Super-Resolution with GAN

Yang Wang, Yangxin Zhong, Qixiang Zhang

leonwy12, yangxin, qixiang

March 2018

1 Introduction

Given a low-resolution (LR) image, trying to recover or estimate its high-resolution (HR) version image is referred to single image super-resolution (SR). There exist methods that recover HR image from multiple LR images, but our project will focus on single image SR. SR is an important topic in computer vision as it can be beneficial to many applications such as texture mapping, enlarging consumer photograph, converting NTSC video content to HDTV, and low-resolution face recognition[3][5].

One challenge of SR problem is that when recovering HR image from a LR image with high upscaling factor, the texture detail of the reconstructed SR image is typically absent[9]. Another challenge is that SR problem is inherently ill-posed, since multiple solutions exist for a given LR image. In other words, it is an underdetermined inverse problem, of which solution is not unique[4]. This makes it harder and ambiguous to evaluate a SR method. Therefore, many different types of methods and evaluation metrics have been proposed so far. We will give a brief overview of some of them in section 2.

In this project, we leverage the power of deep neural network to build several models for super-resolution problem, including SRResnet and SRGAN models[9]. We also succeed to improve the performance of SRGAN by replacing its vanilla GAN with a WGAN-GP model[7]. Other attempts to improve the models, such as ratio tuning strategy and combined content loss, are also explored in this project.

2 Related work

Prediction-based methods were the first methods for SR problem. A naive but straight way to solve SR problem is to use linear interpolation to recover the missing pixels in upscaling SR images. This kind of methods, also called filtering approaches, e.g. bilinear or bicubic, often oversimplify the SR problem and yield solutions with overly smooth textures.

Recently, supervised learning-based methods that leverage the power of deep neural network achieve the state-of-the-art performance in SR problem. In these methods, we have a dataset where the HR version of LR images are known. As a result, the SR problem becomes to learn a mapping model from LR images to its HR version. Recent researches show that enabling the neural network to learn the upscaling filters directly can further increase performance both in terms of accuracy and speed[13]. For example, SRCNN[4] use a 3-layer upscaling convolution neural network (CNN) to learn a end-to-end mapping from LR images to HR images and get promising performance. More recently, a method called SRGAN[9] using generative adversarial networks (GAN) is proposed. By training two different models, i.e. generator and discriminator, and making them compete to each other, this work can achieve even better performance than SRCNN. Additionally, instead of optimizing the traditional pixel-wise mean square error (MSE) loss, it applies a new loss function with a pre-trained CNN to yield HR images of higher human-perceptual quality.

Our project uses the filtering methods as our baseline, and implements SRGAN and train the model on our dataset. Finally, we will compare the performance of different methods and analyze how different architectures and hyper-parameters of deep neural network in SRGAN can affect its performance.

3 Our approach

3.1 Problem Statement

SR problem can be formulated as follows. Given an downsampled image I^{LR} with low resolution, estimate its original high-resolution, super-resolved image I^{HR}. Suppose the input I^{LR} is of spatial size $W \times H$ and has C color channels, the output I^{HR} is of size $rW \times rH \times C$, where r is the downsampling factor.

3.2 SRGAN

We will apply Super Resolution Generative Adversarial Network (SRGAN) [9] to solve this problem. The model consists of two parts. The first part is a generator network that estimates high-resolution images (HR) from low-resolution images(LR). The second part is a discriminator that distinguish between nature HR and the generated image. We use a deep residual network (ResNet) as generator, and a deep convolutional network as discriminator. The problem can be formularized as an adversarial minimax problem:

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

Here D is the discriminative network with parameters θ_D, and G is the generative network with parameters θ_G.

3.3 Generator Network

The architecture of the generator of SRGAN is shown in Figure 1. It adopts the deep convolutional neural net, Residual Network (ResNet)[8], as the architecture. It contains B residual blocks with identical layout (the skip connections). Specifically, each residual block contains two convolutional layers with 64 filters of size 3×3 followed by batch-normalization layers and Parametric ReLU as the activation function. It increases the resolution of the input image with two trained sub-pixel convolutional layers[11].

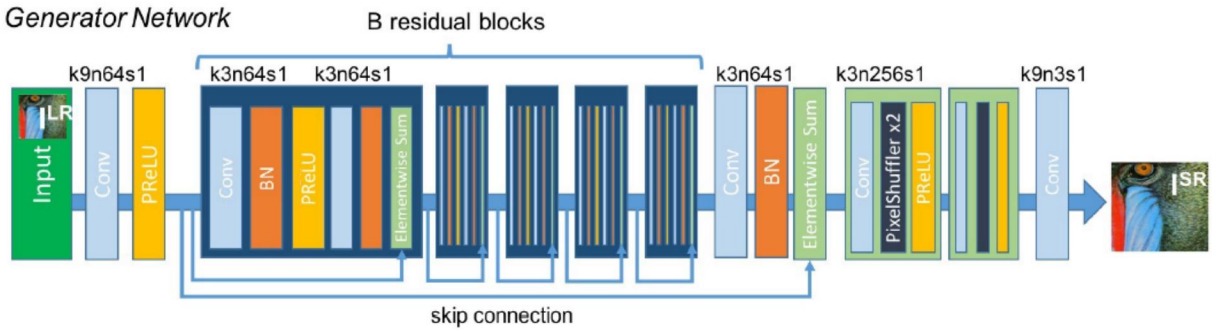


Figure 1: Architecture of Discriminator Network with corresponding kernel size (k), number of feature maps (n) and stride (s) for each convolutional layer.

3.4 Discriminator Network

The architecture of the discriminator of SRGAN is shown in Figure 2. 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 as in the VGG network[12]. The resulting 512 feature maps are followed by two dense layers and a final sigmoid activation function to obtain a probability for sample classification. Note that Leaky ReLU ($\alpha = 0.2$) is used as activation function and no max-pooling is used in this network.

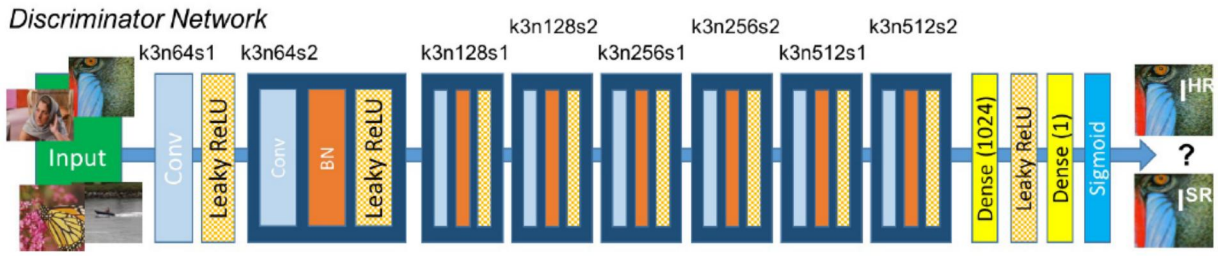


Figure 2: Architecture of Generator Network with corresponding kernel size (k), number of feature maps (n) and stride (s) for each convolutional layer.

3.5 Generator Loss

The loss function of generator network, or generator loss l_{Gen}, is presented as follows:

$$l_{Gen} = l_{Con} + \gamma \cdot l_{Adv}$$

Basically the generator loss contains two components: content loss l_{Con} and adversarial loss l_{Adv} with a scaling factor γ set to 10^{-3} in the original paper.

The content loss l_{Con} shows content of difference between generated HR and authentic HR images during training process. Traditional methods use pixel-wise error measurements such as mean squared error (MSE) loss for this part:

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

However, the generator trained by such pixel-wise MSE loss tends to yield HR images with overly smooth textures which lack high-frequency content, resulting in perceptual dissatisfaction. To solve this problem, SRGAN uses a pre-trained 19 layer VGG network as feature extractor, and define content loss l_{Con} as the mean squared error of feature maps extracted from generated and authentic HR images output from certain layer in the network. Suppose we are using the j-th convolutional layer before the i-th max-pooling layer within the VGG network as the feature extractor, denoted by $\phi_{i,j}$, and let $W_{i,j}$ and $H_{i,j}$ denote the width and height of feature maps of that layer. The VGG loss can be denoted by

$$l_{VGG/i,j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} \left[\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{SR}))_{x,y} \right]^2$$

The adversarial loss l_{Adv} is another component of the generator loss. It is presented as follows:

$$l_{Adv} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(I^{LR}))$$

Here, N is the number of examples in each batch or mini-batch and I^{LR} denotes different LR examples; $D_{\theta_D}(G_{\theta_G}(I^{LR}))$ can be seen as the probability output by the discriminator that indicates how likely it thinks the generated image $G_{\theta_G}(I^{LR})$ is actually a natural HR image. By minimizing l_{Adv}, the generator tries to "fool" the discriminator by generating highly natural HR images. For better gradient behavior, it's usual to minimize $-\log D_{\theta_D}(G_{\theta_G}(I^{LR}))$ instead of $\log \left[1 - D_{\theta_D}(G_{\theta_G}(I^{LR}))\right]$ when training GAN.

3.6 Discriminator Loss

The discriminator loss l_{Dis} is presented as follows:

$$l_{Dis} = \sum_{n=1}^{N} -\log D_{\theta_D}(I^{HR}) + \sum_{n=1}^{N} -\log \left[1 - D_{\theta_D}(G_{\theta_G}(I^{LR}))\right]$$

To minimize l_{Dis}, the discriminator tries to improve itself to discriminating the generated HR images from the natural HR images. On the other hand, the generator will try to improve itself to "fool" this strong discriminator by generating highly natural HR images, which results in higher human perceptual satisfaction in the generated images.

3.7 SRGAN improvement

3.7.1 Discriminator Regularization: Label Smoothing and Input Noise

When training SRGAN with MSE content loss, we found that the discriminator loss converges to a pretty low value very quickly and the adversarial loss of generator goes way too high, this is probably due to an overfitting issue in discriminator. That is, the discriminator perfectly learns the distribution of real HR images and gets too strong for the generator to fool it. There are several ways to relieve this issue [6], including using a label smoothing technique to change the real images label for discriminator from a hard label (1.0) to a soft label (such as 0.9), and adding small Gaussian noise to the input of real images (i.e. the pixels) to make it harder for discriminator to learn. In our experiments, we employ these two techniques as the discriminator regularization, and explore their effects to the performance of SRGAN.

3.7.2 Combined Content Loss

Note that different content losses (i.e. MSE loss and VGG losses from different layers) focus on different aspects of the image quality and they all make sense. In our experiment, we try to incorporate them to get a combined content losses to improve the SRGAN model performance. More concretely, we use the follwing combined content loss:

$$l_{Con} = w_{MSE} \cdot l_{MSE} + w_{VGG/2,2} \cdot l_{VGG/2,2} + w_{VGG/5,4} \cdot l_{VGG/5,4},$$

where w_{MSE}, $w_{VGG/2,2}$ and $w_{VGG/5,4}$ are tunable hyperparameters.

3.7.3 Ratio Tuning in Generator Loss

Since the ratio γ in generator loss (i.e. $l_{Gen} = l_{Con} + \gamma \cdot l_{Adv}$) controls how much the generator would like to improve itself to "fool" the discriminator rather than to recover the HR images, it's necessary to explore the effect of different value of γ instead of using a fixed 10^{-3} proposed in the original paper. In our experiments, the following strategy is explored: we first train generator with a lower γ so as to make it focus more on recovering content of HR images, then we tune up γ and continue training to make it focus more on generating indistinguishable HR images from the real ones.

3.7.4 WGAN-GP

When training GAN, it's usually hard to tell the convergence of the model with loss functions defined above; also, the generator suffer from mode collapse issue (i.e. generator collapses into very narrow distribution and can only generate similar results). Wasserstein GAN (WGAN) [1] was recently proposed to address these two problems. The main differences of WGAN compared to a vanilla GAN are: (1) it removes the logarithm in l_{Adv} and l_{Dis}; (2) it removes the sigmoid output layer in the discriminator; (3) it clips the parameter weights of discriminator to satisfy Lipschitz constraint; (4) it trains discriminator more than generator.

However, WGAN can still generate only poor samples or fail to converge due to the weight clipping [7]. Recently, A model called Wasserstein GAN with gradient penalty (WGAN-GP) was proposed to address this issue. To satisfy Lipschitz constraint, this work adds a gradient penalty term to the discriminator loss, instead of using weight clipping. Their experiments show that WGAN-GP can provide a metric of training convergence, result in a more stable training, and also effectively reduce the mode collapse issue.

In this project, we replace the vanilla GAN with WGAN-GP in SRGAN model to see if it can improve its performance. We name this new model as SRWGAN-GP. We also combine other techniques stated above, i.e. combined content loss and ratio tuning, to see if we can further improve the performance of SRWGAN-GP.

4 Dataset and Evaluation Metrics

4.1 Dataset

We used the the RAISE dataset, which consists of 8156 pictures from categories such as "outdoor", "indoor", "landscape", "nature", "people", "objects" and "buildings". The high resolution pictures are paired with their 4x downsampled counterparts to form our training set. We split the data into three parts: 400 for dev set, 400 for test set, and the rest are our training data. During training, we also applied random crop with crop size 24 by 24.

4.2 Evaluation Metrics

There are many different ways to evaluate a super-resolution models. In this project, we used following metrics to evaluate our models.

4.2.1 MSE (Mean Squared Error)

This is the most simple measurements, which is defined by the pixel to pixel squared error.

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

Note that we only used y-channel here in order to be consistent with the paper[9].

4.2.2 PSNR (Peak Signal-to-Noise Ratio)

$$PSNR = 10 \cdot \log_{10}(\frac{MAX_I^2}{MSE}),$$

where MAX_I is the maximum possible pixel value of the image.

MSE and PSNR are widely used optimization objectives, and PSNR is roughly an denoised version of MSE. However, MSE and PSNR may suffer from over smoothed (blurry) results that ignores the detailed texture of the original images[9]. We may use more sophisticated metrics that are more satisfying for human perception as well.

4.2.3 SSIM (Structural Similarity)

The Structural Similarity[14] (SSIM) Index quality assessment index is based on the computation of three terms, namely the luminance term, the contrast term and the structural term. The overall index is a multiplicative combination of the three terms.

$$SSIM(x, y) = [l(x, y)]^\alpha [c(x, y)]^\beta [s(x, y)]^\gamma,$$

where

$$l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1},$$

$$c(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2},$$

$$s(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3},$$

Here μ_x, μ_y, σ_x, σ_y and σ_{xy} are local means, standard deviation and covariance of two windows x and y. We will set the α, β, γ to be 1, and $C_3 = \frac{1}{2}C_2$.

4.2.4 MOS (Mean Opinion Score)

Although we could use quantitative metrics to evaluate our models, sometimes we may prefer human opinions when it comes to super resolution results, as there are no metric can represent the satisfaction level of human perception enough. In this project, we also conduct MOS test among humans for rating the realisticity of our super resolved images from different models.

5 Experiment Results

In our experiments, we implemented bicubic and bilinear interpolation methods as baselines, and we trained SRResnet, SRGAN and SRWGAN-GP on our dataset. We base our project on a Tensorflow SRGAN implementation[2]. The source code of our project is also uploaded to a Github repository[10].

5.1 Model Hyperparameters and Training Details

5.1.1 SRResnet

In our experiments, we trained SRResnet for 180,000 iterations with a mini-batch size 16 in each iteration. We employed Adam optimizer to train our model with a learning rate of 10^{-4}. The number of residual blocks we used is $B = 16$. We have tried MSE, VGG54, and combined loss as the content loss for SRResnet. However, SRResnet model with VGG54 loss turns out to generate images with unrealistic texture since it only optimizes the high level features loss. As a result, we only report the result of SRResnet model with MSE and combined content loss. we use $w_{MSE} = 5$, $w_{VGG/2,2} = 10^{-4}$ and $w_{VGG/5,4} = 0$ in the combined content loss.

5.1.2 SRGAN

After the training of SRResnet, we then trained the SRGAN model with MSE content loss for 100,000 iterations. We used our trained SRResnet as the pretrained generator of SRGAN, and the discriminator was trained from scratch. The hyperparameter settings are exactly the same as SRResnet. The ratio of adversarial loss we used in the generator loss is $\gamma = 10^{-3}$. We also apply discriminator regularization techniques, i.e. label smoothing and input noise, to SRGAN with MSE loss. Same settings are used to trained these models.

We also train SRGAN model with VGG54 content loss for 100,000 iterations, using SRResnet as pretrained generator. Same hyperameter settings are used here, including $\gamma = 10^{-3}$. Note that we follow the original paper to use a rescaled content loss $l_{con} = w_{VGG/5,4} \cdot l_{VGG/5,4}$, where $w_{VGG/5,4} = 6.1 \times 10^{-3}$. We then use both the generator and discriminator of this model as the pretrained models of SRGAN with ratio tuning strategy and SRGAN with combined content loss. We train each of these two models for another 100,000 iterations. In ratio tuning strategy, we tune the ratio up to $\gamma = 5 \times 10^{-3}$. In combined content loss, we use $w_{MSE} = 10^{-2}$, $w_{VGG/2,2} = 10^{-4}$ and $w_{VGG/5,4} = 6.1 \times 10^{-3}$.

5.1.3 SRWGAN-GP

In SRWGAN-GP, we replace the vanilla GAN model with a WGAN-GP model. We follow the hyperparameter settings in the WGAN-GP paper[7]: gradient penalty weight $\lambda_{GP} = 10$, Adam optimizer hyperparameters $\beta_1 = 0.5$ and $\beta_2 = 0.9$. In each iteration, we train discriminator for 3 steps before training generator for 1 step. For SRWGAN-GP model with MSE content loss, we use the SRResnet as pretrained model, and train it for 20,000 iterations. For SRWGAN-GP model with VGG54 content loss, we use the SRGAN(VGG54) as pretrained model, and train it for 20,000 iterations. After training this model, we use it as the pretrained model to continue training with ratio tuning strategy and combined content loss, respectively. We train each of these two models for another 20,000 iterations. In ratio tuning strategy, we tune the ratio up to $\gamma = 5 \times 10^{-3}$. In combined content loss, we use $w_{MSE} = 10^{-2}$, $w_{VGG/2,2} = 10^{-4}$ and $w_{VGG/5,4} = 6.1 \times 10^{-3}$.

5.2 Model Evaluations and Analysis

The model evaluation results are shown in Table 1. Among these evaluation metrics, a lower MSE, and higher PSNR, SSIM or MOS values indicate a better performance. From the table we can see that while the

baseline methods, bilinear and bicubic interpolations, achieved better-than-nothing performance by averaging on neighboring grids, SR-Resnet, SRGAN and SRWGAN-GP achieved significantly better performance. The SRResnet model with MSE content loss achieved the best MSE, PSNR and SSIM performance. This is consistent with the result of the original paper[9].

However, as pointed out earlier, a model achieving high performance in these pixel-based metrics often generate overly smooth images which result in perceptual dissatisfaction. As a result, when evaluating a model in this section, we will focus more on its Mean Opinion Score (MOS)[9], which is an averaged human rating score based on the clarity and realisticity of the recovered HR images. In general, MOS represents the human-perceptual quality of the generated images.

As we can see in Table 1, the SRGAN model with VGG54 loss achieves a high MOS, which is consistent with the result in the original paper[9]. However, two of the SRWGAN-GP models that we propose achieve even higher MOS performance. We will discuss about these results in details in the following subsections.

Model	Bilinear	Bicubic	SRResnet(MSE)	SRResnet(Combined)
MSE	50.18	50.21	**40.92**	41.83
PSNR	31.49	31.48	**32.59**	32.48
SSIM	0.617	0.621	**0.758**	0.746
MOS	3.33	3.34	4.45	5.02

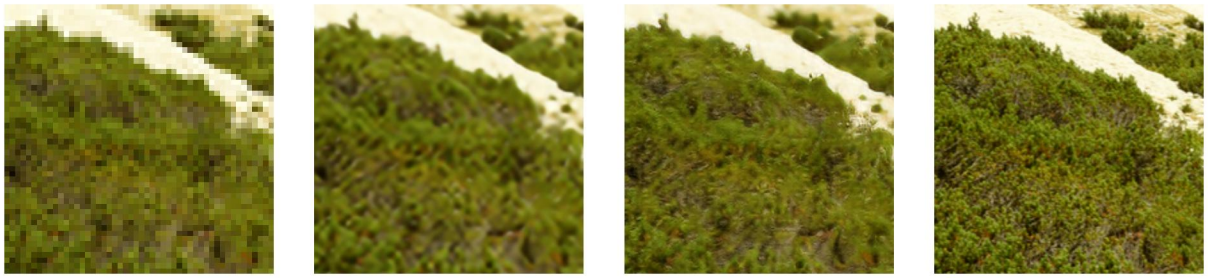
Model	SRGAN(MSE)	SRGAN(MSE+LS)	SRGAN(MSE+IN)	SRGAN(VGG54)	SRGAN(VGG54+RT)	SRGAN(Combined)
MSE	41.28	44.17	43.08	57.41	57.71	53.13
PSNR	32.54	32.19	32.31	30.77	30.74	31.15
SSIM	0.754	0.730	0.736	0.676	0.658	0.677
MOS	4.87	5.81	5.50	**7.05**	6.85	6.86

Model	SRWGAN-GP(MSE)	SRWGAN-GP(VGG54)	SRWGAN-GP(VGG54+RT)	SRWGAN-GP(Combined)
MSE	44.70	60.22	68.96	60.60
PSNR	32.04	30.51	29.82	30.48
SSIM	0.733	0.687	0.665	0.685
MOS	5.59	6.90	**7.32**	**7.34**

Table 1: Performance of Different Super-Resolution Models (Combined: combined content loss. LS: label smoothing. IN: input noise. RT: ratio tuning strategy.)

5.2.1 Effects of GAN

SRResnet is the model with only the generator network, but without discriminator. It is trained with content loss only, which can be the pixel-wise MSE loss, VGG loss, or a combined content loss. As we can see in Table 1, the SRResnet(MSE) model achieves the best MSE, PSNR and SSIM performance. However, the generated images of it have overly smooth texture and lack high-frequency content, as shown in Figure 3. In comparison, the SRGAN(MSE) model using a discriminator network can sharpen the texture and result in a higher perceptual satisfaction. This is because the real HR images are full of high-frequency components; under the GAN training, in order to "fool" the discriminator and make the generated images indistinguishable from real images, the generator is trained to generate images with high-frequency components as well. This shows the power of GAN while training the generator network.



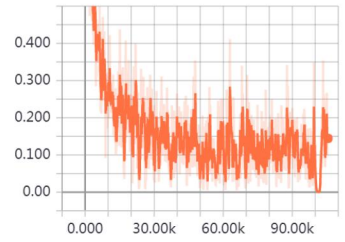
(a) Input LR Image (b) SRResnet(MSE) (c) SRGAN(MSE) (d) Original HR Image

Figure 3: Comparison between SRResnet (no discriminator) and SRGAN (using discriminator)

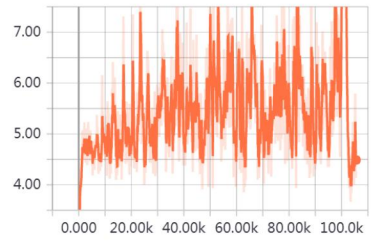
5.2.2 Effects of Discriminator Regularization

In Table 1, we found that SRGAN(MSE) model is only slightly better than SRResnet(MSE) model, in terms of MOS, which indicates the human-perceptual quality of generated images. This is not expected. With the power of discriminator, SRGAN should perform much better than SRResnet, given that they both employ the pixel-wise MSE content loss. We then dived into the plot of training loss of SRGAN(MSE), shown in Figure 4. Note that the discriminator loss converged quickly to a low value around 0.1, and the adversarial loss, which is part of the generator loss, fluctuated around a high value around 6 all the time. As stated in Section 3.7.1, this may indicate an overfitting issue of discriminator. So we tried to use two different techniques, i.e. label smoothing (LS) and input noise (IN), to regularize the discriminator (see Section 3.7.1).

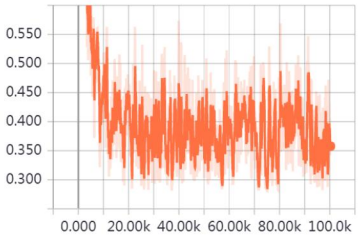
With these regularization techniques, e.g. label smoothing (LS), we found that the discriminator loss still quickly converged but to a much higher value (0.4) than before, as shown in Figure 4. This shows that label smoothing, which involves noise in the real image labels, actually relieve the overfitting issue of discriminator. We also found that the adversarial loss seems to remain almost the same as before. This is good because it means regularizing the discriminator does not harm the training of generator, since the adversarial loss (and its backprop gradient) is still high enough for generator to learn. Actually from Table 1, we can see a clear improvement in SRGAN(MSE+LS) and SRGAN(MSE+IN) compared to SRGAN(MSE) without a discriminator regularization, in terms of MOS. Moreover, there is a huge gap in MOS between SRGAN(MSE+LS) and SRResnet(MSE). This is exactly the gap that we expect to see between GAN and non-GAN models. Figure 5 gives an example of the visual effects of images generated from these models.



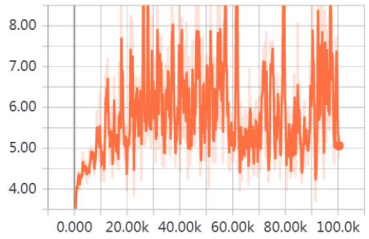
(a) SRGAN(MSE) Discriminator loss



(b) SRGAN(MSE) Adversarial loss



(c) SRGAN(MSE+LS) Discriminator loss



(d) SRGAN(MSE+LS) Adversarial loss

Figure 4: Discriminator and Adversarial loss of SRGAN(MSE) and SRGAN(MSE+LS)

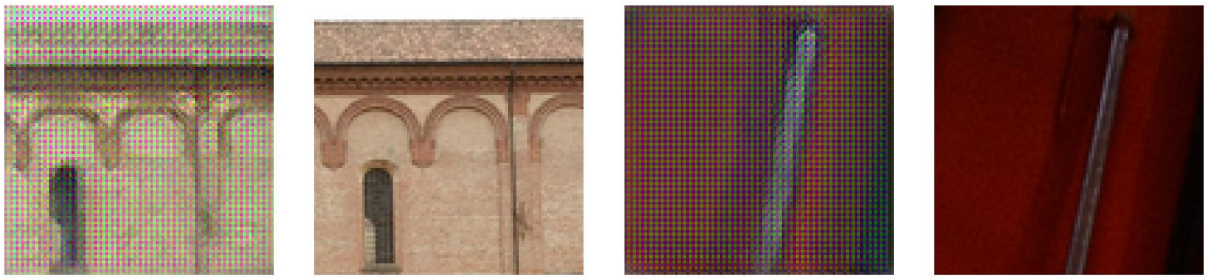
(a) SRGAN(MSE) (b) SRGAN(MSE+IN) (c) SRGAN(MSE+LS) (d) SRResnet(MSE)

Figure 5: Visual effects of SRGAN with/without label smoothing (LS) and input noise (IN)

5.2.3 Effects of Combined Content Loss

There are three types of content loss employed in the model training: pixel-wise MSE, VGG22, and VGG54 loss, as stated in Section 3.5. They focus on different aspects of content when recovering an image: MSE loss focuses on pixel-wise similarity; VGG22 loss uses feature vector of image output from lower layer of the VGG network, so it focuses on the low-level texture of an image; and VGG54 loss is from higher layer of the VGG network, so it focuses on similarity of the high-level texture (or even small object) between images.

They all make sense in some extent but at the same time with different issues. Models trained with MSE content loss can yield overly smooth texture as we stated earlier. Model trained with VGG content loss, which focuses on local texture and ignores the pixel-wise similarity, may bring some texture artifacts to the generated images. This issue is especially severe in SRResnet model, which is trained without a discriminator (see Figure 6). This is why we did not report the result of SRResnet(VGG54) in Table 1 (the original paper[9] did not report, either). However, if the GAN idea is employed and the model is trained with a discriminator, then these texture artifacts can be easily prevented. From Table 1, we can find that for GAN-based models, the ones using VGG54 loss have much higher MOS performance than the ones using pixel-wise MSE loss, since the formers focus more on texture contents.



(a) SRResnet(VGG54) (b) Original HR Image (c) SRResnet(VGG54) (d) Original HR Image

Figure 6: Texture artifacts in images generated by SRResnet (no discriminator) with VGG54 content loss

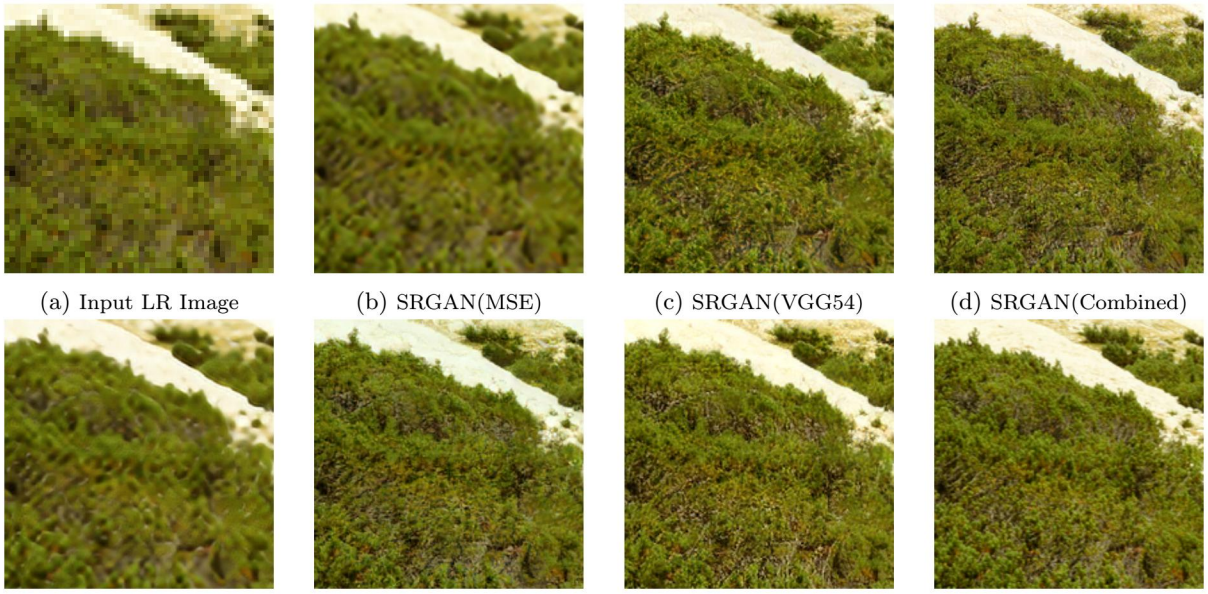
Since different types of content loss focus on different aspects, a content loss considering all of them in a weighted sum way may combine different advantages they have. However, there is no guarantee that using combined content loss will improve the model performance. This is due to two reasons: (1) When training generator in GAN, the generator loss contains both content loss and adversarial loss, and it's hard to understand the combination effects between these two types of losses; (2) three hyperparameters, i.e. w_{MSE}, $w_{VGG/2,2}$ and $w_{VGG/5,4}$, are required to be tuned in combined content loss, so an optimal combination is difficult to be found.

In our experiments, it turns out that combined content loss improves the MOS performance of SRResnet and SRWGAN-GP models, but it does not help SRGAN model (see Table 1). Moreover, the SRWGAN-GP models with combined content loss achieves the highest MOS performance among all models in our experiments. Figure 7 gives an example of the visual effects of models using different content losses.

5.2.4 Effects of Ratio Tuning

As stated in Section 3.7.3, the ratio γ in $l_{Gen} = l_{Con} + \gamma \cdot l_{Adv}$ controls how much the generator would like to improve itself to "fool" the discriminator rather than to recover similar content of HR images. Ideally, if a model could recover the exact HR images, then it could also "fool" the discriminator perfectly. However, this is not always true for our deep neural net models. The L2-based content loss is not a perfect metric for content similarity: we have seen that in order to optimize content loss, the model will yield either overly smooth texture or texture artifacts. But without the constraints of content loss, the generator can yield arbitrary texture and content so long as it can "fool" discriminator; in this case, the generated images could be very different from original HR images. As a result, it is important to find a good way to balance between content loss l_{Con} and adversarial loss l_{Adv}, with the ratio hyperparameter γ.

In our experiments, we found that a γ around 10^{-3} as suggested in the original paper[9] is indeed a good choice. Moreover, we explore the following ratio tuning strategy: we first train generator with a lower γ (say



(a) Input LR Image (b) SRGAN(MSE) (c) SRGAN(VGG54) (d) SRGAN(Combined)

(e) SRWGAN-GP(MSE) (f) SRWGAN-GP(VGG54) (g) SRWGAN-GP(Combined) (h) Original HR Image

Figure 7: Visual effects of SRGAN and SRWGAN-GP with different content losses

10^{-3}) so as to make it focus more on recovering content of HR images, then we continue training with a tuned up γ (say 5×10^{-3}) to remove some constraints of content loss to make it focus more on generating indistinguishable HR images from the real ones. This is kind of like the pretraining plus fine tuning strategy commonly used in neural net training. Intuitively this can make sense, but there is no guarantee that this training trick will actually improve the model performance.

We explore this strategy in the training of SRGAN(VGG54) and SRWGAN-GP(VGG54). As shown in Table 1, ratio tuning (RT) improves the MOS performance of SRWGAN-GP(VGG54) model (and make it one of the best models among all) but does not help SRGAN(VGG54) model. This may be due to WGAN-GP model can provide a better and more stable training for the generator than the vanilla GAN model, but yet the reason of this effect is not fully understood. Figure 8 gives an example of the improvement of SRWGAN-GP model using ratio tuning strategy. Zoom in the image generated by SRWGAN-GP(VGG54+RT) model, we can find that it contains some similar tiny texture as in the original HR image, but the vanilla SRWGAN-GP(VGG54) model fails to produce such textures.

(a) Input LR Image (b) SRWGAN-GP(VGG54) (c) SRWGAN-GP(VGG54+RT) (d) Original HR Image

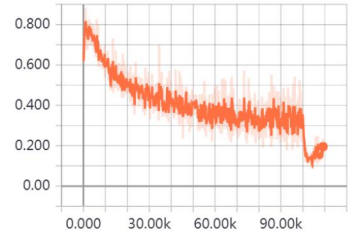
Figure 8: Improvement of SRWGAN-GP(VGG54) model using ratio tuning strategy

5.2.5 Effects of WGAN-GP

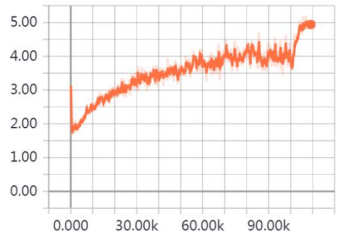
We replace the vanilla GAN in SRGAN model[9] with WGAN-GP[7] to get our own model SRWGAN-GP. Due to the WGAN-GP model, the training loss of SRWGAN-GP has completely different behavior compared

to SRGAN. As shown in Figure 9, the discriminator (adversarial) loss of SRGAN with VGG54 content loss is steadily decreasing (increasing) during the training. The sudden decrease (increase) of loss at the end of training is nothing special but due to the decayed learning rate. This indicates that the discriminator in SRGAN grows stronger steadily and makes the generator harder and harder to "fool" it.

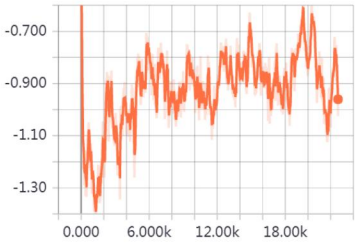
But if we look into the training loss of SRWGAN-GP model with VGG54 content loss in Figure 9, we will find that the discriminator loss first decreases dramatically and then increases steadily and finally converged to a value around -0.9. Note that compared to GAN, the WGAN-GP model removes the sigmoid output layer in discriminator and also removes the logarithm in the discriminator loss function[7]. As a result, the discriminator loss in SRWGAN-GP is just the difference between the unnormalized scores of generated images and real HR images. So, to minimize this discriminator loss is equivalent to maximize the gap between unnormalized scores of real HR images and generated images. Since the discriminator loss is negative all the time, it means unnormalized scores of real images are always higher than the generated images, which is expected. However, the gap is becoming closer and closer to 0 during the training. This means the generator is actually getting better and better to "fool" the discriminator. In the paper of WGAN[1], it is proved that this discriminator loss can actually be used as an indicator of GAN training convergence. A higher discriminator loss indicates a better GAN training. So the loss in Figure 9 indicates that everything goes well in our WGAN-GP training. Note that the magnitude of adversarial loss in SRWGAN-GP is much larger than SRGAN model. But since the key in WGAN-GP training is the difference between real image and generated image scores, the values of these unnormalized scores, one of which is the negative adversarial loss, are not very important to the training.



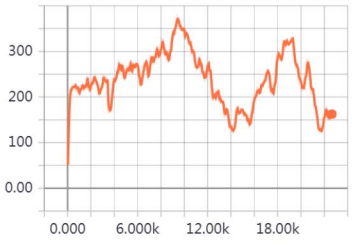
(a) SRGAN(VGG54) Discriminator loss



(b) SRGAN(VGG54) Adversarial loss



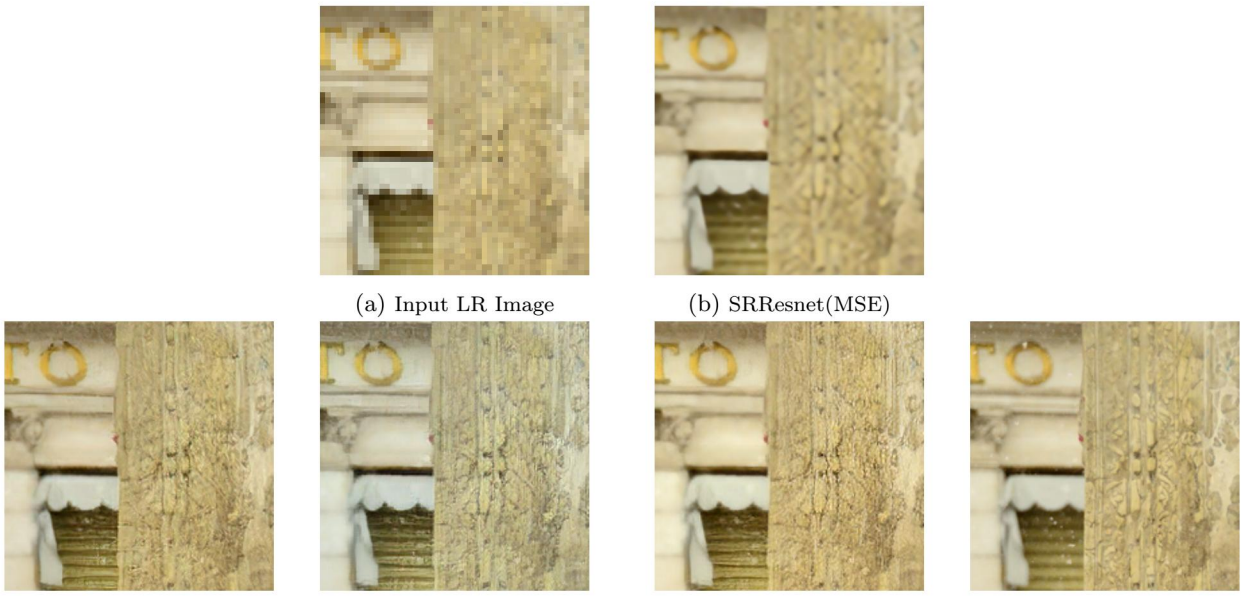
(c) SRWGAN-GP(VGG54) Discriminator loss



(d) SRWGAN-GP(VGG54) Adversarial loss

Figure 9: Discriminator and Adversarial loss of SRGAN(VGG54) and SRWGAN-GP(VGG54)

As shown in papers of WGAN[1] and WGAN-GP[7], the WGAN model can provide a more stable training and relieve the mode collapse issue. However, it does not mean WGAN will necessarily benefit the SRGAN model. From Table 1, we find that using a WGAN-GP model to replace vanilla GAN in SRGAN does not always improve the MOS performance. For example, SRWGAN-GP(MSE) has a higher MOS than SRGAN(MSE) but lower MOS than SRGAN(MSE+LS), which employs label smoothing as discriminator regularization; and SRWGAN-GP(VGG54) has a slightly lower MOS than SRGAN(VGG54). However, with ratio tuning strategy or combined content loss, SRWGAN-GP model can become better than the SRGAN model. In Table 1, we can see that SRWGAN-GP(VGG54+RT) and SRWGAN-GP(Combined) have even higher MOS than the SRGAN(VGG54) model, and become the best models among all. This indicates that SRWGAN-GP can provide stable GAN training, and thus is a promising model to combine with other advanced techniques. Figure 10 gives an example of the comparison of the best SRGAN and SRWGAN-GP models.



(a) Input LR Image (b) SRResnet(MSE)

(c) SRGAN(VGG54) (d) SRWGAN-GP(VGG54+RT) (e) SRWGAN-GP(Combined) (f) Original HR Image

Figure 10: Comparison of visual effects of the best SRGAN and SRWGAN-GP models

5.3 Model Visual Effects

In Figure 11 and Figure 12 in the Appendix, we also show the visual effects of different models evaluating HR from LR images. As we can see, the SRResnet model which achieves the best MSE, PSNR and SSIM values generates overly smooth textures, while more advanced models like SRGAN and SRWGAN-GP can generate images with much higher human-perceptual quality. That is, the high-frequency textures generated by these models result in a high perceptual satisfaction in human eyes.

6 Conclusion and Future Work

To conclude, we focus on the single image super-resolution task in this project. Our method is based on the state-of-the-art SRGAN[9], which is a deep neural net model with GAN. We managed to improve the performance of SRGAN model with different techniques such as discriminator regularization, combined content loss, and ratio tuning strategy. Moreover, we replace the vanilla GAN in the SRGAN model with a WGAN-GP[7] to get our SRWGAN-GP model. Our experiments show that the GAN-based models have much higher performance than the traditional bilinear/bicubic interpolation methods, and also perform much better than the deep neural net model, SRResnet, without GAN, in terms of human-perceptual quality of the images generated. Experiments also show that our SRWGAN-GP model with a combined content loss or ratio tuning strategy performs even better than the state-of-the-art SRGAN model on our dataset.

In the future, We will keep improving our SRWGAN-GP model with other GAN training techniques or other advanced GAN architectures. In addition, we are also interesting in exploring a more effective way to combine different types of content loss and the adversarial loss for generator rather than a naive weighted sum method.

Our code is available at: https://github.com/leonwyang/cs230project

7 Contributions

All three members in this team work together and contribute equally to this project in algorithm designing, model training, method evaluating, result analyzing, and project reporting.

References

[1] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. *CoRR*, abs/1701.07875, 2017.

[2] Ruanting Lin (brade31919 in Github). Srgan-tensorflow. `https://github.com/brade31919/SRGAN-tensorflow`, 2017.

[3] Debabrata Chowdhuri, Sendhil Kumar K. S, M Rajasekhara Babu, and Ch. Pradeep Reddy. Very low resolution face recognition in parallel environment.

[4] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(2):295–307, February 2016.

[5] William T. Freeman, Thouis R. Jones, and Egon C Pasztor. Example-based super-resolution. *IEEE Comput. Graph. Appl.*, 22(2):56–65, March 2002.

[6] Ian J. Goodfellow. NIPS 2016 tutorial: Generative adversarial networks. *CoRR*, abs/1701.00160, 2017.

[7] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[9] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint*, 2016.

[10] Yang Wang (leonwyang in Github). cs230project. `https://github.com/leonwyang/cs230project`, 2018.

[11] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *CoRR*, abs/1609.05158, 2016.

[12] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[13] Y. Wang, L. Wang, H. Wang, and P. Li. End-to-End Image Super-Resolution via Deep and Shallow Convolutional Networks. *ArXiv e-prints*, July 2016.

[14] H. R. Sheikh Z. Wang, A. C. Bovik and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing, 13(4):600–612*, 2004.

(a) Input LR Image (b) SRResnet(MSE) (c) SRGAN(MSE+LS) (d) SRGAN(VGG54)

(e) SRWGAN-GP(VGG54) (f) SRWGAN-GP(VGG54+RT) (g) SRWGAN-GP(Combined) (h) Original HR Image

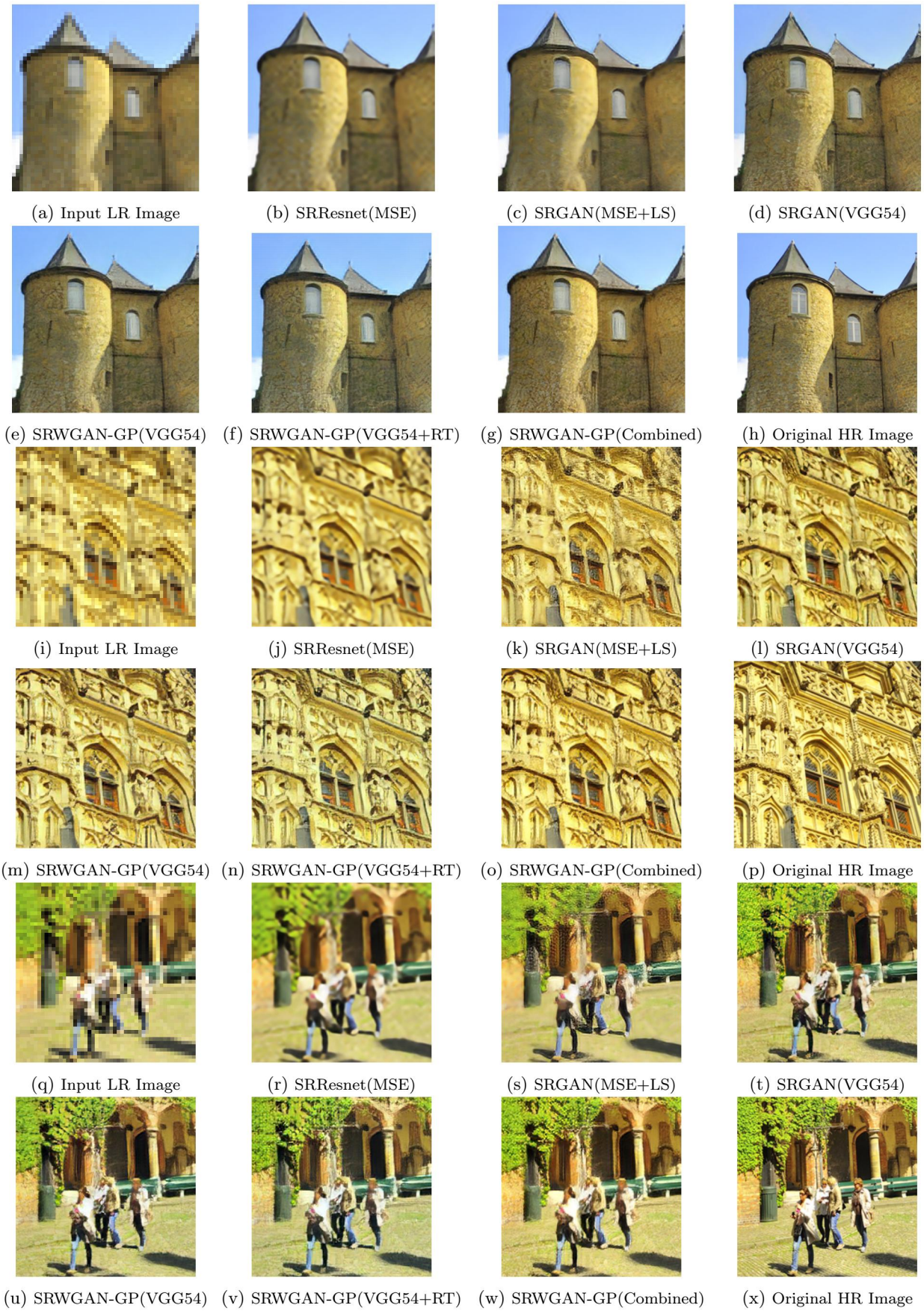
(i) Input LR Image (j) SRResnet(MSE) (k) SRGAN(MSE+LS) (l) SRGAN(VGG54)

(m) SRWGAN-GP(VGG54) (n) SRWGAN-GP(VGG54+RT) (o) SRWGAN-GP(Combined) (p) Original HR Image

(q) Input LR Image (r) SRResnet(MSE) (s) SRGAN(MSE+LS) (t) SRGAN(VGG54)

(u) SRWGAN-GP(VGG54) (v) SRWGAN-GP(VGG54+RT) (w) SRWGAN-GP(Combined) (x) Original HR Image

Figure 11: Comparison of visual effects of images generated from different models



(a) Input LR Image (b) SRResnet(MSE) (c) SRGAN(MSE+LS) (d) SRGAN(VGG54)

(e) SRWGAN-GP(VGG54) (f) SRWGAN-GP(VGG54+RT) (g) SRWGAN-GP(Combined) (h) Original HR Image

(i) Input LR Image (j) SRResnet(MSE) (k) SRGAN(MSE+LS) (l) SRGAN(VGG54)

(m) SRWGAN-GP(VGG54) (n) SRWGAN-GP(VGG54+RT) (o) SRWGAN-GP(Combined) (p) Original HR Image

(q) Input LR Image (r) SRResnet(MSE) (s) SRGAN(MSE+LS) (t) SRGAN(VGG54)

(u) SRWGAN-GP(VGG54) (v) SRWGAN-GP(VGG54+RT) (w) SRWGAN-GP(Combined) (x) Original HR Image

Figure 12: Comparison of visual effects of images generated from different models