Image Super-Resolution Using Generative Adversarial Networks

David Zheng dzd123@mit.edu
George Du gdu@mit.edu
Jingyu Li jingyuli@mit.edu

Abstract

Recent works have demonstrated the effectiveness of deep convolutional neural networks in single image super-resolution. One major difficulty is recovering high-frequency textures that are present in high-resolution images but are lost when the image is downsampled. In some sense, recovering the fine details of an image is ill-posed; there are many possible natural high-resolution images that can be derived from the same low-resolution image. In this paper, we explore different objective functions that can help the neural network learn high-definition super-resolution images. We combine mean squared reconstruction error (MSE) with various adversarial losses, including a discriminator network that examines the Fourier transform of the super resolved image instead of the pixel values. We also explore training our network for the explicit purpose of being recursively applied to a starting image.

1. Introduction

Convolutional neural networks (CNNs) have become increasingly popular for image processing tasks after [3] showed their ability to solve image classification tasks. Recently, CNNs have been applied to image super-resolution [1, 6], the problem of scaling up a low resolution image while retaining the content of the original image. The difficulty of superresolution stems from the desire to create images that also have the details expected of high-res images. These details are not present in the original image, and so they must be hallucinated by the super-resolution method. As described in [5], the ability to upsample images well has many practical applications, such as medical image processing, text image analysis, and facial image analysis.

2. Related Work

The most basic approaches to super-resolution do not require neural networks. These approaches include linear, bicubic (a baseline for our experiments), and Lanczos filtering [2]. While these approaches are much simpler and faster, images turn out to be blurry and details, for example in texture, are lost. Our approach is heavily based off the paper [4] which uses generative adversarial networks.

2.1. Design of SRGAN

The paper proposes a two part neural net architecture. The first, known as the generator, accepts a low resolution image and outputs a high resolution image. The second, known as the discriminator, accepts a single image and outputs a single value in $(0, 1)$, which is interpreted as the probability the image was not generated by the generator. The sole role of the discriminator is to train the generator, though for the discriminator to be effective, it needs to be (simultaneously) trained on real and synthetic images. Previous work has shown that these generative adversarial networks (GANs) can allow the generator to output photorealistic images.

3. Method

In image super-resolution, we wish to produce a high-resolution super-resolved image from a low resolution input image. That is, if the low resolution image is $W \times H \times C$ in size, the model attempts to construct a high-resolution equivalent of size $rW \times rH \times C$ for some scaling factor r. In this paper, we focus on super-resolution by a factor of $r = 2, 4$. During training, we downsample our training images by a factor of r to use as input into the network and compare the model output with the original image. For each mini-batch we crop 16 random 96×96 sub images of distinct training images. We can still apply the model on arbitrary image size as it is fully convolutional.

The core of our model is a generator network that produces high-resolution images. To this end, we follow Ledig et al. and use a deep residual network (ResNet), composed primarily of B residual blocks [4]. Figure 1 outlines the exact generator architecture we use in more detail. Note the use of batch normalization during training and residual connections. Each of the residual blocks maintains the dimensions of its input. In order to add upsampling to our generator network, we use deconvolution blocks at the end of our network. Each deconvolution block upsamples by a factor of 2, and we add as many blocks as implied by the scaling factor r.

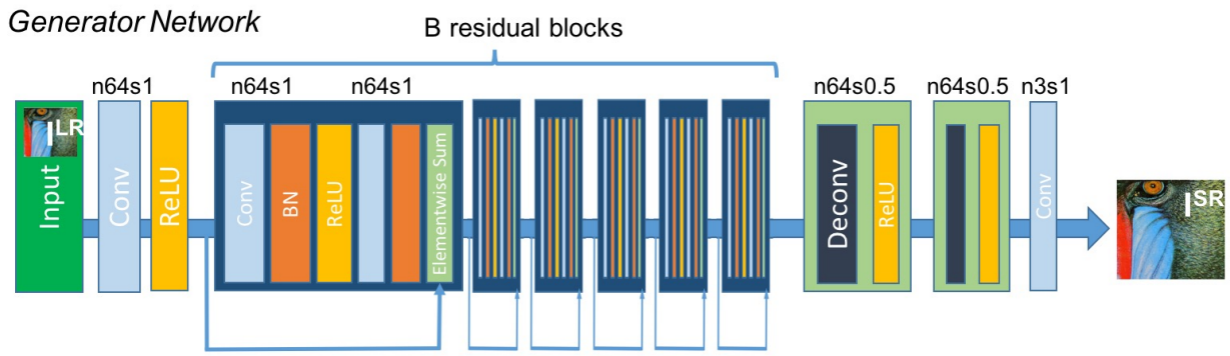

Figure 1. Architecture of generator network with number of depth of each layer(n) and stride(s) [4]. Uses 16 B residual blocks as well as addition deconvolution layers to upsample image.

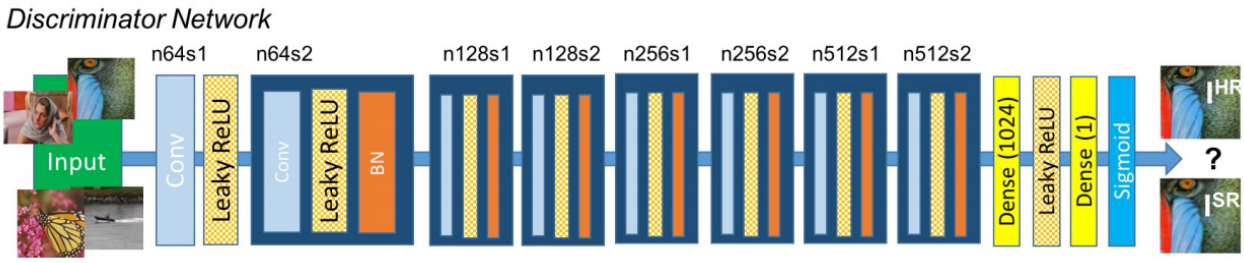

Figure 2. Full architecture of discriminator network [4].

Crucial to the training of the generator network is the choice of the loss function. Ledig et al. makes the distinction between content loss, which rewards outputs close to their corresponding groundtruth images, and adversarial loss, which rewards outputs that are hard to distinguish from the set of high resolution images [4]. For content loss, we use the pixel-wise mean squared error (MSE) of the network's output compared to groundtruth. Minimizing MSE loss has the advantage of simultaneously maximizing peak signal-to-noise ratio in the resulting image. However, as later demonstrated, training on mean squared error along tends to yield overly smooth solutions.

The MSE solution tends to learn a pixel-wise average of possible natural images rather than some particular high-resolution image. Adversarial loss addresses this issue. Following the works of Goodfellow et al., we define a discriminator network, which we optimize to distinguish between natural images and images outputted by the discriminator network. Figure 2 describes our architecture for the discriminator network.

We use an Adam optimizer with learning rate 10^{-4} and $\beta_1 = 0.9$ for training. First, we pretrain the generator network with MSE loss before involving the discriminator. During training, we alternate between two steps. In the first step, we input a low-resolution image into the generator and input its super-resolution output into the discriminator network. In the second step, we input a groundtruth image into the discriminator network. We use a sigmoid layer with cross-entropy loss to train the discriminator to output 0 in the first step and 1 in the second step. We apply MSE loss to the generator, training the network to match the original high-resolution image. We also apply adversarial loss, training the generator to have the discriminator output 1. The relative importance of content and adversarial loss applied to the generator network is of great importance and is discussed in our "Experiments" section. We also present several variations on our original model, including the application of the Fourier transform on the discriminator input.

4. Experiments

In this section, we discuss variations on the model architecture and hyperparameters as well as their effects on different performance metrics. For these experiments, we use the high-resolution mini-places dataset. We use 168,000 train images, 48,000 validation images, and 24,000 test images.

4.1. Pre-training

Initially, the generator outputs uniformly black images, so pre-training is used to get the generator up to an acceptable baseline. Basic upscaling is not difficult, as a simple

solution is to copy each input pixel into each of the r^2 corresponding locations in the upscaled image and apply a blur filter.

The generator network is pre-trained with MSE loss to learn $4\times$ upsampling. As in previous works, we refer to the MSE-based SR network as SRResNet. We train SRResNet for approximately 50 epochs. In Table 1, we compare the MSE loss of SRResNet against that of bicubic interpolation. We also include the average Structural Similarity Index (SSIM), a metric meant to approximate the perceived similarity between two images. Figure 4 shows the result of bicubic interpolation and SRResNet on several test images. Note that the test images appear less pixelated and do not have as many border effects as the bicubic images.

4.2. Adversarial Training

Now, we add in adversarial loss by simultaneously training the generator and discriminator. We refer to the generator trained with MSE and adversarial loss as SRGAN. As with pre-training, we train the model to perform 2x and 4x upsampling.

The adversarial training is expected to increase the MSE, because we no longer directly optimize MSE and sacrifice some reconstruction fidelity to push the generator output toward more natural-looking images. To this end, we tune the "adversarial weight", a hyperparameter that we multiply with adversarial loss before summing with MSE loss to get the generator loss. We find that an adversarial weight of 10 achieves good performance. In Table 1, we summarize our results with SRGAN. When the adversarial weight is low, such that the adversarial loss is very small ($< 4\%$ of MSE), the adversarial training makes very little difference and does not affect the MSE. When the adversarial weight is high ($> 50\%$) of MSE, generator weights are unstable, and large sacrifices in MSE are made in order to compete with the discriminator. Between the two extremes, various artifacts are introduced in exchange for outputting some sharper lines. In Figure 4, we show the results of reconstruction with SRGAN. Unfortunately, rather than learning high resolution details, SRGAN adds "streaky" artifacts to the super resolution image.

4.3. FFT Adversarial Training

The discrete Fourier transform of natural images tends to follow the same general distribution, as the natural world is continuous and dominated by low spatial frequencies. Statistical approaches for computer vision have successfully used the 2D DFT to characterize natural images. Thus, we attempt to simplify the discriminator architecture by passing the input image through a non-trainable FFT layer, extracting the magnitudes, and passing the result through two dense layers. The goal is to retain the performance of the original, more complex deep convolutional network, while increasing the training speed of the discriminator component. We call this network fftGAN.

The resulting discriminator appeared to be weaker than the convolutional discriminator. The network had more stability with higher adversarial loss parameters, staying at a consistently low MSE, even with an adversarial loss multiplier of 100. However, the effect was that the weights did not change much relative to the weights for SRResNet—the image only had a small increase in sharpness.

4.4. Recursive Training

We experimented with training a model that is able to recursively up-sample. To do so, we down-sample the original images by a factor of 2 and by a factor of 4. We first train our network to up-sample the 1X images to match the 2X images (Figure 3). After training, we obtain the SR2X (Figure 3) images by running our neural network on the 1X images to up-sample by a factor of 2. We then train the GAN to up-sample the SR2X images to match the original (4X) images. By doing so, we are training the GAN to up-sample on top of its up-sampled images. We call this new network rGAN. Our goal for this is to train a model that is able to up-sample by higher factors and still produce reasonable images.

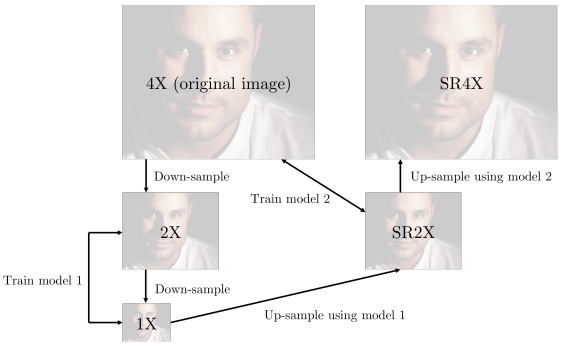

Figure 3. Recursive training pipeline.

As you can tell from the sample images outputted by the neural net, the excessive "sharpness" that was seen in our other adversarial training runs was magnified, resulting in angular, sharply contrasting images. It appears that repeated upsampling causes error to accumulate superlinearly, and rGAN should be applied only for artistic purposes.

2X	bicubic	SRResNet	SRGAN	rGAN(4x)
MSE	384.1	88.3	107.7	2178
SSIM	0.632	0.873	0.822	0.219

4X	bicubic	SRResNet	SRGAN	fftGAN
MSE	619.4	174.0	280.2	183.4
SSIM	0.533	0.738	0.590	0.724

Table 1. Comparison of methods: bicubic, SRResNet, fftGAN, and rGAN (recursive GAN) for 2X and 4X up-sampling.

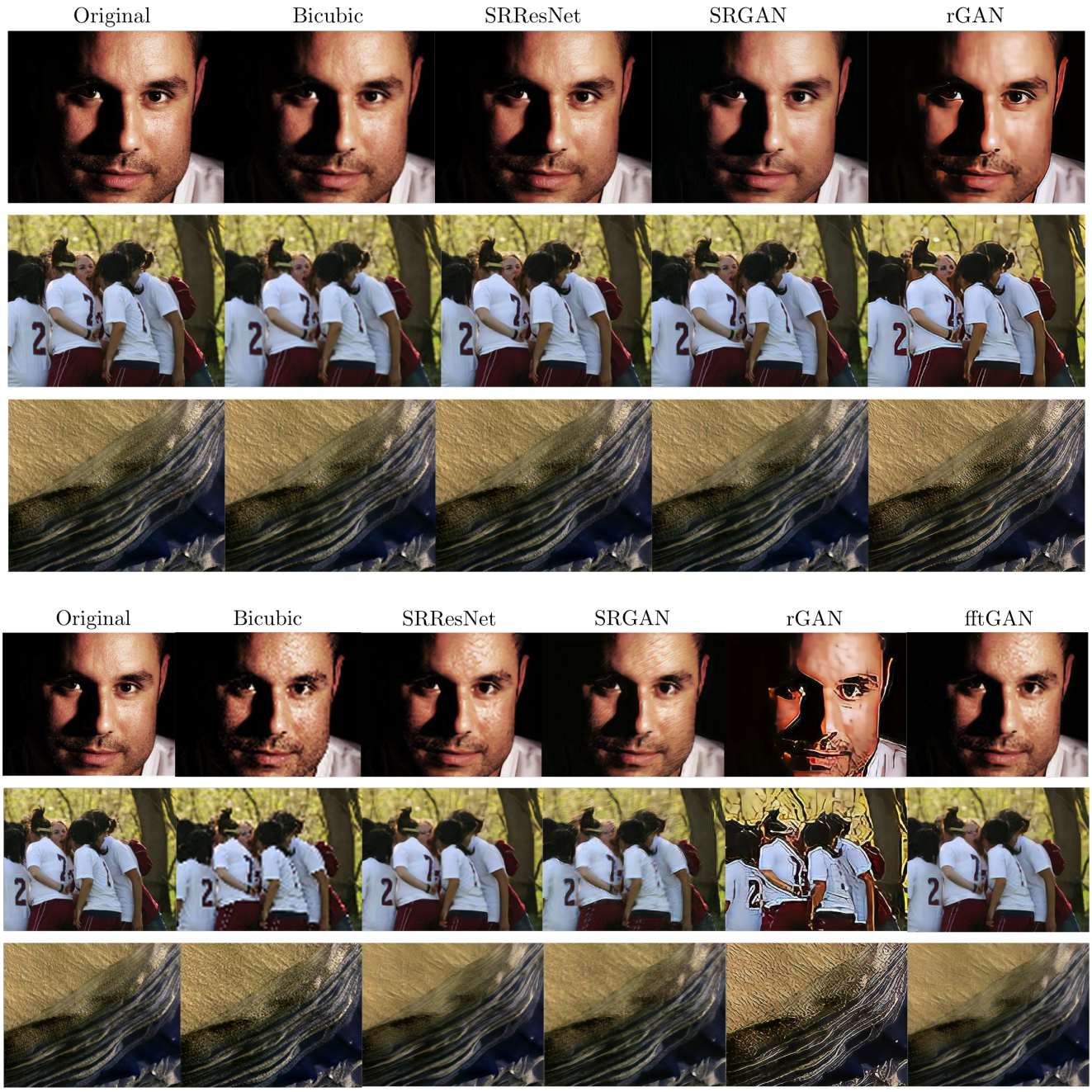

Figure 4. (Top) A comparison of different methods for upsampling images by 2. Recursive GAN achieves an impressive degree of sharpness. (Bottom) Upsampled images using a factor of 4. Here, rGAN is being recursively applied to upsample by 4. Notice that the CNN approaches (with the exception of rGAN) achieve significantly better performance than bicubic interpolation. To view the original images, visit: https://drive.google.com/drive/folders/0BxqA9nYUpfwUTFRwb0pzV1JBeXM

5. Conclusion and Future Work

One way to improve the network is to change the MSE loss function to something more principled. Training using MSE encourages smoothness, which is something that is undesirable for realistic images. One idea is to use another, pretrained neural network to model the visual penalty incurred by pixel differences in a given region. Given more time, we could also improve the neural network performance by investigating the effect of varying the generator and discriminator architectures, e.g. increasing/decreasing

the number of res blocks.

Overall, we see that using neural networks handles super-resolution better than simple mathematical approaches, such as bicubic interpolation. The ability to learn a large number of filters allows our network to better model the textures of actual images. Generative adversarial networks are very tricky to train, and require special care to ensure that the generator and discriminator don't end up unbalanced, but when the training is successful, we see results that deviate in interesting ways from vanilla error minimization.

6. Individual Contribution

I helped develop the initial neural network (various layer blocks, loss functions, etc) with David. After Jingyu found the data, I built the data pipeline using tensorflow input producers and queueing, and wrote the checkpointing system. Later, I implemented fftGAN, and took a significant role in parameter tuning and optimization. Our jobs were largely run on my personal computer, which has a NVIDIA gtx1070. In the evaluation stage, I helped write and debug code that computed SSIM and the MSE loss on our test set.

References

[1] C. Dong, C. C. Loy, K. He, and X. Tang. Image super-resolution using deep convolutional networks. *CoRR*, abs/1501.00092, 2015. 1

[2] C. E. Duchon. Lanczos filtering in one and two dimensions. *Journal of Applied Meteorology*, 18(8):1016–1022, 1979. 1

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012. 1

[4] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. *CoRR*, abs/1609.04802, 2016. 1, 2

[5] K. Nasrollahi and T. B. Moeslund. Super-resolution: a comprehensive survey. *Machine Vision and Applications*, 25(6):1423–1468, 2014. 1

[6] Y. Wang, L. Wang, H. Wang, and P. Li. End-to-end image super-resolution via deep and shallow convolutional networks. *CoRR*, abs/1607.07680, 2016. 1