Cycle-Consistent Super Resolution Generative Adversarial Network

Jonathan Griffin
Stanford University
jgriffi2@stanford.edu

Abstract

This paper presents Cycle-Consistent Super Resolution Generative Adversarial Network (CycleSRGAN), which aims to perform on the Super Resolution (SR) task using an unpaired dataset. This paper shows the advantages of using cycle consistency for the SR task as well as how using unpaired data effects generated results. CycleSRGAN's generated images are compared with SRGAN's generated images as well as ground truth.

1. Introduction

Up-scaling an image from a lower resolution to a higher resolution is known as the Super Resolution (SR) task. This ill-posed problem has been predominantly studied in a variety of means, as Section 3 shows. The SR problem becomes even more daunting with large up-scaling factors, as mentioned in [7].

This work introduces the Cycle-Consistent Super Resolution Generative Adversarial Network (CycleSRGAN), which aims to tackle the SR problem with large up-scaling factors (up to 4 times the original image input) while using the concept of cycle-consistency. Previous works have not considered this approach, however it seems intuitive that if a network can learn a way to translate to the low-resolution domain, it may learn a better way to translate to the high-resolution domain.

In addition, CycleSRGAN incorporates a perceptual loss function as described in [7], which, when optimized on, generates images that are perceptually similar to the input image.

2. Motivation

Generative Adversarial Networks (GANs) have only recently been used for the super resolution task [7]. This method led to perceptually pleasing photos with higher levels of detail not seen in photos generated using other methods [5]. Even then, Ledig et al. used a generic implementation of the GAN architecture, called SRGAN, only changing the cost function to optimize perceptual-loss as opposed to the mean-square-error [7].

In the same year, [10] proposed CycleGAN, a GAN network with two generators and two discriminators where one generator-discriminator pair tries to outperform the other. This method surpassed the baseline results in many areas, such as photo enhancement.

Combining the two, Ledig et al.'s metric of evaluation and Zhu et al.'s CycleGAN architecture, can lead to an overall better performance on the super resolution task. The intuition behind this is that if a network can learn how to compress an image, it may infer a better way on how to enhance an image.

3. Related Work

3.1. Super Resolution

Recent advancements related to the SR problem have come from the use of various deep learning architectures. Some apply conventional Convolutional Neural Network (CNN) frameworks with unconventional non-linear mapping strategies, such as sparse-coding-based SR methods [3] [9], while others use very deep networks [5], recursive networks [6] [1], or GANs [7].

Methods that use the sparse-prior can produce visually pleasing results. However, this requires prior knowledge about how the input image looks. This limits the range of images in the input domain [3] [9], which is why the other methods have been produced without using this prior.

Methods using very deep networks [5] [6] perform well but take a very long time to train. This problem is exacerbated with the potential of the network loss diverging. The same issue comes with recursive networks [6] [1], which is why these networks are limited to small input images.

[7] uses a GAN network, which produced more visually pleasing results than any previous method while up-scaling to 4 times the original image size. An example image of this method is shown in Figure 1.

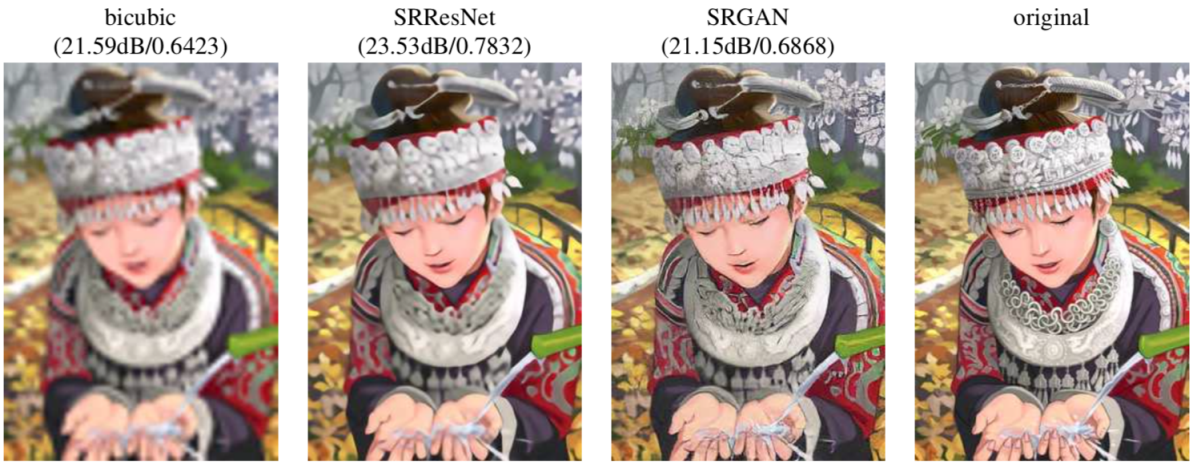

Figure 1: SRGAN results.

Figure 2: CycleGAN results for depth of field.

3.2. Cycle-Consistency

The main work that has been done within cycle-consistency in GANs was created by Zhu et al. [10] introduces the idea that if an image, x, is translated from one domain, X, to another domain, Y, and back to the original domain, the generated image should be the same as the original image: $F(G(x)) \approx x$. This idea can be used in image enhancement, though [10] only uses it for changing depth of field. A diagram of the network is shown in Figure 4 and results of CycleGAN are shown in Figure 2.

4. Methods

4.1. Architecture

CycleSRGAN combines the architecture of CycleGAN and SRGAN. CycleSRGAN has two domains, X and Y, two generators, G and F, and two discriminators, D_X and D_Y. For the purposes of this paper, X is the low-resolution domain and Y is the high-resolution domain. Generator G translates an image from domain X to domain Y and generator F translates an image from domain Y to domain X. Discriminator D_X determines if an image is actually from domain X or is generated by F and discriminator D_Y determines if an image is actually from domain Y or is generated by G. A diagram of this network is shown in Figure 4.

4.1.1 Generators

Generators G and F are almost exactly the same aside from a scaling factor. G and F use the same architecture as the SRGAN model, shown in Figure 3. The generator network begins with a 9x9 convolution operation with 64 filters and stride of 1 followed by a ParametricReLU activation function. The main components of the generator network are 16 residual blocks. Each residual block performs (1) a 3x3 convolution with 64 filters and stride of 1, (2) batch normalization, (3) ParametricReLU, (4) a second 2D convolution function, and (5) a second batch normalization. This is the same procedure as [7] and [4]. After the residual blocks, there is one more convolution/batch normalization layer and the activation's from before the residual blocks and after the residual blocks are summed. Finally, there is a pixel shuffling operation that uses a scaling factor s. The only difference between generator G and generator F's architecture is that the scaling factor for F is the reciprocal of the scaling factor for G. This is to ensure that G up-scales and F down-scales, assuming $s > 1$.

4.1.2 Discriminators

Discriminators D_X and D_Y have the same architecture as SRGAN, shown in Figure 3. The discriminators use the LeakyReLU activation function characterized by $\alpha = 0.2$.

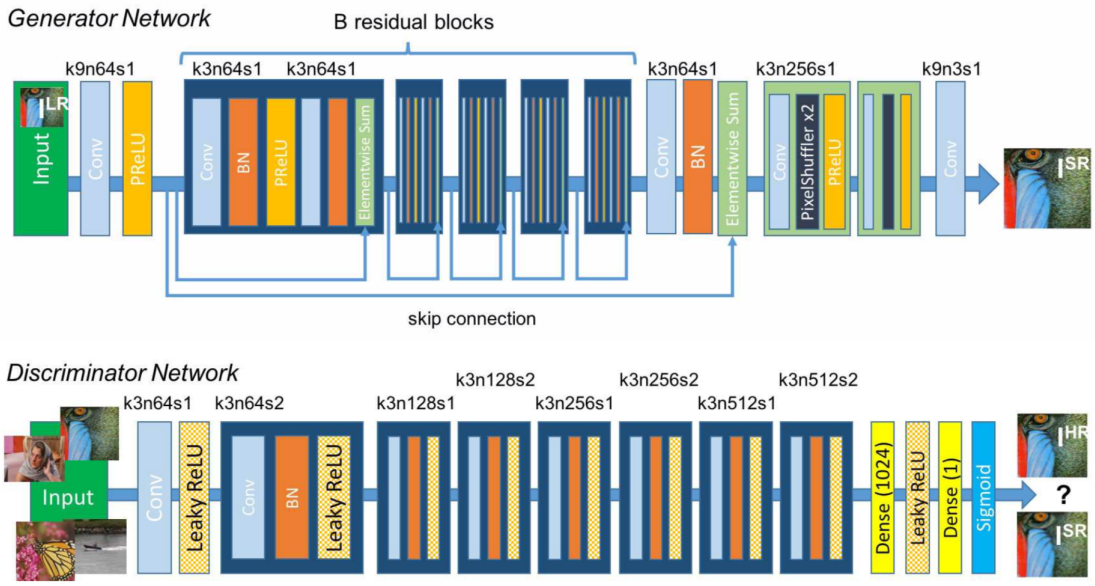

Figure 3: Generator and Discriminator of SRGAN.

All convolutional layers use 3x3 filters and the number of filters increase by a factor of 2 every other convolutional layer. On the second layer of the same number of filters, a stride of 2 is used. Otherwise, the stride is 1. In between each convolution and LeakyReLU, batch normalization is performed. Two fully-connected layers are used at the end, the first followed by a LeakyReLU function and the second follow by a sigmoid function. This design is taken from [7] and [8].

4.2. Loss Functions

CycleSRGAN is optimized on the combination of several loss functions from CycleGAN and SRGAN. Those loss functions are described below.

4.2.1 GAN Loss

The most generic loss function for a vanilla GAN using cycle-consistency is shown in Equation 1. However, this loss function generally does not give the lowest loss after lengthy training [10]. Because of that, Equation 2 is used more often. Equation 2 is the loss function that this paper uses.

$$\mathcal{L}_{GAN}(G, D_Y, X, Y) = \mathbb{E}_{y\sim Y}[log(D_Y(y))] + \mathbb{E}_{x\sim X}[log(1 - D_Y(G(x)))] \quad (1)$$

$$\mathcal{L}_{LSGAN}(G, D_Y, X, Y) = \mathbb{E}_{y\sim Y}[(D_Y(y) - 1)^2] + \mathbb{E}_{x\sim X}[D_Y(G(x))^2] \quad (2)$$

4.2.2 Cycle Loss

CycleGAN incorporates a cycle-consistency loss shown in Equation 3. This loss is used to constrain the model in a way such that if image x from domain X is translated to domain Y and then back to X, then $F(G(x)) \approx x$ [10]. A visual of the cycle-consistency loss is shown in Figure 4.

$$\mathcal{L}_{CYC}(G, F) = \mathbb{E}_{x\sim p_{data}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y\sim p_{data}(y)}[\|G(F(x)) - y\|_1] \quad (3)$$

4.2.3 Perceptual Loss

SRGAN adds a perceptual loss function to their objective [7]. The perceptual loss function they used is shown in Equation 4, where $G_{\theta_G}(I^{LR})$ represents the generated image, I^{HR} is the reference image, H and W are the height and width of an image, and ϕ represents the features from a particular layer of the VGG19 network. Equation 4 only works given a paired dataset with a reference image, I^{HR}. Since CycleSRGAN uses an unpaired dataset, Equation 4 cannot be used. Instead, the perceptual loss function is slightly modified to work with an unpaired dataset with two domains, shown in Equation 5. The intuition behind the perceptual loss is that if two images, x and y, are perceptually similar, then $\phi(x) \approx \phi(y)$.

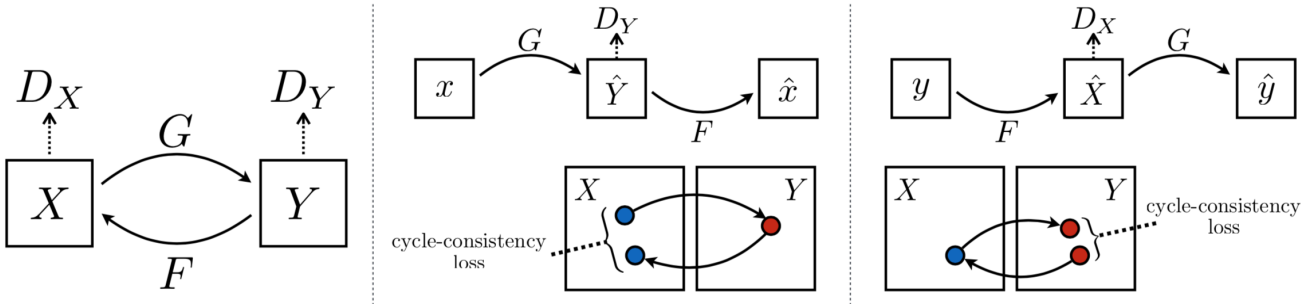

Figure 4: CycleGAN visual, taken from [10].

$$\mathcal{L}^{SR}_{X/i,j} = \frac{1}{H_{i,j}W_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y})^2 \tag{4}$$

$$\mathcal{L}_{SR}(G, F, X, Y) = \mathbb{E}_{x\sim X}[\|\phi(x) - \phi(F(G(x)))\|_2^2] + \mathbb{E}_{y\sim Y}[\|\phi(y) - \phi(G(F(y)))\|_2^2] \tag{5}$$

4.2.4 CycleSRGAN Loss

The proposed CycleSRGAN loss combines Equations 2, 3, and 5 to form Equation 6. Equation 6 is minimized on the two domains, X and Y.

$$\begin{aligned} \mathcal{L}_{CycleSRGAN} &= \mathcal{L}_{LSGAN}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{LSGAN}(F, D_X, X, Y) \\ &+ \lambda_1 \mathcal{L}_{CYC}(G, F) \\ &+ \lambda_2 \mathcal{L}_{SR}(G, F, X, Y) \end{aligned} \tag{6}$$

4.3. Dataset

A subset of **RAISE** is used for the purposes of this paper [2]. The reason the RAISE dataset was used as opposed to **ImageNet** is due to ease of access and available space on local and virtual machines.

4000 unique images were chosen randomly from **RAISE** and were distributed into training and testing sets according to an 80%-20% split (3200 train, 800 test). The training set and testing set were split in half and put into either the low-resolution domain, X, or the high-resolution domain, Y. If the images were put into X, then they were down-sampled to be of size 64x64. If the images were put into Y, then they were first down-sampled to be of size 64x64 and then up-sampled to be of size 256x256 (4 times X domain size). This was done to introduce interpolated values into the images of Y so that the high-resolution discriminator (D_Y) could distinguish them from images of X more easily.

Examples of the images from the dataset are shown in Figure 5. In addition, this dataset is unpaired from domain

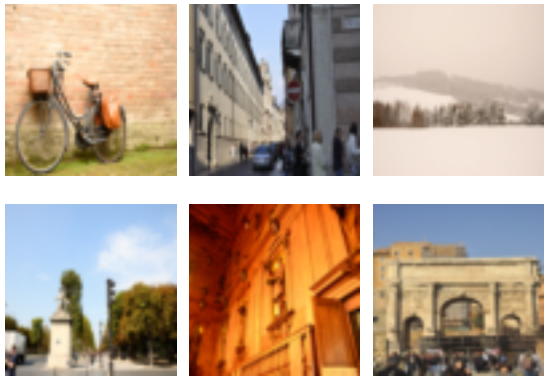

Figure 5: Dataset images: (top row) images from the low-resolution domain X, (bottom row) images from high-resolution domain Y.

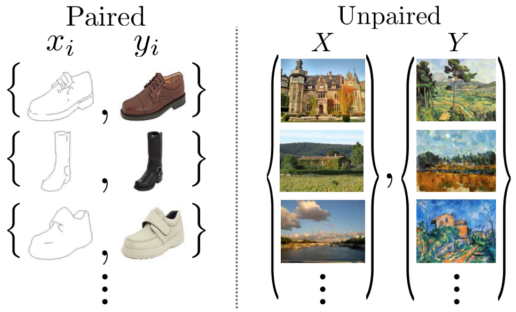

Figure 6: Paired versus unpaired data.

to domain. Unpaired, in this sense, means that there is no ground truth for what the generated image should look like. The reason behind this is so that the model would have no prior knowledge of what kind of image to generate other than that it should look similar to the target domain. Examples of paired versus unpaired data is shown in Figure 6.

Parameters	
Number of Epochs	100
Training Set Size	1600
λ_1	10
λ_2	5

Table 1: Parameter values.

4.4. Implementation

The network was implemented in tensorflow and each model was run for 100 epochs, with a learning rate of 0.0002. $\lambda_1 = 10$ and $\lambda_2 = 5$ gave the best results, shown in Section 5. A summary of the parameters are outlined in Table 1.

5. Results

CycleSRGAN produces results that have higher image quality than the input image, while increasing the resolution. However, the generated images generally have colorization issues and are not as high quality as was hoped. The colorization issues is due to the lack of reference image to compare with during training, so the network learns to care about the resolution of the image but not the colorization of it. The high quality issue comes from lack of training, which is discussed further in Section 6. However, in more high detailed images, CycleSRGAN produces more visually pleasing results, despite being colored differently than the input or ground truth. CycleSRGAN, in this way, acts similar to a blur kernel: removing noise in the image from down-sampling.

Results are shown in Figure 7. A detailed evaluation is discussed in Section 5.1.

5.1. Evaluation

A visual comparison of SRGAN, CycleSRGAN, and the ground truth images is shown in Figure 7. Quantitatively, SRGAN's generated images and CycleSRGAN's generated images are compared with the ground truth images using MSE, SSIM, and PSNR, though [7] discusses how MSE and PSNR are not necessarily the best way to compare perceptually similar images. The quantitative comparison can be seen in Table 2.

Overall, the SRGAN results were visually better, as well as being quantitatively superior. Reasons why this is the case is discussed in Sections 6 and 7.

6. Limitations

CycleSRGAN is limited by the amount of data in domain X and Y. If the domains do not cover a wide variety of scenes, then the the quality of the generated image will be inadequate. This problem is furthered by the fact that CycleSRGAN does not have a ground truth image to compare to during training like SRGAN does [7].

	SRGAN	**CycleSRGAN**
Image 1	0.003/0.788/25.052	0.071/0.753/11.477
Image 2	0.039/0.408/14.058	0.022/0.644/16.666
Image 3	0.004/0.803/23.668	0.142/0.438/8.470
Image 4	0.003/0.855/25.810	0.017/0.769/17.656
Image 5	0.008/0.604/20.794	0.031/0.471/15.027
Image 6	0.002/0.833/26.285	0.072/0.747/11.423
Image 7	0.003/0.828/24.788	0.058/0.712/12.363

Table 2: Quantitative evaluation of generated images with ground truth. Metrics are in (MSE/SSIM/PSNR). Refer to Figure 7 for image numbers.

In terms of the implementation, the project was severely limited by time. GANs take a long time to train and due to the nature of CycleSRGAN's architecture, a pre-trained model was not able to be used to begin training. Because training had to be done from scratch, and multiple models needed to be made to properly tune parameters, each model was not able to be trained until convergence nor was a large training set able to be used.

7. Future Work

The two main paths there are for future work in this area are (1) to increase training time and gather more data, or (2) to use paired data to generate better images.

If (1) is done, the generated images are likely to have better quality and lighting conditions. This is primarily due to the fact that the model was not trained until convergence, so there is still room for improvement if the model was trained longer. In addition, with a larger dataset, the model would become more robust to a larger range of input images. Overall, doing both of these things would lead to better performance.

If (2) is done, the network will have prior information about what the generated result should look like, which follows SRGAN's network and loss function. Instead of using Equation 5 in Equation 6, the new network would use Equation 4. This would mean the generated images, during training, would try to match the features of a ground truth image, leading to generated images that are perceptually similar to some ground truth.

8. Conclusion

The paper presents CycleSRGAN, a Cycle-consistent Super Resolution Generative Adversarial Network. It is a combination of CycleGAN and SRGAN, using the architecture of the generator and discriminator of SRGAN and

Image 1 Image 2 Image 3 Image 4 Image 5 Image 6 Image 7

Figure 7: Visual comparison of methods: (first row) input, (second row) SRGAN generated image, (third row) CycleSRGAN generated image, (fourth row) ground truth. Evaluation metrics are structure in (MSE, SSIM, PSNR) compared with the ground truth.

the overall network structure of CycleGAN while incorporating the loss functions from both. CycleSRGAN uses an unpaired dataset for the SR task which, to my knowledge, has never been done. CycleSRGAN produces results similar to SRGAN but has worse quality. This is larger due to time constraints causing CycleSRGAN to use $\frac{1}{100}$ of the dataset size and train for $\frac{1}{100}$ of the time. With a longer amount of time to train and a larger dataset size, it is likely that CycleSRGAN would produce similar or better results than SRGAN.

References

[1] R. Dahl, M. Norouzi, and J. Shlens. Pixel recursive super resolution. *ICCV*, 2017.

[2] D. Dang-Nguyen, C. Pasquini, V. Conotter, and G. Boato. Raise: a raw images dataset for digital image forensics. *ACM*, 2015.

[3] C. Dong, C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *IEEE*, 2016.

[4] S. Gross and M. Wilber. Training and investigating residual nets. `http://torch.ch/blog/2016/02/04/resnets.html`, 2016.

[5] J. Kim, J. Lee, and K. Lee. Accurate image super-resolution using very deep convolutional networks. *CVPR*, 2016.

[6] J. Kim, J. Lee, and K. Lee. Deeply-recursive convolutional network for image super-resolution. *CVPR*, 2016.

[7] 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. *CVPR*, 2017.

[8] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *ICLR*, 2016.

[9] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang. Deep networks for image super-resolution with sparse prior. *ICCV*, 2015.

[10] J. Zhu, T. Park, P. Isola, and A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CVPR*, 2017.