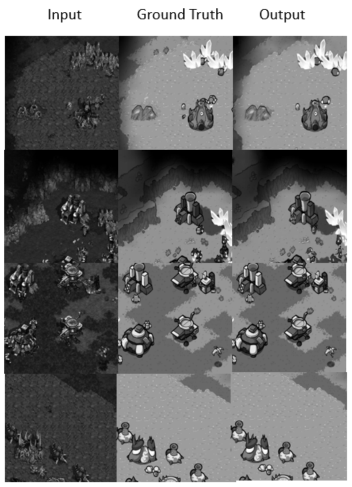


(a) Generated image

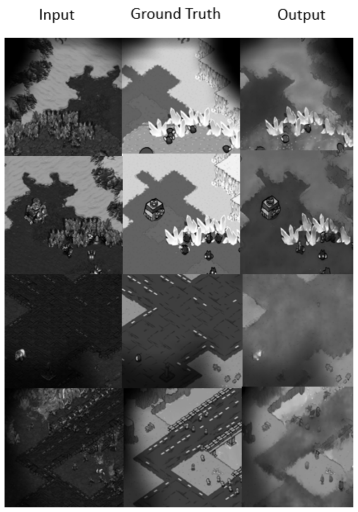


(b) Failure image with color issues

Figure 4: Example results of modified Pix2Pix network on RGB photos



(a) Good test cases



(b) Failure test cases

Figure 5: Stage one generated grayscale cartoon game images on the test data

5 Conclusion and Future Work

In this work, I tackled the problem of generating a cartoon game image given an old graphics original game image. I framed the problem as an image to image translation tasks and explored the solution using an adaptation of the popular Pix2Pix GAN architecture. Using the two stage approach, I aim to generate a realistic looking cartoon image and while maintaining the characteristics of color information specific to each player. The experiment results showed that modified Pix2Pix architecture was able to generate realistic looking cartoon images but had problems in generating correct object colors. Thus, the two stage approach was adopted and the results of stage one show a promising way to tackle the color issue. Future work is needed to complete stage two using novel GAN/deep learning architecture and improve the resolution of generated image and learn more diverse game objects to generalize well on unseen YouTube video.

6 Acknowledgment

I would like to acknowledge Dr. Kangwook Lee for his continuous support during the project and my teammate Sreeranjani Didugu. Special thanks to the authors of Pix2Pix for opening sourcing their code [15]. I also appreciate the help of Google Colab for their GPU server support to train the model.

References

[1] Ian Goodfellow, 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, pages 2672–2680, 2014.

[2] StarCraft game (https://starcraft.com/en-us/)

[3] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A Shi, W. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4681-4690).

[4]Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision, pages 5907–5915, 2017.

[5] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In CVPR, 2017

[6] Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, and Vijay Ramaseshan Chandrasekhar. Efficient gan-based anomaly detection. arXiv preprint arXiv:1802.06222,2018.

[7] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. In Computer Vision and Pattern Recognition (CVPR), 2016.

[8] Zhu, J. Y., Park, T., Isola, P., Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision (pp. 2223-2232).

[9] Choi, Y., Choi, M., Kim, M., Ha, J. W., Kim, S., Choo, J. (2018). 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).

[10] Brock, A., Donahue, J., Simonyan, K. (2018). Large scale gan training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096.

[11] Iizuka, S., Simo-Serra, E. (2019). DeepRemaster: temporal source-reference attention networks for comprehensive video enhancement. ACM Transactions on Graphics (TOG), 38(6), 1-13.

[12] Videogorillas (https://videogorillas.com/)

[13] StarCraft replay video recording (http://bwreplays.com/)

[14] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022, 2016.

[15] Pix2Pix Source code (https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix)