# Anime Super Resolution Using GANs

**Jin Huang**
huang86@stanford.edu

## Abstract

In this paper, I investigate anime super resolution using GANs. I start by reproducing the SRGAN model proposed by Christian Ledig et al., then explore various network architectures, loss functions and training strategies. I get the intuition of the difference between anime and photo-realistic super resolution, which lead me to the proper solutions. I evaluate different models and conclude that the SRWGAN-GP (SRGAN referring to WGAN with gradient penalty) outperforms other solutions with about 10k training examples.

## 1 Introduction

Anime images and videos are at most 1080p, often lower resolution if they are old or from a streaming site. I have seen recently lots of promising super resolution results on images and videos, using deep neural networks, which mostly are trained on real life images and videos. Because anime resides in a lower dimension in terms of color and feature, I see it as a unique opportunity to specialize the networks. Most importantly, I'm an anime fan.

## 2 Previous Work

The state of the art for super resolution problem is SRGAN. Comparing with the regular deep CNN solution, which commonly focus on minimization of the mean squared error between the recovered HR image and the ground truth, SRGAN is more on improving the perceptual quality by produces reconstruction with a lot of high frequency content [1].

## 3 Dataset

I use 11328 anime images randomly picked from Kaggle dataset *tagged-anime-illustrations* [5]. I filter out all the images with size of one side less than 512 pixels before I pick my images. Finally, my dataset for the problem includes 10816 training images, 256 validation images and 256 test images. Each image is composed of RGB channels.

Fig 1. Dataset sample

### 3.1 Data Preprocessing

The input images of the SRGAN network are the pairs of one image in the dataset and a 4x4 bicubic downsampled image. I call the downsampled images as LR, and the images in the dataset as HR in this paper.
To avoid the margins, and also the single-color areas shown in fig 1, I do CenterCrop(384) as the first step.
Due to the GPU memory limitation, only 128 x 128 HR paired with 32 x 32 LR is acceptable. For SRWGAN-GP that computes gradient penalty, I have to use 96 x 96 HR with 24 x 24 LR. So, a RandomCrop is followed.

## 4. Difference Analysis

Pixels of anime and photos are on different distributions. Photos have content on each pixel and content changes with a high frequency among pixels. Anime images usually have areas with single color. Pixels have no or only slight change in these areas. In other areas, the content is also with low frequency. Gradients from adversarial loss contribute much to SR on this trend. I failed many times and tried different hyper parameters to prove that strong adversarial loss and a batch with large single-color areas can suddenly crash the training. For this problem, large input size and batch size are preferred.
However, a weak adversarial loss, as provided in the original SRGAN paper, does not work either, as it can only produce very sparse noises that are not perceptually noticeable, which leads to very similar blurry SR from ResNet.

## 5. Methods

SRGAN consists of two major parts: generator G and discriminator D. The task of G is to produce a SR image indistinguishable from a HR image and "fool" D. The task of D is to distinguish between HR and SR from G, given LR.

### 5.1 Loss function

The loss function of the original SRGAN includes three parts: MSE loss, VGG loss and adversarial loss.
VGG loss is based on the ReLU activation layers of the pre-trained 19 layers VGG network, which is the euclidean distance between the feature representations of SR and HR. [1] For anime, no such pre-trained model as VGG19 is available. Also, the features retrieved from the model trained with real life photos are not working for anime. I reproduce SRGAN model with VGG loss to find it only brings unexpected noises to the anime SR.
I dropped VGG loss and proposed the new generator loss function as:

$$L_{Generator} = L_{MSE} + \lambda * L_{Adversarial} \quad (1)$$

My SRGAN solution uses the standard GAN binary cross-entropy adversarial loss and discriminator loss. While, SRWGAN-GP refers to the L1 adversarial loss and discriminator loss with gradient penalty proposed by Ishaan Gulrajani el at. [2]

$$L = \underset{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}{\mathbb{E}}\left[D(\tilde{\boldsymbol{x}})\right] - \underset{\boldsymbol{x}\sim\mathbb{P}_r}{\mathbb{E}}\left[D(\boldsymbol{x})\right] + \lambda \underset{\hat{\boldsymbol{x}}\sim\mathbb{P}_{\hat{\boldsymbol{x}}}}{\mathbb{E}}\left[(\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1)^2\right]. \tag{2}$$

### 5.2 Network Architecture

#### 5.2.1 Generator

I leverage the generator of SRGAN. The only change I made is to reduce the number of residual blocks from 16 to 8. I trained the generator without discriminator, with 4, 8 and 16 residual blocks. They achieve very similar MSE loss. 8 residual blocks slightly outperform.

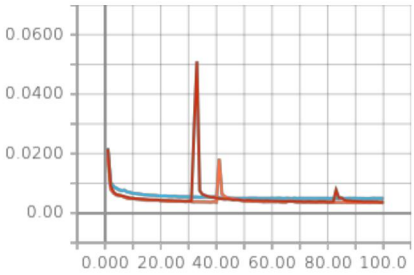


**Fig 2. ResNet MSE Loss**

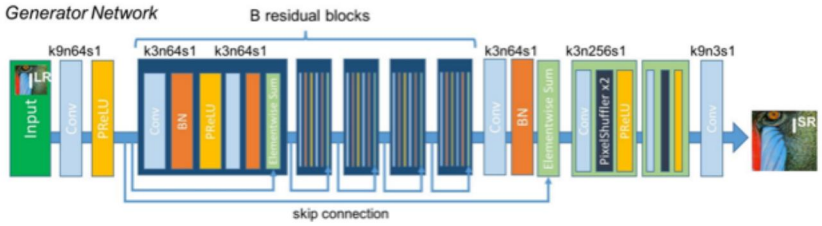


**Fig 3. SRGAN Generator Network**

#### 5.2.2 Discriminator

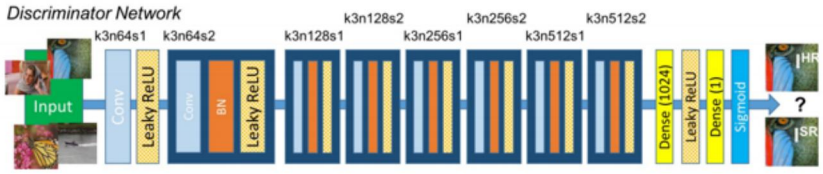


**Fig 4 SRGAN Discriminator Network**

I keep the discriminator the same for the SRGAN solution. For SRWGAN-GP, all the BatchNorm layers and the last Sigmoid layer are removed, because BatchNorm creates correlation between samples in the same batch that impacts the effectiveness of the gradient penalty. [2]

## 6 Training

I train the 2 models both for 500 epochs. For each epoch during training, I alternate between one step of gradient decent on D then G. Default initializations of each type of layers are applied by PyTorch. I use ADAM optimization with default betas (0.9, 0.999) and epsilon 1e-8.

Besides, I print abs mean of the gradients of the weights in the last layers of generator and discriminator to give me a better sense if the model and training are working well. [4]

### 6.1 SRGAN

I train the SRGAN model with learning rate α = 1e-3, which is the default option of PyTorch ADAM optimization. For generator loss shown in formula (1), I use λ = 1e-2 to let the adversarial loss stronger than the original SRGAN paper. I also apply label smoothing by setting REAL as (0.85, 1.10) and FAKE as (0, 0.15) and make labels noisy by flipping REAL and FAKE with a probability of 0.05. [3]

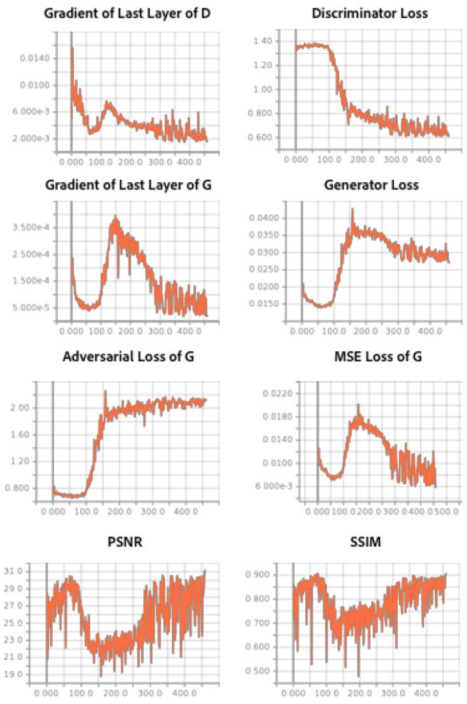


Fig 5. SRGAN Training

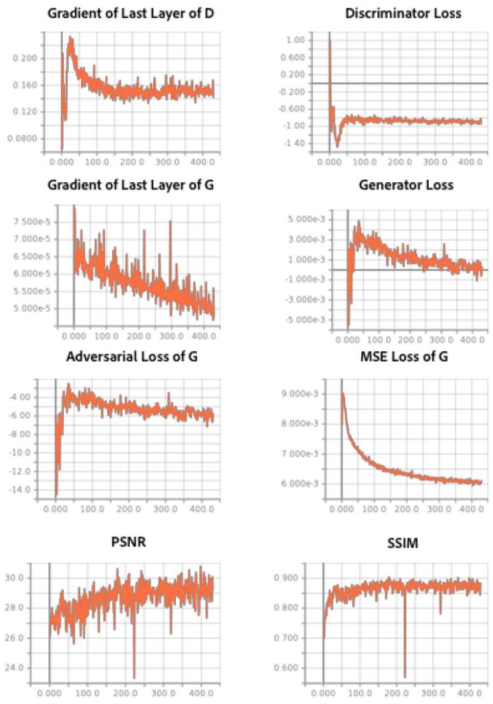


Fig 6. SRWGAN-GP Training

### 6.2 SRWGAN-GP

The training of SRWGAN-GP is slightly different. I use a smaller learning rate α = 1e-4, as it seems to converge very fast according to the training graph. For λ in formula (1), I pick 1e-3 which is smaller, because SWGAN-GP's discriminator loss is between (-infinity, infinity), which makes adversarial loss very strong and contribute a larger gradient when training generator.

## 7. Model Evaluation

I finally set ResNet-8 as the baseline instead of original SRGAN, because 1) VGG19 cannot show me meaningful features for anime 2) With same hyper parameters in the original SRGAN paper, without leveraging with pretrained VGG19 model, I get the same results of ResNet-8.

I have the methods to compute PSNR and SSIM for the results. But ResNet-8 gets highest PSNR (31.86 db) and SSIM (0.917) with blurry results. While SWGAN-GP model gets lowest PSNR (30.02 db) and SSIM (0.889) with the best results.
Basically, I cannot rely on PSNR and SSIM as the metric of the problem, which is the same case as many different generation problems using GANs. Instead, I have to visually check and review the results by myself and a group of my friends and colleagues to get a perceptual score from human eyes. [1]

## 8 Results

The following results compare 1) Bicubic unsampled from LR 2) SR by ResNet 3) SR by SRGAN with modified loss function 4) SR by SRWGAN-GP 5) HR in the dataset. Generally, ResNet gets blurry results. SRWGAN-GP results outperform the others. I attached more results at the end of the paper.

**Fig 7. Bicubic / ResNet / SRGAN / SRWGAN-GP / HR**

## 9 Conclusion and Future Work

Anime super resolution is a different problem to solve, as anime and photos are on different distributions. SRGAN works on Anime Super Resolution with modified loss function, different hyper parameters and removal of the input from pretrained models. SRWGAN-GP solution overall outperforms SRGAN on the problem, with less training time, better results and perception of successful cases.
Later, I plan to explore and apply more GAN models to improve the results of single anime image, and also take advantage of RNN to work on anime videos to get consistent anime frames.

## 10 Contributions

I created the PyTorch implementation of SRGAN and SRWGAN-GP from scratch. Also prepared the code to preprocess data, visualize intermediate results, evaluate different models, predict results and generate the comparation between different solutions. My code is on GitHub at https://github.com/goldhuang/SRGAN-PyTorch.

I tuned hyper parameters and trained the SRGAN and SRWGAN-GP models without any transfer learning from others' pretrained models.

## Acknowledgements

I would like to thank Cristian Bartolome for his helpful feedback during office hours. Also appreciate the suggestions from my colleague Xin Lu. Thanks Yu-shun Cheng to show me the Kaggle dataset.

## Reference

[1] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi
Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network.
[2] Ishaan Gulrajani1, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, Aaron Courville
Improved Training of Wasserstein GANs.
[3] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen
Improved Techniques for Training GANs.
[4] Keep Calm and train a GAN. Pitfalls and Tips on training Generative Adversarial Networks.
https://medium.com/@utk.is.here/keep-calm-and-train-a-gan-pitfalls-and-tips-on-training-generative-adversarial-networks-edd529764aa9
[5] tagged-anime-illustrations https://www.kaggle.com/mylesoneill/tagged-anime-illustrations

## Supplemental Results

Bicubic / ResNet / SRGAN / SRWGAN-GP / HR