Image Super-Resolution using Generative Adversarial Network

Siyuan Cheng and Liwei Kang
Shanghai Jiao Tong University

1. INTRODUCTION

Super-resolution imaging is a class of techniques that enhance the resolution of an imaging system. In our project, we mainly focus on Single image super-resolution (SISR), which aims at recovering a high-resolution (HR) image from a single low-resolution (LR) one. Recent years, deep convolution neural network (CNN) approaches have brought prosperous development to SR performance. Many network architecture designs and training strategies have continuously improved the SR performance, especially the Peak Signal-to-Noise Ratio (PSNR) value. However, these PSNR approaches are to minimize the PSNR value of a image, which will often leads to an over-smoothed result. So how to perceptually evaluate the quality of an image becomes important. Several perceptual-driven methods have been proposed to improve the visual quality of SR results. For example, perceptual loss[ESRGAN] is proposed to optimize super-resolution model in a feature space rather than pixel space. Generative adversarial network is also introduced to SR to encourage the network to favor solutions that look more like natural images. One of the milestones in pursuing visually pleasing results is SRGAN, which significantly improves the overall visual quality of reconstruction over PSNR-oriented methods.
However, the HR image generated from SRGAN is still clearly different from the ground-truth image. In this paper, we will introduce a model based on generative adversarial network using a combination of perceptual loss functions to make the generated HR image looks perceptually better.

2. BACKGROUND AND RELATED WORKS

2.1 Super Resolution

Single image super-resolution (SR) is a classical problem in computer vision. Researchers have used many methods to solve this problem. Methods including example-based methods which either exploit internal similarities of the same image, or learn mapping functions from external low- and high-resolution exemplar pairs. One of the representative methods for external example-based super-resolution is sparse-coding-based method. Some researchers also proposed methods based on deep convolutional neural network (SRCNN), which takes the low-resolution image as the input and outputs the high-resolution one. Since generative adversarial network has achieved great success in generating realistic images, researchers have also exploited it to solve super-resolution problems. One representative method using generative adversarial network is SRGAN.

2.2 Perceptual Loss

When consider image transformation problems, where an input image is transformed into an output image, an usual method is to train a feed-forward convolutional neural networks using a per-pixel loss between the output and the ground-truth images. The recent works of researchers have shown that high-quality images can be generated by defining and optimizing perceptual loss functions based on high-level features extracted from pretrained networks.

2.3 Generative Adversarial Network

A generative adversarial network is a framework for estimating generative models via an adversarial process, in which we simutaneously train two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake.

3. MOTIVATION

Super-resolution is a popular problem in computer vision, many researchers have proposed many methods for this problem. However, there is still a gap between the state-of-art generated high resolution (HR) image and the ground-truth image. After reading many papers, we find that many of these methods are using pixelwise loss functions, which might not lead to a perceptually good result. So

we decide to do some research on perceptual loss function and combine it with generative adversarial network to solve super resolution problem.

4. PROBLEM FORMULATION

Our problem can be formulated as training a model that takes a low-resolution image as input and outputs an high-resolution image. The training procedure is to minimize the perceptual difference between generated high-resolution image and ground-truth image.

5. PROPOSED METHOD

SISR is intended to transform the low resolution image into corresponding high resolution one. In our method, we first collect numerous images of high resolution and downsampling them with factor r (demo is 4). Now that we get the training pairs of I^{LR} and I^{HR}, it's the time to implement the training model. Our ultimate goal is to improve the transformed high resolution images' visual quality, that is make them look more perceptually better.

5.1 Model Architecture

We decide to use GAN (Generative Adversarial Network) to train our data. We further define the generator G and the discriminator D together to solve the min-max problem:

$$\min_{\theta_G}\max_{\theta_D}\ \mathbb{E}_{I^{HR}\sim p_{\text{train}}(I^{HR})}[\log D_{\theta_D}(I^{HR})]+ \mathbb{E}_{I^{LR}\sim p_G(I^{LR})}[\log(1-D_{\theta_D}(G_{\theta_G}(I^{LR}))]$$

The general idea behind this formulation is that it allows one to train a generative model G with the goal of fooling a differentiable discriminator D that is trained to distinguish super-resolved images from real images. With this approach our generator can learn to create solutions that are highly similar to real images and thus difficult to classify by D. This encourages perceptually superior solutions residing in the subspace, the manifold, of natural images. This is in contrast to SR solutions obtained by minimizing pixel-wise error measurements, such as the MSE.

5.1.1 *Generator Model*

Figure 1 shows our generator model. The powerful design choice that eases the training of deep CNNs is the residual blocks(RB). We also use upsampling blocks(UB) to create images of high resolution. We choose PReLu as the activation function instead of usual ReLu to optimal the batch normalization. In all the following figures, for each convolutional layer 'k' means kernel size, 'n' means number of feature maps and 's' means stride.

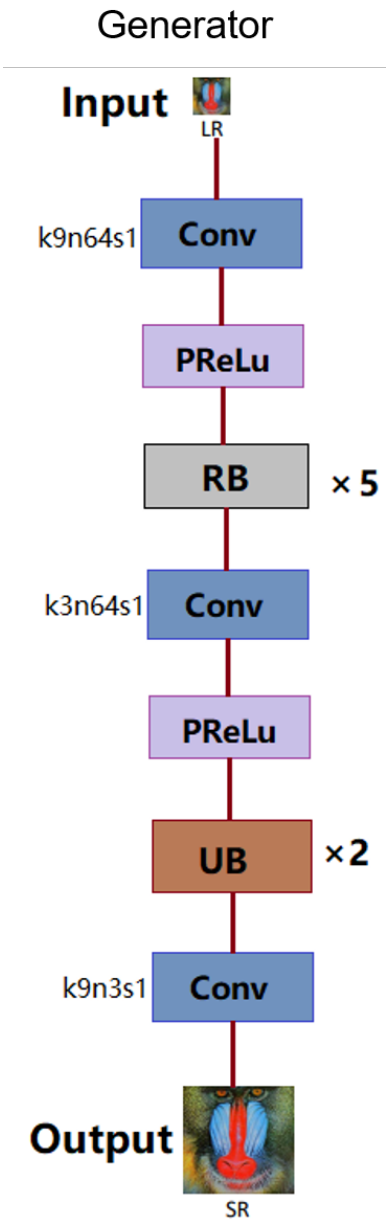

Fig. 1.

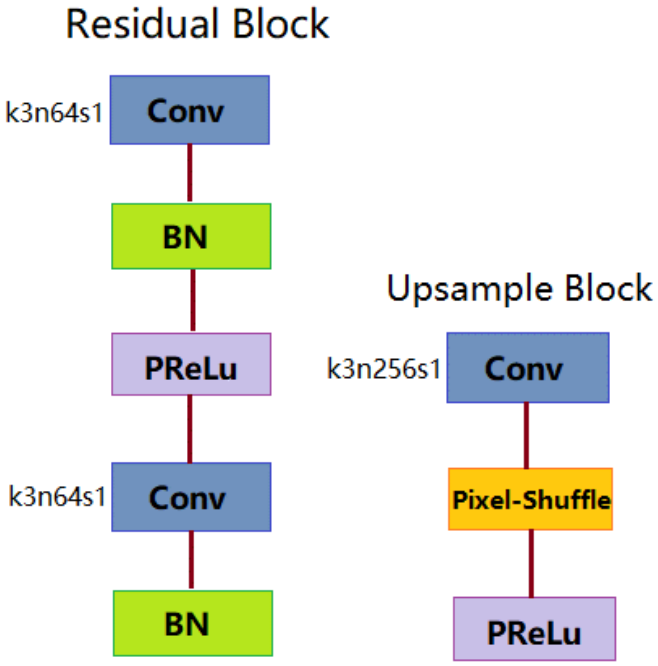

Fig. 2.

5.1.2 *Discriminator Model*

Figure 3 shows our discriminator model. To discriminate real HR images from generated SR samples we train a discriminator network. We use LeakyReLU activation (α = 0.2) and avoid max-pooling throughout the network. The discriminator network is trained to solve the maximization problem. It contains eight convolutional layers with an increasing number of 3 3 filter kernels, increasing by a factor of 2 from 64 to 512 kernels. Strided convolutions are used to reduce the image resolution each time the number of features is doubled. The resulting 512 feature maps are followed by two convolution layers and a final sigmoid activation function to obtain a probability for sample classification.

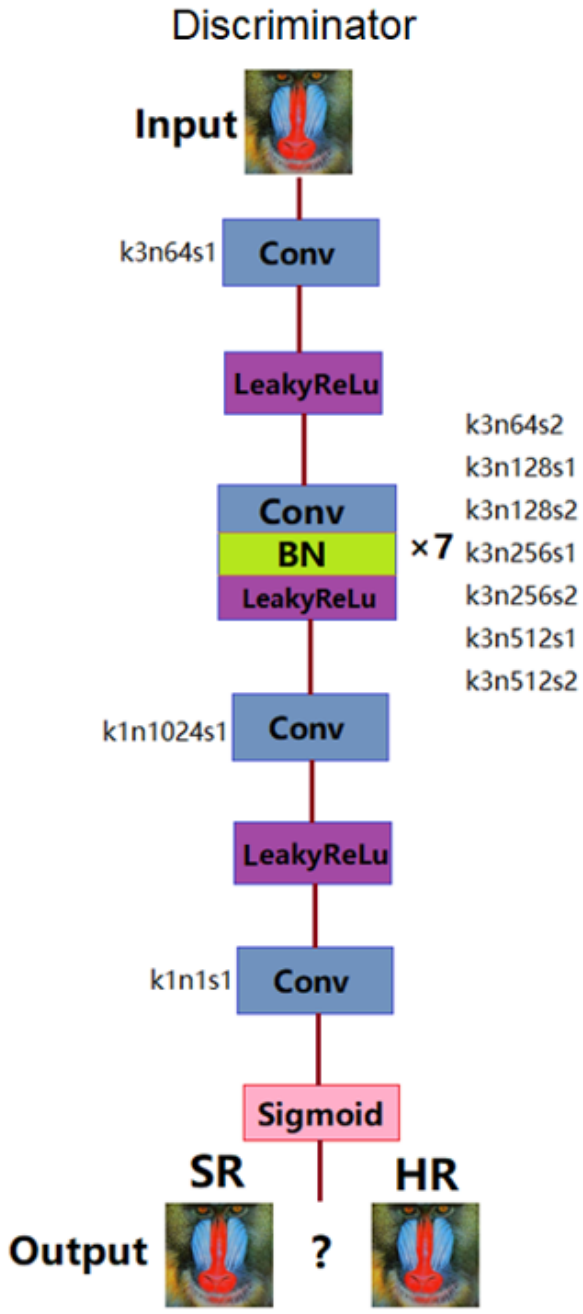

Fig. 3.

5.2 Perceptual Loss Function

The definition of our perceptual loss function SR is critical for the performance of our generator network. While SR is commonly modeled only based on the MSE, we design a loss function that assesses a solution with respect to perceptually relevant characteristics. We formulate the total loss as the weighted sum of the following losses.

5.2.1 *MSE Loss*

MSE, that is the Mean Squared Error between our output and the real HR image.

$$Loss_{MSE} = \frac{1}{WH}\sum_{x=1}^{W}\sum_{y=1}^{H}(I_{x,y}^{HR} - G(I_{x,y}^{LR})) \quad (1)$$

This is the most widely used optimization target for image SR on which many state-of-the-art approaches rely. However, although optimizing the model based on MSE could acquire particularly high PSNR, the solution often lacks high frequency content which results in perceptually unsatisfying solutions with overly smooth textures.

5.2.2 *Adversarial Loss*

We add the generative component of our GAN to the total loss. This encourages our network to favor solutions that reside on the manifold of natural images, by trying to fool the discriminator network. Here are the loss function of the generator and the discriminator:

$$Loss_G = \frac{1}{N}\sum_{n=1}^{N}(1 - D(G(I^{LR}))) \tag{2}$$

$$Loss_D = \frac{1}{N}\sum_{n=1}^{N}(1-D(I^{HR})+D(G(I^{LR}))) \tag{3}$$

The loss function of the discriminator is the final one, and we will further modify the loss function of the generator.

5.2.3 *VGG Loss*

This is the VGG loss based on the ReLU activation layers of the pre-trained 19 layer VGG network. We then define the VGG loss as the Euclidean distance between the feature representations of a reconstructed image $G(I^{LR})$ the reference image I^{HR}.

$$Loss_{VGG} = \frac{1}{WH}\sum_{x=1}^{W}\sum_{y=1}^{H}(\Phi(I^{HR}_{x,y}) - \Phi(G(I^{LR}_{x,y}))) \tag{4}$$

With $\Phi(I)$ we indicate the value in the feature map of the image through the VGG19 network, which we consider given.

5.2.4 *TV Loss*

The total variation (TV) loss. It is based on the principle that signals with excessive and possibly fake details have high TV, that is, the integral of the absolute gradient of the signal is high, so TV loss encourages unwanted noise and spatial smoothness in the generated image and can sometimes improve the results, so we add a little to the total loss.

$$Loss_{TV} = \sum_{i=1}^{W}\sum_{j=1}^{H}\sqrt{|I_{i+1,j} - I_{i,j}|^2 + |I_{i,j+1} - I_{i,j}|^2} \tag{5}$$

5.2.5 *Total Perceptual Loss*

Here is our total loss:

$$Loss_{Perception} = Loss_{MSE} + 10^{-3}Loss_{Adversarial} + 6 \times 10^{-3}Loss_{VGG} + 2 \times 10^{-8}Loss_{TV} \tag{6}$$

For adversarial and VGG loss, since they are somehow large in value compared to MSE loss, so we time 10^{-3} to it and just add a little TV loss to remove the useless detailed noise.

6. EXPERIMENTS

6.1 Dataset

We have collected 16700 high-resolution images as the training data. We train our models in RGB channels and augment the training dataset with random horizontal flips and 90 degree rotations. We evaluate our models on another 425 images to see both the visual and statistic effect.

6.2 Statistic Standard

6.2.1 *PSNR*

Peak signal-to-noise ratio, often abbreviated PSNR.

$$PSNR = 10 \times log_{10}(\frac{MAX_I^2}{MSE}) \tag{7}$$

$$= 20 \times log_{10}(\frac{MAX_I}{\sqrt{MSE}}) \tag{8}$$

$$= 20 \times log_{10}(MAX_I) - 10 \times log_{10}(MSE) \tag{9}$$

Although a higher PSNR generally indicates that the reconstruction is of higher quality, in some cases it may not. Generally, PSNR has been shown to perform poorly compared to other quality metricswhen it comes to estimating the quality of images as perceived by humans.

6.2.2 *SSIM*

Thestructural similarity(SSIM) index is a method for predicting the perceived quality of cinematic pictures. SSIM is a perception-based model that considers image degradation asperceived change in structural information. SSIM is

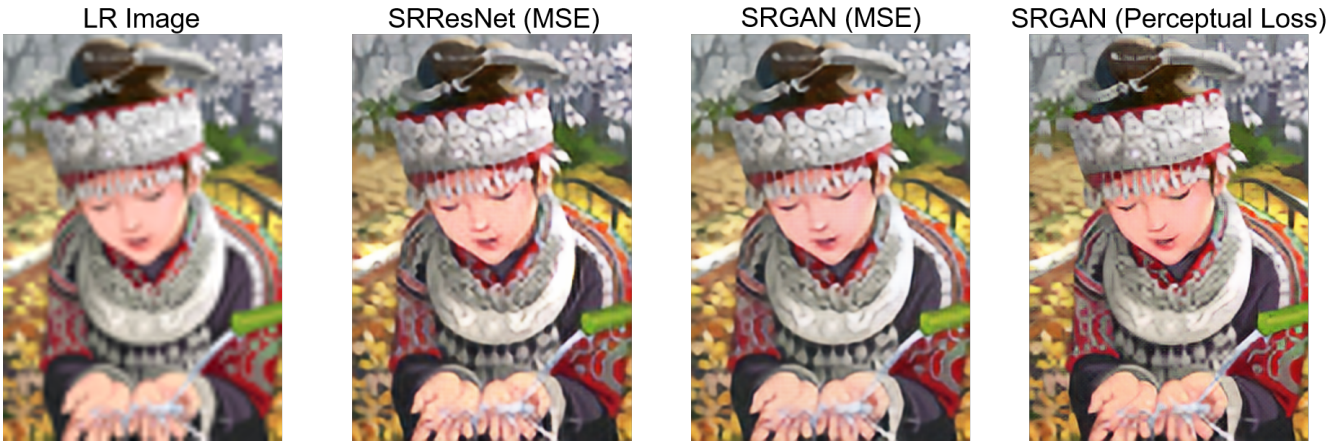

Fig. 4.

designed to improve on the traditional methods like PSNR.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{10}$$

6.3 Training Details

6.3.1 *Parameters*

We have trained all the networks on a NVIDIA GeForce GTX 965M GPU.

Parameter	Number
Batch Size	32
Upscale Factor	4
Epochs	30

We obtained the LR images by down-sampling the HR images using bicubic kernel with down-sampling factor r = 4.
For optimization we use Adam with a learning rate of 10^{-4}. We alternately update the generator and discriminator network and print the progress bar and record the current loss to evaluate the whole process.

6.3.2 *Training Shots*

We record the loss into .csv file to see the training effect and figure 5 shows the shots.

6.3.3 *Final Effect*

We have tried several models to make comparison with our final method, that is bicubic interpolation, SRResNet based on MSE, SRGAN based on MSE and our SRGAN based on more perceptual loss function. Figure 4 shows our effect.

Epoch	Loss_D	Loss_G	PSNR	SSIM
1	0.93967	0.01277	22.5965	0.64579
2	1.00487	0.00855	23.0164	0.65781
3	1.0039	0.00798	23.1941	0.66426
4	0.99808	0.00838	23.2478	0.67171
5	1	0.0082	23.3486	0.67313
6	1	0.00802	23.3171	0.67082
7	1	0.00794	23.3809	0.6801
8	1	0.00797	23.2256	0.68087
9	1	0.00788	23.5045	0.68321
10	1	0.00782	23.3934	0.68602
11	1	0.00777	23.5545	0.68605
12	1	0.00771	23.3053	0.68433
13	1	0.00768	23.5976	0.68552
14	1	0.00764	23.6499	0.6893
15	1	0.00751	23.5636	0.68804

Fig. 5.

And the statistic comparison based on PSNR and SSIM:

Method	PSNR	SSIM
bicubic	21.5938dB	0.6423
SRResNet(MSE)	23.3288dB	0.6829
SRGAN(MSE)	23.7727dB	0.6884
SRGAN(Ours)	23.7617dB	0.6929

We can find that although our PSNR is lower than the one using SRGAN(MSE), ours has the higher SSIM value and looks better in visual quality.

7. CONCLUSION

We have made use of residual blocks when building our model and take numerous images as the input data in the training step. In the experiment, we have compared the several method intended for SISR and find that GAN is better than ResNet in terms of model while our perceptual loss function is better in visual quality than just based on

MSE. We have highlighted some limitations of this PSNR and put out SSIM to together evaluate the final statistic effect. Here is our output and real image of high resolution:

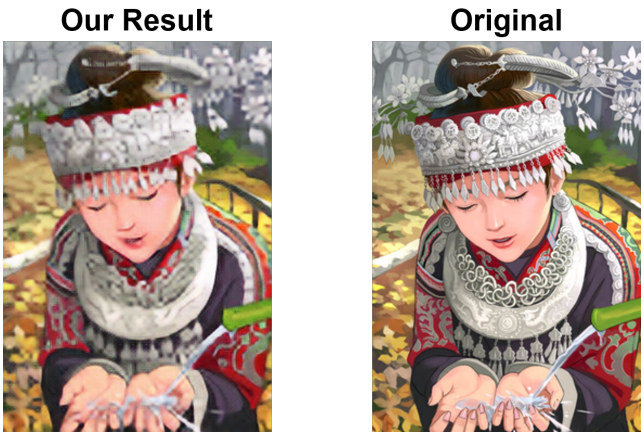

Fig. 6.

The final result shows our successful work, but there is still some part with less detail compared with the real HR image, telling that we have still large space to improve in both the model architecture and the perceptual loss function.