Response to review

From group 53

1. **why do you guys want to use a GAN if SRCNN and SRResNet already have good results? What makes SRGAN a good network despite its complex structure and computational cost?**

   Sure, SRCNN and SRResNet give us acceptable results, and they were widely used for the super resolution field previously. However, GAN method gives us a new perspective, and we are happy to see that the effect of SRGAN is also very good. Although the structure of SRGAN is more complex, it is worthy to find that SRGAN performs much better when recovering subtle detail.

2. **When mentioned "trimmed" to desired input size in page 8, introducing the trimming process in more detail is probably much better. Were the 2K images divided into patches of size 384x384 and 96x96? Or were some other methods used in the preprocessing?**

   For a rare image, first, we use random crop API from tensorflow to cut into a 384x384 image. Second, we use random flip API from tensor flow to flip the image. This is the target image we need. Then we downsample the image into 96*96*3 image, which is the source image for training.

From group 49

1. **We do know that SRCNN and SRResNet are classification models. So what is your training data and corresponding label when training SRCNN and SRResNet? What kind of loss equation do you use to compute the loss value?**

   Our training data is described in the response of Group53 (2), we use the same training data for SRCNN, SRResNet and SRGAN. For the corresponding label, we use a small strick, we treate the 384x384 image, from the first step of producing traning data, as a label. So labels and traning data can match well. For the loss function, we simply use the pixel-wise mean square error (MSE) to compute the loss.

2. **The loss diagram of SRGAN discriminator shows that the discriminator's loss doesn't drop down. It seems the discriminator doesn't converge. You need more analysis on those diagram.**

   It is safe to say that the loss function of SRGAN does not have numerical meaning, which means we actually do not care about whether it will converge or not. We present the plots of loss function to show the game process of generator and discriminator during the training process.

3. **What images do you use in the final model evaluation part?**

   Images from DIV2K dataset

4. **What is the pros and cons among these three models?**

   SRCNN has the most simple structure, thereby cosuming relatively less time for training, but the accuracy of it is the lowest. For SRGAN, the structure of it is the most complex, but it also provides the best visiable results especially in processing subtle detail. For SRResNet, the complexity of the structure is at a median position, the result of it is acceptable in most of cases and have similar, even better scores compared with SRGAN, but when processing subtle lines, the performance of it is not good enough.

5. **For the demo part, could you use Jupyter Notebook to generate the result?**

   Yes, we can.

From group 79

1. **What kind of images are in the dataset? Are they random scenes, or are they all birds, like the example image? We are concerned that there might be issues with transfer learning if there is not enough variety in the dataset.**

   See the response of Group53 (2). The number of image we use for training is 800, and they are all different. Therefore, the variety in the dataset is enough.

2. **Have you considered using a grayscale input rather than a 3-channel RGB input into the GAN? It might be interesting to see how performance is affected.**

   This is an interesting idea, we will consider it. Thank you so much.

3. **One suggestion might be to try running it on an entirely new low-quality dataset and see how well it is able to transfer.**

   This is a really noce suggestion. We have tried another dataset BSD100 and have similar results in our paper.

4. **It would be nice to know how long each model takes to run just so that we can understand the scale of each architecture.**

   It takes around hour to train SRCNN, two hours to train SRResNet, four hours to train SRGAN.

Super-Resolution Image Recovery Using Generative Adversarial Network

Xupeng Yu
xuy004@eng.ucsd.edu

Hongquan Zhang
h4zhang@eng.ucsd.edu

Guoren Zhong
gzhong@eng.ucsd.edu

***Abstract*—In this work, we study the problem of generating a high-resolution (HR) image from a low-resolution (LR) one, which is referred to as super-resolution (SR) problem. Nowadays, motivated by the development of cell phones, an increasing number of people are becoming fans of taking photos. However, due to some technical and knowledge restraints, it is sometimes hard for people to get high-quality photos. Base on this motivation, we implement four methods, which are the traditional bicubic interpolation[1], SRCNN [2], SRResNet [3], and SRGAN [3], then we have comparisons of their performances based on different metrics. Specifically, we evaluate the scores by the peak signal-to-noise ratio (PSNR), structural similarity index (SSIM), and mean-square error (MSE), respectively, and the results show that SRResNet and SRGAN have better performances than the other two methods. Between these two models, SRResNet has a slightly higher scores under all metrics than SRGAN. However, we actually found that the output images of SRGAN had a better recovery in detailed textures when comparing them using our naked eyes. Therefore, we may need to introduce a new way to calculate the scores of each model in future works.**

***Index Terms*—Super-resolution, bicubic interpolation, SRCNN, SRResNet, SRGAN.**

I. Introduction

SUPER-resolution problem becomes increasingly popular due to the development of machine learning techniques. The SR problem is intrinsically a challenging ill-posed problem, where the texture details are usually missing when applying upscaling from a LR image to a HR one. There are many traditional methods in image reconstruction, such as nearest neighbor, bilinear, and bicubic interpolation. Recent works on improving images' quality are focusing on using convolutional neural networks (CNN) to enhance the resolution of images, which in general, is a powerful method. Generative Adversarial Network is a very popular deep learning model recently. Therefore in this work, we will mainly discuss an approach of dealing with Image SR problem by GANs, namely SRGAN, proposed by C. Ledig et al. in [3]. Besides, comparisons of performances measured by different metrics (PSNR, SSIM, and MES) will be provided.

A. Related work

Super-resolution (SR) problem is a traditional problem in the image processing world. A natural way to solve the problem is to use bicubic interpolation, which is discussed in the book *Digital Image Processing* by Rafael C. Gonzalez [11]. In 2014, an idea of using neural networks to solve this problem came into people's mind, and [2] becomes the first paper using neural network called SRCNN to deal with the SR problem. This method directly learns an end-to-end mapping between the low and high-resolution images.

Later, more complicated and deeper networks are studied, such as SRResNet, which makes use of residual learning[4] to deal with the problem of gradient explosion and vanishing in deep neural network. Enhanced Deep Super-Resolution Network (EDSR)[7] is an improved model based on SRResNet, which simply removes the batch normalization layers inside the residual block and the ReLU activation layers outside the residual blocks.

In more recent years, Generative Adversarial Network (GAN) [6] has developed rapidly. A GAN can learn to generate new data with the same statistics as the training set. Due to its architectural nature, the traditional loss function is replaced by a minimax problem. With outstanding performances, GAN is now playing an important role in many computer vision problems.

B. Dataset

In this work, we train and test our models using the DIV2K dataset [5]. It provides 1000 HR images (divided into 800 train images and 200 valid images) and the corresponding LR images with different downscaling types, from which we choose the set of LR images with a bicubic downscaling factor of 4. An example pair from the dataset is shown in Fig.1. Among all these pairs of images, a subset of 200 pairs would be our test data and the rest 800 are used for training.

(a) (b)

Fig. 1: Part of an example pair from the DIV2K dataset. Image (a) is the HR image, and image (b) is the corresponding LR image with a bicubic downscaling factor of 4.

As we know, a neural network can only take inputs with the same size. Besides, the original images are too large in size for training. Therefore, for all the models, we randomly cropped $384 \times 384 \times 3$ parts from HR images and use them

TABLE I: The architecture of SRGAN. (a) is the generator, and (b) is the discriminator.

(a) Generator

		Type	Filters	Size	Output
		Input			96×96
		Conv	64	3×3	96×96
1x	16x	Conv	64	3×3	96×96
		Conv	64	3×3	96×96
		Residual			96×96
		Conv	64	3×3	96×96
		Residual			96×96
		Conv	256	3×3	96×96
		Sub-pixel Conv			192×192
		Conv	256	3×3	192×192
		Sub-pixel Conv			384×384
		Conv	3	3×3	384×384

(b) Discriminator

		Type	Filters	Size	Output
		Input			384×384
		Conv	64	4×4 / 2	192×192
1x	1x	Conv	128	4×4 / 2	96×96
		Conv	256	4×4 / 2	48×48
		Conv	512	4×4 / 2	24×24
		Conv	1024	4×4 / 2	12×12
		Conv	2048	4×4 / 2	6×6
		Conv	1024	1×1	6×6
		Conv	512	1×1	6×6
		Residual	512		
		Conv	128	3×3	6×6
		Conv	128		6×6
		Conv	512		6×6
		Residual	512		
		Flatten			
		Dense	$18,432$		

as inputs. In the meantime, the correlated LR images are also cropped to the size of $96 \times 96 \times 3$ showing the same part of the images as the cropped HR one. The cropping and flipping preprocessing also act as a way of data augmentation.

II. Methods

A. Model

In single image super-resolution (SISR) problem, the goal is to generate a super-resolved image, denoted as I^{SR}, from a LR image I^{LR}, which is a downscaling version of the HR image I^{HR}. During training, I^{HR} and the I^{LR} generated from I^{HR} by applying downsampling operation with downsampling factor of r are fed to the model. Then, with C color channels, I^{LR} has a size of $W \times H \times C$, and I^{HR}, I^{SR} have a size of $rW \times rH \times C$. In this section, we will briefly introduce the bicubic interpolation, SRCNN and SRResNet, and lay most emphasis on SRGAN.

Our main model in this work is a generative adversarial network, so it will be introduced first. Basically, SRGAN is a GAN based on pretrained VGG19 model, and it consists of 2 networks, a generator G_{θ_G} and a discriminator D_{θ_D}. Inspired by Goodfellow et al.[6], the goal of SRGAN can be expressed as an adversarial min-max problem:

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

where $\mathbb{E}$ is the cost function. The idea of (1) is that the discriminator D would try to identify whether an image is real or fake, while the generator G would try to generate such images that can fool the discriminator. Since the discriminator D_{θ_D} can only output the probability for an image to be real, which is a value between 0 and 1, then specifically in (1), the discriminator wants to maximize the objective by driving $D_{\theta_D}(I^{HR})$ to 1 and $D_{\theta_D}(G_{\theta_G}(I^{LR}))$ to 0. In the meantime, the generator wants to minimize the objective by making the discriminator output a larger value for $D_{\theta_D}(G_{\theta_G}(I^{LR}))$.

The architecture of the generator network G is shown in Table.Ia , where the main part of G are B residual blocks. For each block, there are two convolutional layers with 3×3 filters and 64 feature maps followed by batch-normalization layers [8] and activated by ParametricReLU [9]. After these residual blocks, two sub-pixel convolutional layers [10] are applied to increase the resolution of an image. Briefly speaking, an input of r^2 number of channels is needed in order to increase the image's resolution by a factor of r.

Apart from G, a discriminator D is trained to judge whether an image is real by solving the maximization problem in (1). The architecture of D is shown in Table.Ib. The main part of D are 8 convolutional layers, each of which is followed by batch-normalization except the first one, and use a LeakyReLU ($\alpha = 0.2$) as activation. These convolutional layers all equipped with 3×3 filter kernels, but the number of them is increased by a factor of 2 at every other layer from 64 to 512. Besides, stride-2 convolution is applied each time when the number of features is doubled, in order to reduce the resolution of an image.

B. Evaluation Metrics

Four metrics are used for evaluating the performance of SRGAN, which are PSNR,MSE, SSIM,and MOS.

1) MSE: MSE (mean square error) measures the average squared difference between the estimated values and the actual value:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2 \tag{2}$$

(a) Target image

(b) Output of SRCNN

(c) Output of SRResnet

(d) Output of SRGAN

Fig. 2: Test example. (a) is the target HR image; (b) is the image generated by SRCNN; (c) is the image generated by SRResNet; and (d) is the image generated by SRGAN.

2) PSNR: PSNR (peak signal to noise ratio) is an engineering term for the ratio between the maximum possible power of a signal and the power of corrupting noise. Since many signals have very wide dynamic ranges, PSNR is usually expressed in terms of the logarithmic decibel scale as:

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right) \tag{3}$$

3) SSIM: SSIM (structural similarity index) measures the similarity between two images, and the expression for it is:

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{4}$$

4) MOS: PSNR, MSE and SSIM give traditional methods to score images and evaluate the similarity between images, thereby evaluating the resolution in terms of computer vision field. However, based on hundreds of experiments, when resolutions of two images are close, having very similar SPNR, MSE and SSIM scores, it would be hard to tell which image has a higher resolution using these three metrics. Thus, according to the paper [3], we introduce another metric called mean opinion score (MOS) testing, whose value is obtained based on people's judgements by naked eyes. Specifically in this work, we asked 17 raters to assign an integral score from 1 (bad quality) to 5 (excellent quality) to the SR images, then took the mean for the final MOS. This is a very straightforward way to discriminate the higher resolution image.

III. Results

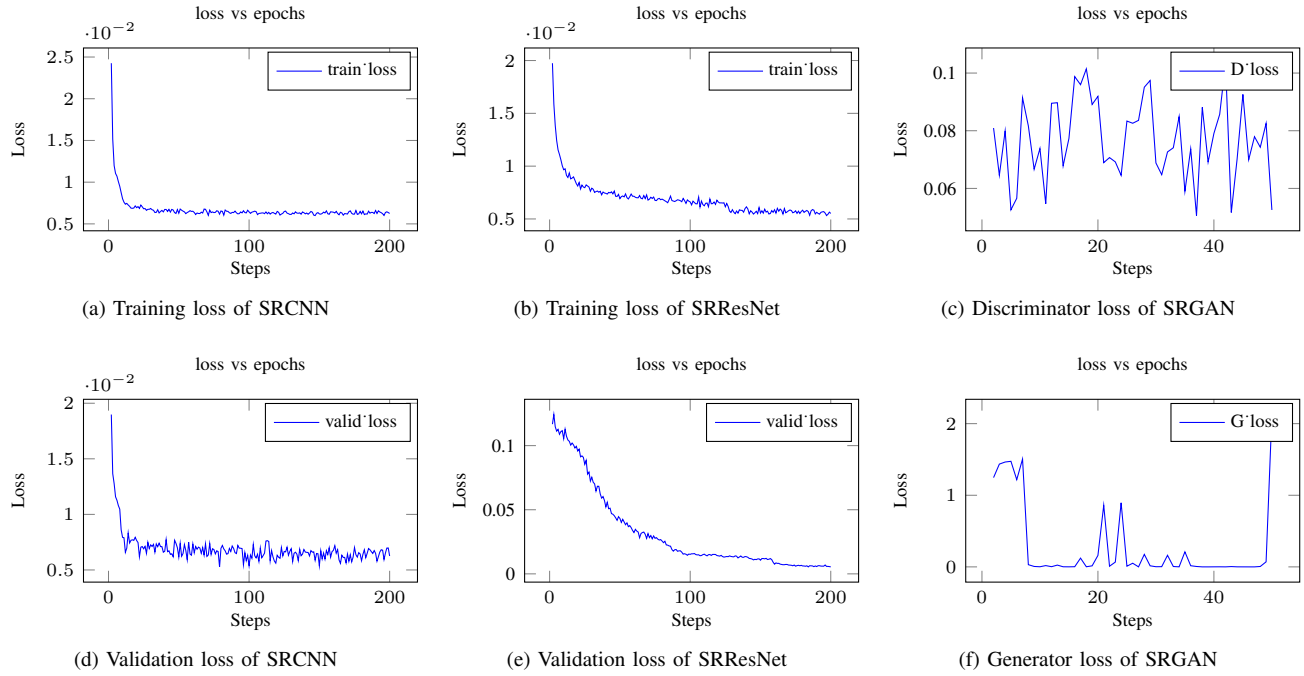
In this work, we trained three models on UCSD datahub, which are SRCNN, SRResNet and SRGAN. Due to the hardware limitation, we fix our batchsize to 8 and set the learning rate to 10^{-5} during trainings. Fig.2a shows our target test image, which is used to compare with outputs from the three models.

A. SRCNN and SRResNet

For SRCNN and SRResNet models, we trained them for 200 epochs, and the output images are shown in Fig.2b and 2c. It can be seen that SRResNet recovers the image better in texture details than SRCNN. Besides, the training and validation loss plots are shown in Fig.3, from which we can see that the losses of SRCNN (Fig.3a and 3d) takes about 20 epochs to converge, while those of SRResNet (Fig.3b and 3e) takes more time to converge. This is foreseeable since SRResNet model is much more complicated than SRCNN model. Despite this, losses of both models converge to a relatively low level eventually.

B. SRGAN

SRGAN is an extremely complex model where it consists of two deep networks. Besides, we also apply learning rate decay during training which means the learning rate will become even smaller after a number of epochs. For these reasons, it takes much more time to train SRGAN than the other two models mentioned above. In this test, we only trained it for 50 epochs, and set the initial learning rate to 10^{-5}, which will become even smaller after 25 epochs, then the output image is shown in Fig.2d. By comparing the outputs carefully, we found that SRGAN can recover even more refined details than SRResNet. The losses are also plotted in Fig.3c and 3f, and they are relatively irregular comparing to the those of SRCNN and SRResNet. In fact, this phenomenon is reasonable because GAN is a minimax game, where the generator wants to minimize the loss and the discriminator wants to maximize it. This irregular nature makes it hard to find useful information from the loss plots, which is also one of the reasons why a GAN is hard to train.



(a) Training loss of SRCNN (b) Training loss of SRResNet (c) Discriminator loss of SRGAN

(d) Validation loss of SRCNN (e) Validation loss of SRResNet (f) Generator loss of SRGAN

Fig. 3: Loss plots for three models. (a) and (d) are the training and validation loss of SRCNN model; (b) and (e) are the training and validation loss of SRResNet; (c) and (f) are the losses of discriminator and generator networks of SRGAN respectively.

C. Comparison of performances

Additionally, we also try the traditional bicubic interpolation method, and the metric scores of all four methods are shown in Table.II, where we can find that all of the machine learning methods perform better than the traditional bicubic interpolation method. Moreover, SRGAN and SRResNet are slightly better than the simple 4-layer SRCNN model. Based on traditional computer vision metrics (PSNR, MSE, SSIM), in most cases, SRResNet has the best performance among all methods. However, in terms of MOS, SRGAN gets a relatively better score than SRResNet, thereby having a highest resolution when processing images, which also matches our conclusion from Fig.2.

metrics	bicubic	SRCNN	SRResNet	SRGAN
MSE	0.0032	0.0019	0.0018	0.0019
PSNR	25.84	26.38	27.46	27.03
SSIM2	0.68	0.75	0.77	0.75
MOS	1.20	1.96	2.406	3.36

TABLE II: Comparison of metrics on different models

IV. Conclusion

For super resolution (SR) problem, we successfully implemented the SRGAN model, which performed outstandingly when processing subtle details and dense lines. Besides, we compared the results of SRGAN with those of another two models, SRCNN and SRResNet, and a traditional method, bicubic interpolation. Based on the tests on two different datasets with enough data, we can conclude that the bicubic interpolation method is not satisfying in SR problem. For the machine learning models, we concluded that the recovery performance is positively related to the complexity of model architectures. In other words, SRCNN gives a baseline for SR problem, and SRGAN performs the best. Compared with SRResNet, the main advantage of SRGAN is that it refines detail much better, while the relevant cost is a longer running time. In future works, if more computational resources are available, we will try to adjust the number of layers in the model and test for more epochs. For example, it is surprising to find that the enhanced deep residual network (EDSR) [7] achieves not only higher accuracy but also lower computational cost by just removing batch normalization layers form the residual blocks. We are looking forward to such exciting findings.

Contribution

We would like to thank Professor Peter Gerstoft for introducing our project team into the topic of machine learning for physical applications. We would also like to thank the team members in this project for the amazing work. Hongquan was responsible for setting up the data loader and pre-processing data for models and help making part of the slides and the report. Guoren delved into the implementation of SRGAN model and proofread the slides and the report. Xupeng implemented the SRCNN and SRResNet models and did the majority part of testing all three models. Without their contribution, this project wouldn't be accomplished.

REFERENCES

[1] Digital Image Processing, 4th Edition. Rafael C. Gonzalez, University of Tennessee. Richard E. Woods, MedData Interactive.

[2] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision (ECCV)*, pages 184–199. Springer, 2014. 3, 6, 8

[3] C. Ledig et al.,"Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 2017, pp. 105-114.

[4] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778.

[5] R. Timofte et al., "NTIRE 2017 Challenge on Single Image Super-Resolution: Methods and Results," *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Honolulu, HI, 2017, pp. 1110-1121.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2672–2680, 2014. 3, 4, 6

[7] Lim, B., Son, S., Kim, H., Nah, S. and Mu Lee, K., 2017. Enhanced deep residual networks for single image super-resolution. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops (pp. 136-144).

[8] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of The 32nd International Conference on Machine Learning (ICML)*, pages 448–456, 2015. 3, 4, 6

[9] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015. 4

[10] W. Shi et al., "Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 1874-1883.

[11] Gonzalez, R. C., Woods, R. E. (2008). Digital image processing. Upper Saddle River, N.J.: Prentice Hall.