CS230

Human Portrait Super Resolution Using GANs

Yujie Shu
yujieshu@stanford.edu

Figure 1: Input LR 32x32, SRPGGAN 8x Output 256x256, and Original HR 256x256

Abstract

Human portrait photo super resolution (SR) is a sub-category of image-to-image translation problem. We implement our own version of SRResnet and SRGAN from scratch to train human face datasets for 4x upscaling factors. For SRGAN, we experiment with NSGAN, WGAN-GP, and PGGAN respectively to improve image quality. We also explore different loss functions to obtain more realistic human portrait images with finer face and hair texture details. Transfer learning are also used to accelerate and stabilize the training process. Empirically, we find GAN networks are much harder to train but generate images with better quality. Our SRWGAN-GP and SRPGGAN models produce best super resolution results among our 4x upscaling factor tests. Then we extend SRPGGAN which grows the model progressively to do a 8x upscaling and get Figure 1.

1 Introduction

Building a robust neural network to automatically reconstruct low resolution images to high resolution images is quite challenging but useful in various surveillance application, medical imaging, satellite image analysis or old photo recovery. Compared to traditional methods, deep learning has a breakthrough in super resolution in accuracy and speed. For this project, we narrow down the problem to single image super resolution (SISR) on human portrait. Since human precision has a higher standard for faces and can really tell the nuances, we explore different neural network architectures and train our model to match this level of expectation.

2 Related Work

CNN solutions in SISR is proved to be very effective since SRCNN [8], which is using deep convolutional neural network to solve SR problem and demonstrate fast speed and good quality

compared to traditional sparse-coding-base SR methods. As CNN evolves, Residual Network architecture [12] is created to go much deeper with skip connections, but without suffering vanishing gradients problem of plain networks. SRResnet in SRGAN paper[2] then uses deep residual network and shows improved results. We build a SRResnet as our baseline model.

GAN, brought up by Ian Goodfellow [13] in 2014, is a newer architecture in deep learning research area but generates superior results in various topics. SRGAN [2], using SRResnet as its generator capable of inferring photo-realistic images for 4x upscaling factors, established the state-of-the-art in SR. Therefore, we build our own version of SRGAN to generate HR human portrait images and compare the results with SRResnet outputs. In first model of SRGAN, NSGAN loss function is applied. WGAN [8] is claimed to make GAN training more stable, by enforcing the discriminator stay within the space of 1-Lipschitz functions. WGAN-GP [8] adds gradient penalty as an alternative to weight clipping in original WGAN to further stabilize the training. We integrate WGAN-GP in our SRWGAN-GP model. Last but not least, PGGAN [9] can magically turn the noise input into 1024^2 HD face images and introduce the idea of progressively grow the resolution of generator and discriminator by adding new layers along the training process. We integrate this idea and build a SRPGGAN model to compare with earlier models.

MSE and PSNR are common metrics to evaluate SR algorithms before GAN [3, 10]. However, other papers claims that they only evaluate pixel-wise differences but failed to catch the perceptual differences. Therefore, perceptual loss [5, 6] are encouraged by GANs. And this perceptual loss would use a pre-trained VGG[4] network on ImageNet which is available from tensorflow GitHub [14]. We use both MSE and VGG perceptual loss for comparison in our models. Also, efficient sub-pixel convolutional neural network (ESPCN) [7] introduced a computational-efficient pixel shuffle layer to upscale low resolution (LR) to high resolution (HR) feature maps. We adapted this idea in my model as well.

3 Dataset

For dataset, we use CelebA-HQ [9] dataset which is built upon CelebA [1] dataset but a higher-quality version. This dataset is a large-scale face attributes dataset with 30k celebrity images at 1024x1024 resolution. Most of the images are cropped front face portrait. Before training, the dataset is preprocessed to different resolution: 16x16, 32x32, 64x64, 128x128, 256x256, 512x512. We have split the dataset into 28000/1000/1000 as train/dev/test dataset. For data preprocessing, we have normalized the input images from [0, 255] to [0,1].

In the state-of-the-art SRGAN paper [2], their data sets are Set5, Set14, BSD100 with LR input of 144x144, 256x256 or similar resolution in between. We decide our input LR resolution of 32x32. In our early tests, we experiment with different resolution inputs: 16x16 images lose too much facial information, 64x64 images are good enough to see the whole shape of eyes. So we pick 32x32 images which keeps certain important facial information but no details at all and challenging enough to test the deep learning's limit which traditional methods can never imagine.

4 Methods

Our model follows the guideline of SRGAN paper [2] and PGGAN paper [9] with modification. We implement our own version of SRResnet and SRGAN. Our first model is SRResnet with MSE loss. After explorations, we build our second model: SRGAN with mix loss. Then we further optimize our model by integrate WGAN-GP as our SRWGAN-GP model with mix loss and transfer learning. The fourth model is SRPGGAN with mix loss and WGAN-GP included. The details of the 4 models are discussed in this section.

4.1 Content Loss Function

Content loss is measuring the difference between generated outputs and target images. The most common way for content loss is to calculate the pixel-wise MSE loss as:

$$l_{MSE}^{SR} = \frac{1}{r^2WH}\sum_{x=1}^{rW}\sum_{y=1}^{rH}(I_{x,y}^{HR} - G_{\theta_G}(I^{LR})_{x,y})^2$$

We use MSE loss in our SRResnet model. However, MSE loss tends to generate overly smooth textures for the output images. Then we also explore perceptual loss, which calculates the feature

map differences to preserve the texture details. To extract the feature map, a pre-trained VGG network on ImageNet is used. The perceptual VGG loss is defined as:

$$l^{SR}_{VGG_{i,j}} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y})^2$$

where $W_{i,j}$ and $H_{i,j}$ are the dimensions of the feature maps within the VGG 19 network.

4.2 SRResnet

The SRResnet model, proposed in SRGAN paper [2], is shown in Figure 2. Residual blocks and skip connection are used. Each residual block has two convolutional layers followed by batch norm and parametricRelu layers. For upscale 4x factors, two sub-pixel convolution layers, proposed by Shi et al [7], are added after resnet blocks. We implement SRResnet with MSE as our baseline model for comparison.

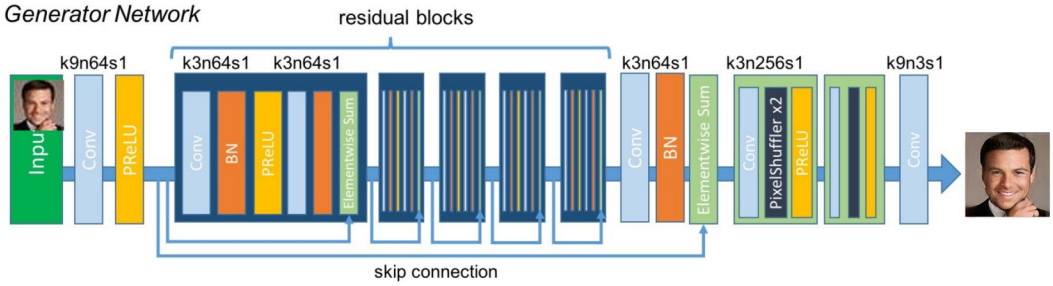

Figure 2: SRResnet model and Generator Network.

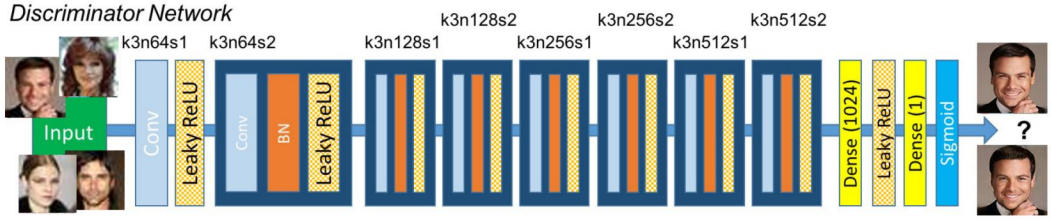

Figure 3: Discriminator Network.

4.3 SRGAN

SRGAN uses the SRResnet model as its generator whose goal is to generate high resolution and high quality images to fool the discriminator. The discriminator is a separate CNN network to classify target images are real and generated outputs are fake as shown in Figure 3. Therefore, the objective function of GAN is a minimax game:

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

where the $I^{HR} \sim p_{train}(I^{HR})$ are the target HR images from the input data, $I^{LR} \sim p_G(I^{LR})$ are the LR images from the input data, D_{θ_D} is the output of the discriminator and G_{θ_G} is the output of the generator. The gradient descent of the generator and discriminator is calculated alternately. The loss function of generator has two components: a content loss l^{SR}_X and an adversarial loss l^{SR}_G:

$$l^{SR} = l^{SR}_X + 10^{-3} l^{SR}_G$$

We have experimented the content loss to be MSE, perceptual VGG loss, and the mix loss off MSE and VGG. We found that mix loss, which adds up MSE and VGG, works best in our

scenario.

4.4 SRWGAN-GP

WGAN tries to solve the problem of model collapse and stabilizes the training process. We use improved WGAN-GP method, proposed by Gulrajani et al [8], to calculate Wasserstein distance and penalize the norm of gradient to enforce a Lipschitz contraint. The objective function is also changed to:

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

WGAN-GP is less sensitive to hyperparameters and more stable to get good image results. For SRWGAN-GP model's content loss, we also use mix loss.

4.5 SRPGGAN

PGGAN introduces a new training strategy of growing both the generator and the discriminator progressively. It starts with a low resolution of 4x4, then gradually adds new layers to the model and generates high resolution images of 1024x1024. We are inspired by this idea and make a SRPGGAN model as shown in Figure 4. We starts with resolution of 32x32. As the training advances, we incrementally increases resolution to 64x64, then to 128x128 for 4x scaling factor, and 256x256 for 8 scaling factor. When doubling the resolution, a transition weight is used to fade in the new layers smoothly and avoid sudden shock to the trained layers. All existing layers stays trainable. Mix loss and WGAN-GP techniques are also used in this model.

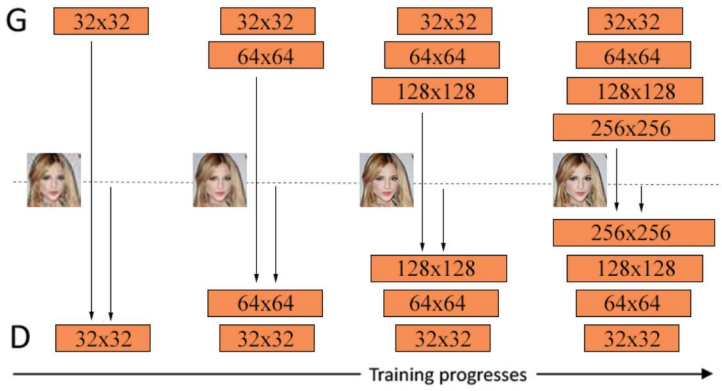

Figure 4: SRPGGAN Networks

5 Experiments

5.1 Hyperparameter Tuning and Feature Selection

For SRResnet model, MSE loss is used for content loss. Regarding hyperparameters, the training iteration number is 200k, the batch size is 8, which is the maximum number to fit our GPU memory. The learning rate is 8e-5 with Adam optimizer. We have tried learning rate from 1e-3 to 1e-5, and 8e-5 works best. The number of the residual blocks is 8. As this number increasing, the net gets deeper and the training time increases. Input size is 32x32, scaling factor is 4x, and output size is 128x128.

For SRGAN model, we test MSE loss first but find output no significant difference with SRResnet model. Then we explore perceptual VGG loss. However, only using the extracted feature map from VGG model would result checkerboard artifacts. Therefore, we create a mix loss, adding up MSE and VGG loss, to remove checkerboard artifacts and improve image quality. SRGAN is using the NSGAN object function. The same hyperparameters as

SRResnet are used here. The adversarial loss weight is 10^{-3}, which is recommended in SRGAN paper. We experiment to increase the weight, the output tends to be more art-stylistic instead of photo-realistic. The Generator and the Discriminator is trained 1 step respectively at each iteration.

For SRWGAN-GP model, we adapt good features of mix loss and hyperparameters from SRGAN model. Compared to SRGAN, WGAN-GP is a replacement of NSGAN. To use WGAN, we remove the last sigmoid layer of discriminator, and remove log calculation in adversarial loss. We test gradient penalty with lambda of 10 to stabilize the training. After getting good results from this model, we also investigate transfer learning: using the pre-trained model of SRResnet as the starting point then train SRWGAN-GP. After training just 20k interactions, We got better results than before. Since only the Generator is pretrained, we want the Discriminator to catch up. Therefore, 3 steps of D and 1 steps of G is set up in the training process.

For SRPGGAN model, the architecture has been changed accordingly as describe in Section 4.5. We set up the structure that we can grow the layers on the run. Also, a transition flag is passed to do the fade-in if transitioning from lower resolution to higher resolution. For content loss, only the last round with targeted resolution, we use mix loss, otherwise MSE loss is used. The 4x SRPGGAN has comparable results as SRWGAN-GP. However, the flexibility to increase the layers to get to higher resolution is very attractive and effective. Therefore, we experiment 8x scaling factor and got results shown in Figure 1.

5.2 Model Evaluation

Figure 5 shows the 4x scaling outputs comparison for different models. SRResnet has much better quality than bicubic interpolation. However, the face texture is overly smoothed and the image looks blurry. SRGAN has better details than SRResnet which means the mix loss is working well. SRWGAN-GP and SRPGGAN both have better results than SRGAN with more details in faces and hairs. Regarding to PSNR, it relates to MSE loss and only good at pixel-wise distance measurement but fails to catch perceptual differences used in GAN. Therefore, SRResnet has the highest PSNR score but not best image quality.

Figure 5: Input LR 32x32, SRResnet 4x output, SRGAN 4x output, SRWGAN-GP 4x output, SRPGGAN 4x output, Origianl HR 128x128

6 Conclusion

We implement both CNN and GAN architectures on the CelebA-HQ dataset in a very similar hyperparameter settings and compare the results side-by-side. We find that GAN models generate more visually pleasing images than CNN models. Then we focus on GAN and experiment with different GAN architectures within SRGAN framework. We have successfully integrate WGAN-GP with SRGAN to form our SRWGAN-GP model, integrate PGGAN to form our SRPGGAN model. Both SRWGAN-GP and SRPGGAN have achieved photo-realistic quality results in 4x scaling. And SRPGGAN shows its powerful capability of extending the model easily to 8x scaling factor and achieve photo-realistic images.

Code is uploaded in github: https://github.com/yshu/SR, and we have sent a collaborator invitation to cs230-stanford.

Acknowledgments

We would like to thank our TA Jay Whang for his feedback during office hours. Also many thanks to all the teaching stuff of CS230 to deliver this great course.

References

[1] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep Learning Face Attributes in the Wild. Proceedings of International Conference on Computer Vision (ICCV), 2018

[2] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, W. Shi. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. arXiv preprint arXiv: 1609.04802, 2017

[3] X. Yu and F. Porikili. Ultra-resolving face images by discriminative generative networks. In European Conference on Computer Vision (ECCV), page 318-333, 2016.

[4] K. Simonyan and A.Zisserman. Very deep convolutional networks for large-scale image recognition. In Internatioanl Conference on Learning Representations (ICLR), 2015.

[5] J. Bruna, P. Sprechmann, and Y. Lecun. Super-resolution with deep convolutional sufficient statistics. In Internatioanl Conference on Learning Representations (ICLR), 2016.

[6] J. Johnson, A. Alahi, and F. Li. Perceptual losses for real-time syle transfer and super-resolution. In European Conference on Computer Vision (ECCV), pages 694-711. Springer, 2016.

[7] W. Shi, J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Ruechert, and Z. Wang. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Netwokr. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1874-1883, 2016.

[8] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville. Improved Training of Wasserstein GANs. arXiv preprint arXiv: 1704.00028, 2017

[9] T. Karras, T. Aila, S. Laine, J. Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. arXiv preprint arXiv: 1710.10196, 2017

[10] C. Dong, C. C. Loy, K. He, X. Tang. Image Super-Resolution Using Deep Convolutional Networks. arXiv preprint arXiv: 1501.00092, 2015

[11] A. Aitken, C. Ledig, L. Theis, J. Caballero, Z. Wang, W. Shi. Checkboard Artifact Free Sub-Pixel Convolution. arXiv preprint arXiv: 1707.02937, 2017

[12] K, He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770-778, 2016

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio. Generative Adversarial Networks. arXiv: 1406.2661, 2014

[14] https://github.com/tensorflow/models/tree/master/research/slim

[15] https://github.com/kostyaev/ICNR