RankSRGAN: Generative Adversarial Networks with Ranker for Image Super-Resolution

Wenlong Zhang[1] Yihao Liu[1,2] Chao Dong[1,†] Yu Qiao[1]
[1]ShenZhen Key Lab of Computer Vision and Pattern Recognition, SIAT-SenseTime Joint Lab, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
[2]University of Chinese Academy of Sciences
{wl.zhang1, yh.liu4, chao.dong, yu.qiao}@siat.ac.cn

Abstract

Generative Adversarial Networks (GAN) have demonstrated the potential to recover realistic details for single image super-resolution (SISR). To further improve the visual quality of super-resolved results, PIRM2018-SR Challenge employed perceptual metrics to assess the perceptual quality, such as PI, NIQE, and Ma. However, existing methods cannot directly optimize these indifferentiable perceptual metrics, which are shown to be highly correlated with human ratings. To address the problem, we propose Super-Resolution Generative Adversarial Networks with Ranker (RankSRGAN) to optimize generator in the direction of perceptual metrics. Specifically, we first train a Ranker which can learn the behavior of perceptual metrics and then introduce a novel rank-content loss to optimize the perceptual quality. The most appealing part is that the proposed method can combine the strengths of different SR methods to generate better results. Extensive experiments show that RankSRGAN achieves visually pleasing results and reaches state-of-the-art performance in perceptual metrics. Project page: https://wenlongzhang0724.github.io/Projects/RankSRGAN

1. Introduction

Single image super resolution aims at reconstructing/generating a high-resolution (HR) image from a low-resolution (LR) observation. Thanks to the strong learning capability, Convolutional Neural Networks (CNNs) have demonstrated superior performance [10, 24, 42] to the conventional example-based [40] and interpolation-based [41] algorithms. Recent CNN-based methods can be divided into two groups. The first one regards SR as a reconstruction problem and adopts MSE as the loss function to achieve high PSNR values. However, due to the conflict between the reconstruction accuracy and visual quality, they tend to produce overly smoothed/sharpened images. To favor better visual quality, the second group casts SR as an image generation problem [22]. By incorporating the perceptual loss [6, 18] and adversarial learning [22], these perceptual SR methods have potential to generate realistic textures and details, thus attracted increasing attention in recent years.

Figure 1. The comparison of RankSRGAN and the state-of-the-art perceptual SR methods on ×4. NIQE: lower is better. PSNR: higher is better.

The most challenging problem faced with perceptual SR methods is the evaluation. Most related works resort to user study for subjectively evaluating the visual quality [2, 34]. However, without an objective metric like PSNR/SSIM, it is hard to compare different algorithms on a fair platform, which largely prevents them from rapid development. To address this issue, a number of no-reference image quality assessment (NR-IQA) metrics are proposed, and some of them are proven to be highly correlated with human ratings [2], such as NIQE [29] (correlation 0.76) and PI [2] (correlation 0.83). Specially, the PIRM2018-SR challenge [2] introduced the PI metric as perceptual criteria and successfully ranked the entries. Nevertheless, most of these NR-IQA metrics are not differentiable (e.g., they include handcrafted feature extraction or statistic regression operation), making them infeasible to serve as loss functions. Without

†Corresponding author (e-mail: chao.dong@siat.ac.cn)

considering NR-IQA metrics in optimization, existing perceptual SR methods could not show stable performance in the orientation of objective perceptual criteria.

To overcome this obstacle, we propose a general and differentiable model – Ranker, which can mimic any NR-IQA metric and provide a clear goal (as loss function) for optimizing perceptual quality. Specifically, Ranker is a Siamese CNN that simulates the behavior of the perceptual metric by learning to rank approach [7]. Notably, as NR-IQA metrics have various dynamic ranges, Ranker learns their output ranking orders instead of absolute values. Just like in the real world, people tend to rank the quality of images rather than give a specific value. We equip Ranker with the standard SRGAN model and form a new perceptual SR framework – RankSRGAN (Super-Resolution Generative Adversarial Networks with Ranker). In addition to SRGAN, the proposed framework has a rank-content loss using a well-trained Ranker to measure the output image quality. Then the SR model can be stably optimized in the orientation of specific perceptual metrics.

To train the proposed Ranker, we prepare another training dataset by labeling the outputs of different SR algorithms. Then the Ranker, with a Siamese-like architecture, could learn these ranking orders with high accuracy. The effectiveness of the Ranker is largely determined by the selected SR algorithms. To achieve the best performance, we adopt two state-of-the-art perceptual SR models – SRGAN [22] and ESRGAN [35]. As the champion of PIRM2018-SR challenge [2], ESRGAN is superior to SRGAN on average scores, but can not outperform SRGAN on all test images. When evaluating with NIQE [29], we obtain mixed orders for these two methods. Then the Ranker will favor different algorithms on different images, rather than simply classifying an image into a binary class (SRGAN/ESRGAN). After adopting the rank-content loss, the generative network will output results with higher ranking scores. In other words, the learned SR model could combine the better parts of SRGAN and ESRGAN, and achieve superior performance both in perceptual metric and visual quality. Figure 1 shows an example of RankSRGAN, which fuses the imagery effects of SRGAN and ESRGAN and obtains better NIQE score.

We have done comprehensive ablation studies to further validate the effectiveness of the proposed method. First, we distinguish our Ranker from the regression/classification network that could also mimic the perceptual metric. Then, we train and test RankSRGAN with several perceptual metrics (i.e. NIQE [29], Ma [26], PI [2]). We further show that adopting different SR algorithms to build the dataset achieves different performance. Besides, we have also investigated the effect of different loss designs and combinations. With proper formulation, our method can clearly surpass ESRGAN and achieve state-of-the-art performance.

In summary, the contributions of this paper are threefold. (1) We propose a general perceptual SR framework – RankSRGAN that can optimize generator in the direction of indifferentiable perceptual metrics and achieve the state-of-the-art performance. (2) We, for the first time, utilize results of other SR methods to build training dataset. The proposed method combines the strengths of different SR methods and generates better results. (3) The proposed SR framework is highly flexible and produce diverse results given different rank datasets, perceptual metrics, and loss combinations.

2. Related work

Super resolution. Since Dong et al. [10] first introduced convolutional neural networks (CNNs) to the SR task, a series of learning-based works [40, 15, 20, 16, 13, 14] have achieved great improvements in terms of PSNR. For example, Kim et al. [20] propose a deep network VDSR with gradient clipping. The residual and dense block [24, 42] are explored to improve the super-resolved results. In addition, SRGAN [22] is proposed to generate more realistic images. Then, texture matching [31] and semantic prior [34] are introduced to improve perceptual quality. Furthermore, the perceptual index [2] consisting of NIQE [29] and Ma [26] is adopted to measure the perceptual SR methods in the PIRM2018-SR Challenge at ECCV [2]. In the Challenge, ESRGAN [35] achieves the state-of-the-art performance by improving network architecture and loss functions.

CNN for NR-IQA. No-reference Image Quality Assessment (NR-IQA) can be implemented by learning-based models, which extract hand-crafted features from Natural Scene Statistics (NSS), such as CBIQ [37], NIQE [29], and Ma [26], etc. In [23], Li et al. develop a general regression neural network to fit human subjective opinion scores with pre-extracted features. Kang et al. [19, 4] integrate a general CNN framework which can predict image quality on local regions. In addition, Liu et al. [25] propose RankIQA to tackle the problem of lacking human-annotated data in NR-IQA. They first generate large distorted images in different distortion level. Then they train a Siamese Network to learn the rank of the quality of those images, which can improve the performance of the image quality scores.

Learning to rank. It has been demonstrated that learning to rank approach is effective in computer vision. For instance, Devi Parikh et al. [30] model relative attributes using a well-learned ranking function. Yang et al. [36] first employ CNN for relative attribute ranking in a unified framework. One of the most relevant studies to our work is RankCGAN [32], which investigates the use of GAN to tackle the task of image generation with semantic attributes. Unlike standard GANs that generate the image from noise input (CGAN [28]), RankCGAN incorporates a pairwise Ranker into CGAN architecture so that it can handle continuous attribute values with subjective measures.

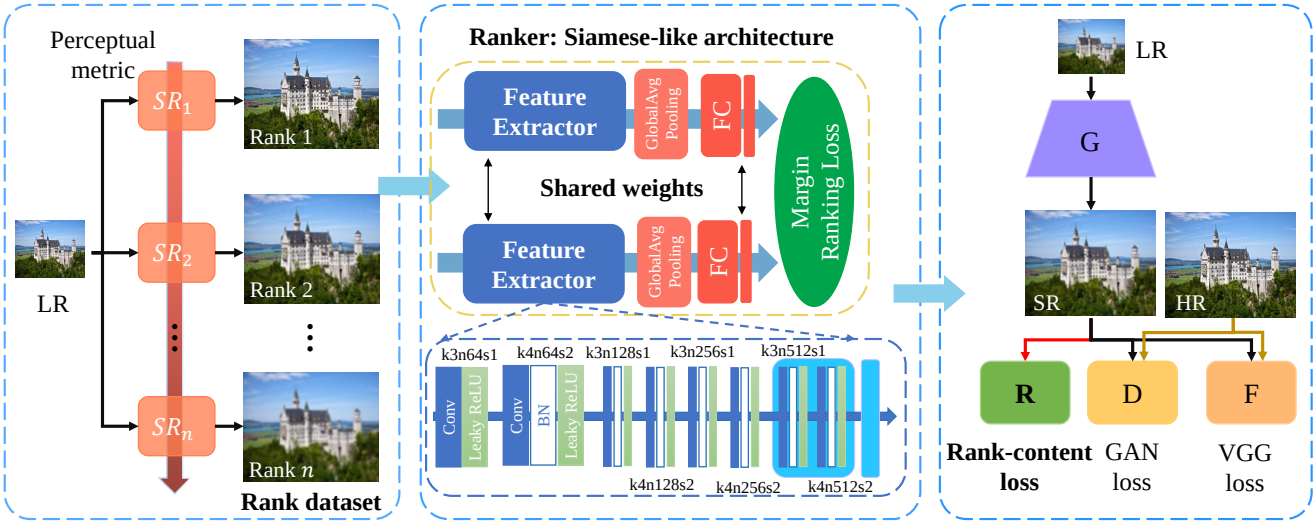


Figure 2. **Overview of the proposed method. Stage 1:** Generate pair-wise rank images by different SR models in the orientation of perceptual metrics. **Stage 2:** Train Siamese-like Ranker network. **Stage 3:** Introduce rank-content loss derived from well-trained Ranker to guide GAN training. RankSRGAN consists of a generator(G), discriminator(D), a fixed Feature extractor(F) and Ranker(R).

3. Method

3.1. Overview of RankSRGAN

The proposed framework is built upon the GAN-based [22] SR approach, which consists of a generator and a discriminator. The discriminator network tries to distinguish the ground-truth images from the super-resolved results, while the generator network is trained to fool the discriminator. To obtain more natural textures, we propose to add additional constraints on the standard SRGAN [22] by exploiting the prior knowledge of perceptual metrics to improve the visual quality of output images. The overall framework of our approach is depicted in Figure 2. The pipeline involves the following three stages:

Stage 1: Generate pair-wise rank images. First, we employ different SR methods to generate super-resolved images on public SR datasets. Then we apply a chosen perceptual metric (e.g. NIQE) on the generated images. After that, we can pick up two images of the same content to form a pair and rank the pair-wise images according to the quality score calculated by the perceptual metric. Finally, we obtain the pair-wise images and the associated ranking labels. More details will be presented in Section 4.1.

Stage 2: Train Ranker. The Ranker adopts a Siamese architecture to learn the beheviour of perceptual metrics and the network structure is depicted in Section 3.2. We adopt margin-ranking loss, which is commonly used in "learning to rank" [7], as the cost function to optimize Ranker. The learned Ranker is supposed to have the ability to rank images according to their perceptual scores.

Stage 3: Introduce rank-content loss. Once the Ranker is well-trained, we use it to define a rank-content loss for a standard SRGAN to generate visually pleasing images. Please see the rank-content loss in Section 3.3.

3.2. Ranker

Rank dataset. Similar to [8, 25], we use super-resolution results of different SR methods to represent different perceptual levels. With a given perceptual metric, we can rank these results in a pair-wise manner. Picking any two SR images, we can get their ranking order according to the quality score measured by the perpetual metric. These pair-wise data with ranking labels form a new dataset, which is defined as the rank dataset. Then we let the proposed Ranker learn the ranking orders. Specifically, given two input images y_1, y_2, the ranking scores s_1 and s_2 can be obtained by

$$s_1 = R(y_1; \Theta_R) \tag{1}$$

$$s_2 = R(y_2; \Theta_R), \tag{2}$$

where Θ_R represents the network weights and $R(.)$ indicates the mapping function of Ranker. In order to make the Ranker output similar ranking orders as the perceptual metric, we can formulate:

$$\begin{cases} s_1 < s_2 & if \quad m_{y_1} < m_{y_2} \\ s_1 > s_2 & if \quad m_{y_1} > m_{y_2} \end{cases}, \tag{3}$$

where m_{y_1} and m_{y_2} represent the quality scores of image y_1 and image y_2, respectively. A well-trained Ranker could guide the SR model to be optimized in the orientation of the given perceptual metric.

Siamese architecture. The Ranker uses a Siamese-like architecture [5, 9, 38], which is effective for pair-wise inputs. The architecture of Ranker is shown in Figure 2. It

has two identical network branches which contain a series of convolutional, LeakyReLU, pooling and full-connected layers. Here we use a Global Average Pooling layer after the Feature Extractor, thus the architecture can get rid of the limit of input size. To obtain the ranking scores, we employ a fully-connected layer as a regressor to quantify the rank results. Note that we do not aim to predict the real values of the perceptual metric since we only care about the ranking information. Finally, the outputs of two branches are passed to the margin-ranking loss module, where we can compute the gradients and apply back-propagation to update parameters of the whole network.

Optimization. To train Ranker, we employ margin-ranking loss that is commonly used in sorting problems [36, 25]. The margin-ranking loss is given below:

$$\begin{aligned} L(s_1,s_2;\gamma) &= \max(0,(s_1-s_2)*\gamma+\varepsilon) \\ &\begin{cases} \gamma = -1 & if \quad m_{y_1} < m_{y_2} \\ \gamma = 1 & if \quad m_{y_1} > m_{y_2} \end{cases}, \end{aligned} \tag{4}$$

where the s_1 and s_2 represent the ranking scores of pair-wise images. The γ is the rank label of the pair-wise training images. The margin ε can control the distance between s_1 and s_2. Therefore, the N pair-wise training images can be optimized by:

$$\begin{aligned} \hat{\Theta} &= \arg\min_{\Theta_R} \frac{1}{N}\sum_{i=1}^{N} L(s_1^{(i)}, s_2^{(i)}; \gamma^{(i)}) \\ &= \arg\min_{\Theta_R} \frac{1}{N}\sum_{i=1}^{N} L(R(y_1^{(i)};\Theta_R), R(y_2^{(i)};\Theta_R); \gamma^{(i)}) \end{aligned} \tag{5}$$

3.3. RankSRGAN

RankSRGAN consists of a standard SRGAN and the proposed Ranker, as shown in Figure 2. Compared with existing SRGAN, our framework simply adds a well-trained Ranker to constrain the generator in SR space. To obtain visually pleasing super-resolved results, adversarial learning [22, 31] is applied to our framework where the generator and discriminator are jointly optimized with the objective given below:

$$\min_{\theta}\max_{\eta} E_{y\backsim p_{HR}} log D_\eta(y) + E_{y\backsim p_{LR}} log(1 - D_\eta(G_\theta(x))), \tag{6}$$

where p_{HR} and p_{LR} represent the probability distributions of HR and LR samples, respectively. In order to demonstrate the effectiveness of the proposed Ranker, we do not use complex architectural designs of GAN [35] but use the general SRGAN [22].

Perceptual loss. In [12, 18], the perceptual loss is proposed to measure the perceptual similarity between two images. Instead of computing distances in image pixel space, the images are first mapped into feature space and the perceptual loss can be presented as:

$$L_P = \sum_i ||\phi(\hat{y}_i) - \phi(y_i)||_2^2, \tag{7}$$

where $\phi(y_i)$ and $\phi(\hat{y}_i)$ represent the feature maps of HR and SR images, respectively. Here ϕ is obtained by the 5-th convolution (before maxpooling) layer within VGG19 network [33].

Adversarial loss. Adversarial training [22, 31] is recently used to produce natural-looking images. A discriminator is trained to distinguish the real image from the generated image. This is a minimax game approach where the generator loss L_G is defined based on the output of discriminator:

$$L_G = -log D(G(x_i)), \tag{8}$$

where x_i is the LR image, $D(G(x_i))$ represents the probability of the discriminator over all training samples.

Rank-content loss. The generated image is put into the Ranker to predict the ranking score. Then, the rank-content loss can be defined as:

$$L_R = sigmoid(R(G(x_i))), \tag{9}$$

where $R(G(x_i))$ is the ranking score of the generated image. A lower ranking score indicates better perceptual quality. After applying the sigmoid function, L_R represents ranking-content loss ranging from 0 to 1.

3.4. Analysis of Ranker

The proposed Ranker possesses an appealing property: by elaborately selecting the SR algorithms and the perceptual metric, the RankSRGAN has the potential to surpass the upper bound of these methods and achieve superior performance. To validate this comment, we select the state-of-the-art perceptual SR methods – SRGAN [22] and ESRGAN [35] to build the rank dataset. Then we use the perceptual metric NIQE [29] for evaluation. NIQE is demonstrated to be highly correlated with human ratings and easy to implement. A lower NIQE value indicates better perceptual quality. When measured with NIQE on the PIRM-Test [2] dataset, the average scores of SRGAN and ESRGAN are 2.70 and 2.55, respectively. ESRGAN obtains better NIQE scores for most images but not all images, indicating that SRGAN and ESRGAN have mixed ranking orders with NIQE.

In order to examine the effectiveness of our proposed Ranker, we compare two ranking strategies – metric rank and model classification. Metric rank, which is our proposed method, uses perceptual metrics to rank the images. For example, in each image pair, the one with a lower score is labeled to 1 and the other is 2. The model classification, as the comparison method, ranks images according to the used SR methods, i.e., all results of ESRGAN are labeled to 1 and those of SRGAN are labeled to 2. We then give an analysis of the upper bound of these two methods. The upper bound can be calculated as:

$$\begin{aligned} UB_{MC} &= Mean(PM_{SR2-L} + PM_{SR2-H}) \\ UB_{MR} &= Mean(PM_{SR2-L} + PM_{SR1-L}) \\ where &: PM_{SR1-L} < PM_{SR2-H}, \end{aligned} \tag{10}$$

where UB_{MC} and UB_{MR} represent the upper bound of model classification and metric rank, respectively. PM

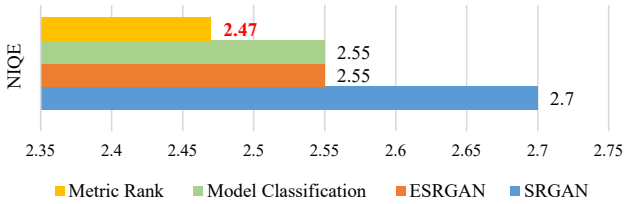


Figure 3. The upper bound (average NIQE value) of SRGAN, ESRGAN, model rank and model classification.

(Perceptual Metric) is the perceptual score for each image in the corresponding class. (SR1, SR2) represents two SR results of the same LR. Subscripts $-L$ and $-H$ indicate the Lower and Higher perceptual score in (SR1, SR2). We use (SRGAN, ESRGAN) as (SR1, SR2) to obtain the upper bound of these methods, as shown in Figure 3. Obviously, metric rank could combine the better parts of different algorithms and exceed the upper bound of a single algorithm.

We further conduct SR experiments to support the above analysis. We use the metric rank and model classification approach to label the rank dataset. Then the Ranker-MC (model classification) and Ranker-MR (metric rank) are used to train separate RankSRGAN models. Figure 4 shows the quantitative results (NIQE), where RankSRGAN-MR outperforms ESRGAN and RankSRGAN-MC. This demonstrates that our method can exceed the upper bound of all chosen SR algorithms.

4. Experiments

4.1. Training details of Ranker

Datasets. We use DIV2K (800 images) [1] and Flickr2K (2650 images) [1] datasets to generate pair-wise images as rank dataset for training. Three different SR algorithms (SRResNet [22], SRGAN [22] and ESRGAN [35]) are used to generate super-resolved images as three perceptual levels, as shown in Table 1.

PIRM-Test	SRResNet	SRGAN	ESRGAN
NIQE	5.968	2.705	2.557
PSNR	28.33	25.62	25.30

Table 1. The performance of super-resolved results with three perceptual levels in PIRM-Test [2]

We extract patches from those pair-wise images with a stride of 200 and size of 296×296. For one perceptual level (SR algorithm), we can generate 150 K patches (10% for validation, 90% for training). Inspired by PIRM2018-SR Challenge [2], we use NIQE [29] as the perceptual metric, while other metrics will be investigated in Section 4.4. Finally, we label every image pair to (3,2,1) according to the order of corresponding NIQE value (the one with the best NIQE value is set to 1).

Implementation details. As shown in Figure 2, we utilize VGG [33] structure to implement the Ranker [25], which includes 10 convolutional layers, a series of batch

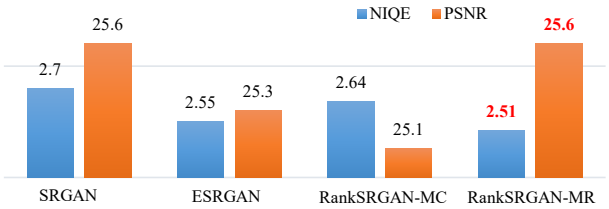


Figure 4. The NIQE of RankSRGAN-MR exceeds that of SRGAN, ESRGAN and RankSRGAN-MC.

normalization and LeakyReLU operations. Instead of max-pooling, we apply convolutional layer with a kernel size 4 and stride 2 to downsample the features. In one iteration, two patches with different perceptual levels are randomly selected as the input of Ranker. For optimization, we use Adam [21] optimizer with weight decay 1×10^{-4}. The learning rate is initialized to 1×10^{-3} and decreases with a factor 0.5 of every 10×10^4 iterations for total 30×10^4 iterations. The margin ϵ of margin-ranking loss is set to 0.5. For weight initialization, we use He. [17] method to initialize the weights of Ranker.

Evaluation. The Spearman Rank Order Correlation Coefficient (SROCC) [25] is a traditional evaluation metric to evaluate the performance of image quality assessment algorithms. In our experiment, SROCC is employed to measure the monotonic relationship between the label and the ranking score. Given N images, the SROCC is computed as:

$$SROCC = 1 - \frac{6\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N(N^2-1)}, \tag{11}$$

where y_i represents the order of label, and $\hat{y}_i$ is the order of output score of the Ranker. SROCC has the ability to measure the accuracy of Ranker. The larger value of SROCC represents the better accuracy of Ranker. For validation dataset, the ranker achieves a SROCC of 0.88, which is an adequate performance compared with those in the related work [8, 25].

4.2. Training details of RankSRGAN

We use the DIV2K [1] dataset to train RankSRGAN. The patch sizes of HR and LR are set to 296 and 74, respectively. For testing, we use benchmark datasets Set14 [39], BSD100 [27] and PIRM-test [2]. PIRM-test is used to measure the perceptual quality of SR methods in PIRM2018-SR [2]. Following the settings of SRGAN [22], we employ a standard SRGAN [22] as our base model. The generator is built with 16 residual blocks, and the batch-normalization layers are removed [35]. The discriminator utilizes the VGG network [33] with ten convolutional layers. The mini-batch size is set to 8. At each training step, the combination of loss functions (Section 3.3) for the generator is:

$$L_{total} = L_P + 0.005L_G + 0.03L_R, \tag{12}$$

where the weights of L_G and L_R are determined empirically to obtain high perceptual improvement[8, 22, 35]. The

Dataset	Metric	Bicubic	FSRCNN	SRResNet	SRGAN	ESRGAN	RankSRGAN (ours)
Set14	NIQE	7.61	6.92	6.12	3.82	3.28	**3.28**
	PI	6.97	6.16	5.36	2.98	2.61	**2.61**
	PSNR	26.08	27.66	**28.57**	26.68	26.39	26.57
BSD100	NIQE	7.60	7.11	6.43	3.29	3.21	**3.01**
	PI	6.94	6.17	5.34	2.37	2.27	**2.15**
	PSNR	25.96	26.94	**27.61**	25.67	25.72	25.57
PIRM-Test	NIQE	7.45	6.86	5.98	2.71	2.56	**2.51**
	PI	7.33	6.02	5.18	2.09	1.98	**1.95**
	PSNR	26.45	27.57	**28.33**	25.60	25.30	25.62

Table 2. Average NIQE [29], PI [2] and PSNR values on the Set14 [39], BSD100 [27] and PIRM-Test [2].

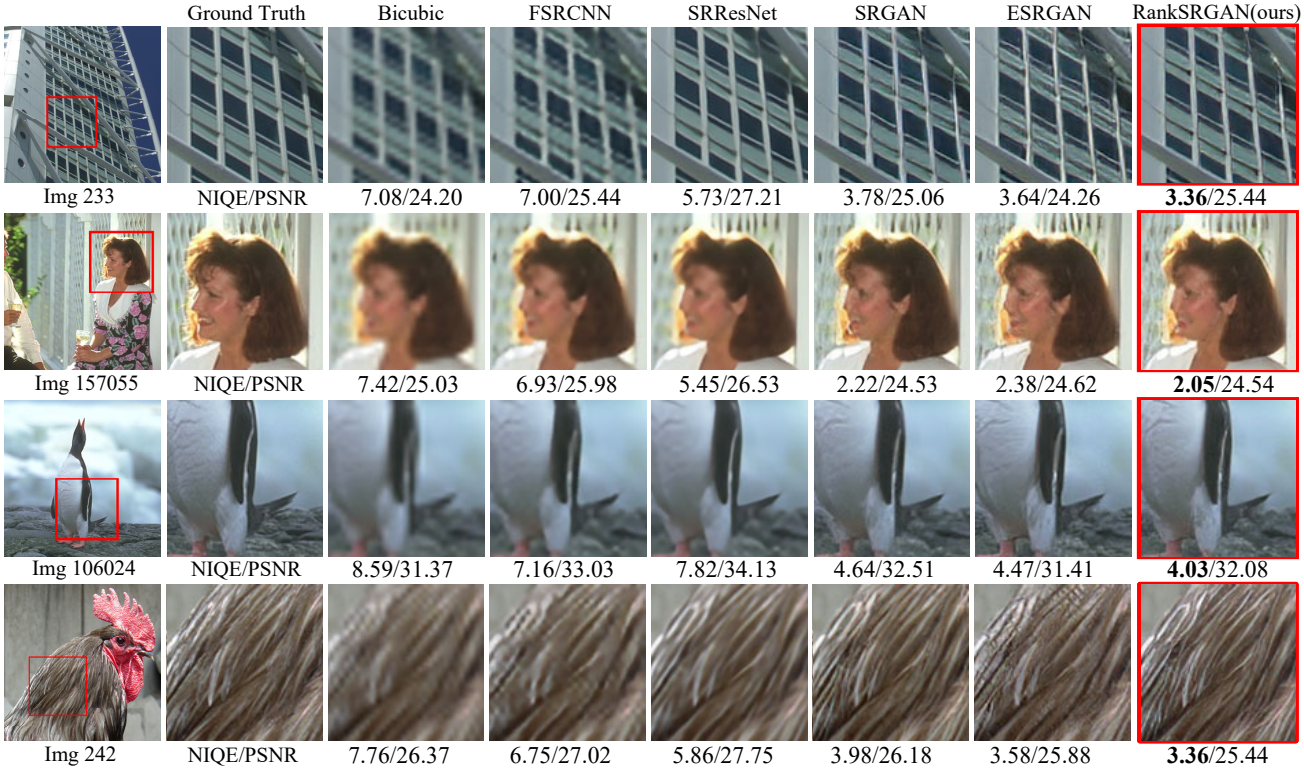


Figure 5. Visual results comparison of our model with other works on $\times 4$ super-resolution. Lower NIQE value indicates better perceptual quality, while higher PSNR indicates less distortion.

Adam[21] optimization method with $\beta_1 = 0.9$ is used for training. For generator and discriminator, the initial learning rate is set to 1×10^{-4} which is reduced by a half for multi-step $[50 \times 10^3, 100 \times 10^3, 200 \times 10^3, 300 \times 10^3]$. A total of 600×10^3 iterations are executed by PyTorch. In training procedure, we add Ranker to the standard SRGAN. The Ranker takes some time to predict the ranking score, thus the traning time is a little slower (about 1.18 times) than standard SRGAN[22]. For the generator, the number of parameters remains the same as SRGAN[22].

4.3. Comparison with the-state-of-the-arts

We compare the performance of the proposed method with the state-of-the-art perceptual SR methods ESRGAN [35]/ SRGAN [22] and the PSNR-orientated methods FSRCNN [11] and SRResNet [22] [1]. The evaluation metrics include NIQE [29], PI [2] and PSNR. Table 2 shows their performance on three test datasets – Set14, BSD100 and PIRM-Test. Note that lower NIQE/PI indicates better visual quality. When comparing our method with SRGAN and ESRGAN, we find that RankSRGAN achieves the best NIQE and PI performance on all test sets. Furthermore, the improvement of perceptual scores does not come at the price of PSNR. Note that in PIRM-Test, RankSRGAN also obtains the highest PSNR values among perceptual SR methods. Figure 5 shows some visual examples, where we observe that our method could generate more realistic textures without introducing additional artifacts (please see the win-

[1] Our implementation of SRResNet and SRGAN achieve even better performance than that reported in the original paper.

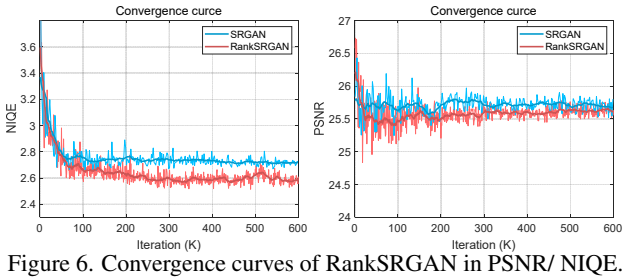


Figure 6. Convergence curves of RankSRGAN in PSNR/ NIQE.

dows in Img 233 and feathers in Img 242).

As the results may vary across different iterations, we further show the convergence curves of RankSRGAN in Figure 6. Their performance on NIQE and PSNR are relatively stable during the training process. For PSNR, they obtain comparable results. But for NIQE, RankSRGAN is consistently better than SRGAN by a large margin.

4.4. Ablation study

Effect of different rank datasets. The key factor that influences the performance of Ranker is the choice of SR algorithms. In the main experiments, we use (SRResNet, SRGAN, ESRGAN) to generate the rank dataset. Then what if we select other SR algorithms? Will we always obtain better results than SRGAN? To answer the question, we first analyze the reason of using these three algorithms, then conduct another experiment using a different combination.

As our baseline model is SRGAN, we need the Ranker to have the ability to rank the outputs of SRGAN. Since the training of SRGAN starts from the pre-trained model SRResNet, the Ranker should recognize the results between SRResNet and SRGAN. That is the reason why we choose SRResNet and SRGAN. Then the next step is to find a better algorithm that could guide the model to achieve better results. We choose ESRGAN as it surpasses SRGAN by a large margin in the PIRM-SR2018 challenge [2]. Therefore, we believe that a better algorithm than SRGAN could always lead to better performance.

Method	SRGAN	RankSRGAN	RankSRGAN-HR
NIQE	2.70	**2.51**	2.58
PSNR	25.62	25.60	**26.00**

Table 3. Comparison with RankSRGAN and RankSRGAN-HR.

To validate this comment, we directly use the ground truth HR as the third algorithm, which is the extreme case. We still apply NIQE for evaluation. Interestingly, although HR images have infinite PSNR values, they cannot surpass all the results of SRGAN on NIQE. Similar to ESRGAN, HR and SRGAN have mixed ranking orders. We train our Ranker with (SRResNet, SRGAN, HR) and obtain the new SR model – RankSRGAN-HR. Table 3 compares its results with SRGAN and RankSRGAN. As expected, RankSRGAN-HR achieves better NIQE values than SRGAN. But at the same time, RankSRGAN-HR also improves the PSNR by almost 0.4 dB. It achieves a good balance between the perceptual metric and PSNR. This also indicates that the model could always have an improvement space as long as we have better algorithms for guidance.

Effect of different perceptual metrics. As we claim that Ranker can guide the SR model to be optimized in the direction of perceptual metrics, we need to verify whether it works for other perceptual metrics. We choose Ma [27] and PI [2], which show high correlation with Mean-Opinion-Score (Ma: 0.61, PI: 0.83) in [2]. We use Ma and PI as the evaluation metric to generate the rank dataset. All other settings remain the same as RankSRGAN with NIQE. The only difference in these experiments is the ranking labels in the rank dataset. The results are summarized in Table 4, where we observe that the Ranker could help RankSRGAN achieve the best performance in the chosen metric. This shows that our method can generalize well on different perceptual metrics.

Method	NIQE	10-Ma	PI	PSNR
SRGAN	2.71	1.47	2.09	25.62
ESRGAN	2.56	1.40	1.98	25.30
RankSRGAN$_N$	**2.51**	1.39	1.95	25.62
RankSRGAN$_M$	2.65	**1.38**	2.01	25.21
RankSRGAN$_{PI}$	2.49	1.39	**1.94**	25.49

Table 4. The performance of RankSRGAN with different Rankers. N: Ranker with NIQE [29], M: Ranker with Ma [26] and PI: Ranker with PI [2].

Effect of Ranker: Rank VS. Regression. To train our Ranker, we choose to use the ranking orders instead of the real values of the perceptual metric. Actually, we can also let the network directly learn the real values. In [8], Choi et al. use a regression network to predict a subjective score for a given image and define a corresponding subjective score loss. To compare these two strategies, we train a "regression" Ranker with MSE loss instead of the margin-ranking loss. The labels in the rank dataset are real values of the perceptual metric. We use NIQE and Ma to generate the labels of rank dataset. All the other settings remain the same as RankSRGAN.

Metric	Method	$E(\|SR_1 - SR_2\|)$
NIQE	regression	0.06
NIQE	rank	**0.11**
Ma	regression	0.09
Ma	rank	**0.15**

Table 5. The distance between SR_1 and SR_2 with regression and rank.

Theoretically, the real values of perceptual metrics may distribute unevenly among different algorithms. For example, SRGAN and ESRGAN are very close to each other on NIQE values. This presents a difficulty for the learning of regression. On the contrary, learning ranking orders

can simply ignore these variances. In experiments, we first measure the distances between the outputs of SRGAN and ESRGAN with different strategies. Table 5 shows the mean absolute distances of these two strategies. Obviously, results with rank have larger distances than results with regression. When applying these Rankers in SR training, the rank strategy achieves better performance than the regression strategy on the selected perceptual metric. Results are shown in Table 6.

Method	NIQE	10-Ma	PSNR
SRGAN	2.71	1.47	25.62
ESRGAN	2.55	1.40	25.30
RankSRGAN-Re$_N$	2.53	1.42	25.58
RankSRGAN$_N$	**2.51**	1.39	25.60
RankSRGAN-Re$_M$	2.61	1.43	25.23
RankSRGAN$_M$	2.65	**1.38**	25.21

Table 6. The performance of RankSRGAN with different rankers. Re: Ranker with regression, N: Ranker with NIQE, M: Ranker with Ma.

Effect of different losses. To test the effects of rank-content loss, we expect to add MSE loss to achieve improvement in PSNR. Table 7 shows the performance of our method trained with the combination of loss functions.

Method	Loss	NIQE	PSNR
SRGAN	L_P	2.71	25.62
ESRGAN	$L_P + 10L_M$	2.55	25.30
RankSRGAN	$L_P + L_R$	**2.51**	25.62
RankSRGAN-M$_1$	$L_P + L_R + \alpha_1 L_M$	2.55	25.87
RankSRGAN-M$_2$	$L_P + L_R + \alpha_2 L_M$	2.72	**26.62**

Table 7. The performance of RankSRGAN with the combination of loss functions (P: perceptual loss, R: rank-content loss, M: MSE loss). $\alpha_1, \alpha_2 : \{1, 5\}$

As expected, increasing the contribution of the MSE loss with the larger α results in higher PSNR values. On the other hand, the NIQE values are increased which is a trade-off between PSNR and NIQE as mentioned in [3], and our method has a capability to deal with the priorities by adjusting the weights of the loss functions.

4.5. User study

To demonstrate the effectiveness and superiority of RankSRGAN, we conduct a user study against state-of-the-art models, i.e. SRGAN [22] and ESRGAN [35]. In the first session, two different SR images are shown at the same time where one is generated by the proposed RankSRGAN and the other is generated by SRGAN or ESRGAN. The participants are required to pick the image that is more visually pleasant (more natural and realistic). We use the PIRM-Test [2] dataset as the testing dataset. There are a total of 100 images, from which 30 images are randomly selected for each participant. To make a better comparison, one small patch from the image is zoomed in. In the second session,

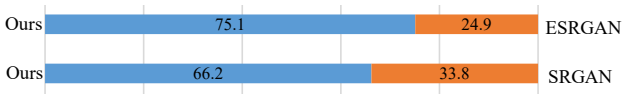


Figure 7. The results of user studies, comparing our method with SRGAN [22] and ESRGAN [35].

we focus on the perceptual quality of different typical SR methods in a sorting manner. The participants are asked to rank 4 versions of each image: SRResNet [22], ESRGAN [35], RankSRGAN, and the Ground Truth (GT) image according to their visual qualities. Similar to the first session, 20 images are randomly shown for each participant. There are totally 30 participants to finish the user study.

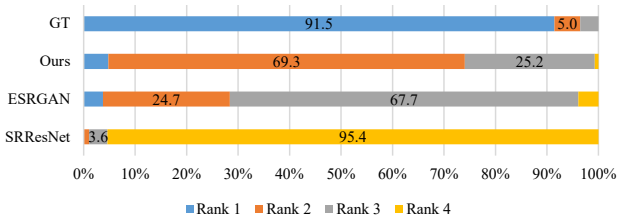


Figure 8. The ranking results of user studies: SRResNet [22], ESRGAN [35], RankSRGAN (ours), and the original HR image.

As suggested in Figure 7, RankSRGAN has achieved better visual performance against ESRGAN and SRGAN. Since RankSRGAN consists of a base model SRGAN and the proposed Ranker, it can naturally inherit the characteristics of SRGAN and achieve better performance in perceptual metric. Thus, RankSRGAN performs more similar to SRGAN than ESRGAN. Figure 8 shows the ranking results of different SR methods. As RankSRGAN has the best performance in perceptual metric, the ranking results of RankSRGAN are second to GT images, but sometimes it even produces images comparable to GT.

5. Conclusion

For perceptual super-resolution, we propose RankSRGAN to optimize SR model in the orientation of perceptual metrics. The key idea is introducing a Ranker to learn the behavior of the perceptual metrics by learning to rank approach. Moreover, our proposed method can combine the strengths of different SR methods and generate better results. Extensive experiments well demonstrate that our RankSRGAN is a flexible framework, which can achieve superiority over state-of-the-art methods in perceptual metric and have the ability to recover more realistic textures.

Acknowledgements. This work is partially supported by National Natural Science Foundation of China (61876176, U1613211), Shenzhen Basic Research Program (JCYJ20170818164704758), the Joint Lab of CAS-HK.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, volume 3, page 2, 2017.

[2] Yochai Blau, Roey Mechrez, Radu Timofte, Tomer Michaeli, and Lihi Zelnik-Manor. 2018 pirm challenge on perceptual image super-resolution. *arXiv preprint arXiv:1809.07517*, 2018.

[3] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *Proc. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, Utah, USA*, pages 6228–6237, 2018.

[4] Sebastian Bosse, Dominique Maniry, Klaus-Robert Müller, Thomas Wiegand, and Wojciech Samek. Deep neural networks for no-reference and full-reference image quality assessment. *IEEE Transactions on Image Processing*, 27(1):206–219, 2018.

[5] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744, 1994.

[6] Joan Bruna, Pablo Sprechmann, and Yann LeCun. Super-resolution with deep convolutional sufficient statistics. *arXiv preprint arXiv:1511.05666*, 2015.

[7] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96. ACM, 2005.

[8] Jun-Ho Choi, Jun-Hyuk Kim, Manri Cheon, and Jong-Seok Lee. Deep learning-based image super-resolution considering quantitative and perceptual quality. *arXiv preprint arXiv:1809.04789*, 2018.

[9] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 539–546. IEEE, 2005.

[10] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*, pages 184–199. Springer, 2014.

[11] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *European Conference on Computer Vision*, pages 391–407. Springer, 2016.

[12] Alexey Dosovitskiy and Thomas Brox. Generating images with perceptual similarity metrics based on deep networks. In *Advances in Neural Information Processing Systems*, pages 658–666, 2016.

[13] Ruicheng Feng, Jinjin Gu, Yu Qiao, and Chao Dong. Suppressing model overfitting for image super-resolution networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019.

[14] Jinjin Gu, Hannan Lu, Wangmeng Zuo, and Chao Dong. Blind super-resolution with iterative kernel correction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1604–1613, 2019.

[15] Muhammad Haris, Greg Shakhnarovich, and Norimichi Ukita. Deep backprojection networks for super-resolution. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[16] Jingwen He, Chao Dong, and Yu Qiao. Modulating image restoration with continual levels via adaptive feature modification layers. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[18] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016.

[19] Le Kang, Peng Ye, Yi Li, and David Doermann. Convolutional neural networks for no-reference image quality assessment. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1733–1740, 2014.

[20] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016.

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[22] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew P Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, volume 2, page 4, 2017.

[23] Chaofeng Li, Alan Conrad Bovik, and Xiaojun Wu. Blind image quality assessment using a general regression neural network. *IEEE Transactions on Neural Networks*, 22(5):793–799, 2011.

[24] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *The IEEE conference on computer vision and pattern recognition (CVPR) workshops*, volume 1, page 4, 2017.

[25] Xialei Liu, Joost van de Weijer, and Andrew D Bagdanov. Rankiqa: Learning from rankings for no-reference image quality assessment. *Computer Vision and Pattern Recognition, https://arxiv. org/abs/1707.08347 v1*, 2017.

[26] Chao Ma, Chih-Yuan Yang, Xiaokang Yang, and Ming-Hsuan Yang. Learning a no-reference quality metric for single-image super-resolution. *Computer Vision and Image Understanding*, 158:1–16, 2017.

[27] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 416–423. IEEE, 2001.

[28] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[29] Anish Mittal, Rajiv Soundararajan, and Alan C Bovik. Making a" completely blind" image quality analyzer. *IEEE Signal Process. Lett.*, 20(3):209–212, 2013.

[30] Devi Parikh and Kristen Grauman. Relative attributes. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 503–510. IEEE, 2011.

[31] Mehdi SM Sajjadi, Bernhard Schölkopf, and Michael Hirsch. Enhancenet: Single image super-resolution through automated texture synthesis. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 4501–4510. IEEE, 2017.

[32] Yassir Saquil, Kwang In Kim, and Peter Hall. Ranking cgans: Subjective control over semantic image attributes. In *Proc. of British Machine Vision Conference (BMVC)*, 2018.

[33] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[34] Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Recovering realistic texture in image super-resolution by deep spatial feature transform. *arXiv preprint arXiv:1804.02815*, 2018.

[35] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *The European Conference on Computer Vision (ECCV) Workshops*, September 2018.

[36] Xiaoshan Yang, Tianzhu Zhang, Changsheng Xu, Shuicheng Yan, M Shamim Hossain, and Ahmed Ghoneim. Deep relative attributes. *IEEE Transactions on Multimedia*, 18(9):1832–1842, 2016.

[37] Peng Ye and David S Doermann. No-reference image quality assessment based on visual codebook. In *ICIP*, pages 3089–3092. Citeseer, 2011.

[38] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4353–4361, 2015.

[39] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *International conference on curves and surfaces*, pages 711–730. Springer, 2010.

[40] Kaibing Zhang, Xinbo Gao, Dacheng Tao, Xuelong Li, et al. Single image super-resolution with non-local means and steering kernel regression. *Image*, 11:12, 2012.

[41] Lei Zhang and Xiaolin Wu. An edge-guided image interpolation algorithm via directional filtering and data fusion. *IEEE transactions on Image Processing*, 15(8):2226–2238, 2006.

[42] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

RankSRGAN: Generative Adversarial Networks with Ranker for Image Super-Resolution Supplementary File

Wenlong Zhang[1] Yihao Liu[1,2] Chao Dong[1,†] Yu Qiao[1]
[1]ShenZhen Key Lab of Computer Vision and Pattern Recognition, SIAT-SenseTime Joint Lab, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
[2]University of Chinese Academy of Sciences
{wl.zhang1, yh.liu4, chao.dong, yu.qiao}@siat.ac.cn

Abstract

In this supplementary file, we first present more details and additional experimental results of our proposed Ranker. Then, we provide the curves showing the performance of different RankSRGAN models in the ablation study. Finally, we provide more additional qualitative results to compare our networks with the state-of-the-art methods.

1. Details of Ranker

1.1. Dataset

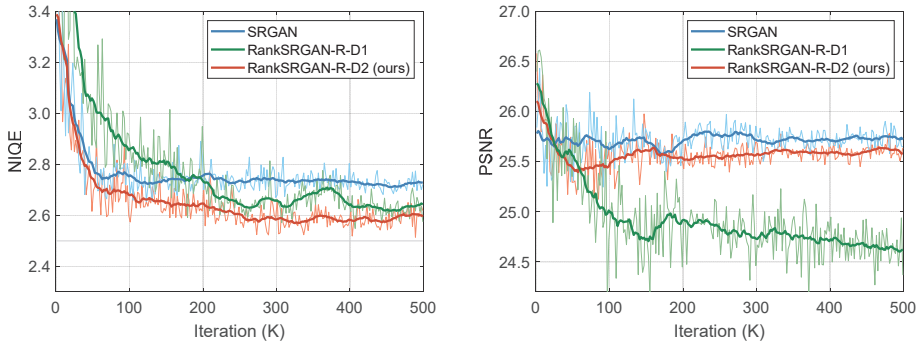


Figure 1. The convergence curves of RankSRGAN with Ranker1 and Ranker2 in NIQE and PSNR.

Method	Ranker	Dataset	Data Size (k)	SROCC	NIQE	PSNR
RankSRGAN-R-D1	Ranker1	DIV2K	15	0.78	2.53	24.54
RankSRGAN-R-D2	Ranker2	DIV2K+Flickr2K	150	0.88	2.51	25.62

Table 1. The performance of RankSRGAN with Ranker1 and Ranker2. R-D1: rank dataset1 (15 K), R-D2: rank dataset2 (150 K)

To analyze the effects of Ranker on RankSRGAN, we use SRResNet [3], SRGAN [3] and ESRGAN [9] to generate two rank datasets with different sizes. We first employ DIV2K [1] to generate rank dataset1 with **15 K** image pairs. Besides, we use DIV2K+Flickr2k [1] to generate rank dataset2 with **150 K** image pairs. Then, we utilize rank dataset1 and rank dataset2 to train Ranker1 and Ranker2, respectively. Finally, the well-trained Ranker1 and Ranker2 are applied on RankSRGAN. Table 1 shows that more data leads to better SROCC, and the Ranker2 with higher SROCC can reach better performance in NIQE and PSNR. The convergence curves are shown in Figure 1.

†Corresponding author (e-mail: chao.dong@siat.ac.cn)

1.2. Network Architecture

The architecture of Ranker is based on the VGG network [8]. We train three VGG networks varying from shallow to deep ones: VGG-8, VGG-12 and VGG-16. Table 2 shows the architecture, the number of parameters, and the performance in different models. Since the VGG-12 can achieve the same accuracy as VGG-16, we apply the VGG-8 and VGG-12 on RankSRGAN. Figure 2 shows the performance of RankSRGAN with different Rankers. The Ranker with higher value of SROCC can achieve better performance when applied on RankSRGAN.

Model	VGG-8	VGG-12 (Ours)	VGG-16
Architecture	Conv3S1-64 Conv4S2-64, BN, LReLU	Conv3S1-64 Conv4S2-64, BN, LReLU	Conv3S1-64, LReLU Conv4S2-64, BN, LReLU
			Conv3S1-128, BN, LReLU
		Conv3S1-128, BN, LReLU	Conv3S1-128, BN, LReLU
	Conv4S2-128, BN, LReLU	Conv4S2-128, BN, LReLU	Conv4S2-128, BN, LReLU
			Conv3S1-256, BN, LReLU
		Conv3S1-256, BN, LReLU	Conv3S1-256, BN, LReLU
	Conv4S2-256, BN, LReLU	Conv4S2-256, BN, LReLU	Conv4S2-256, BN, LReLU
			Conv3S1-512, BN, LReLU
		Conv3S1-512, BN, LReLU	Conv3S1-512, BN, LReLU
	Conv4S2-512, BN, LReLU	Conv4S2-512, BN, LReLU	Conv4S2-512, BN, LReLU
			Conv3S1-512, BN, LReLU
		Conv3S1-512, BN, LReLU	Conv3S1-512, BN, LReLU
	Conv4S2-512, BN, LReLU	Conv4S2-512, BN, LReLU	Conv4S2-512, BN, LReLU
	Average pooling		
	FC-100		
	FC-1		
Number of params (K)	7,069	13,734	19,194
SROCC	0.83	0.88	0.88

Table 2. The network architecture of Rankers with different depths. The network design draws inspiration from VGG [8] but uses Leaky ReLU activations [5] and strided convolutions instead of pooling layers [7]. Conv3S1-64: Convolutional layer with kernel size 3×3, stride 1 and channel 64. BN: Batch Normalization. LReLU: Leaky ReLU.

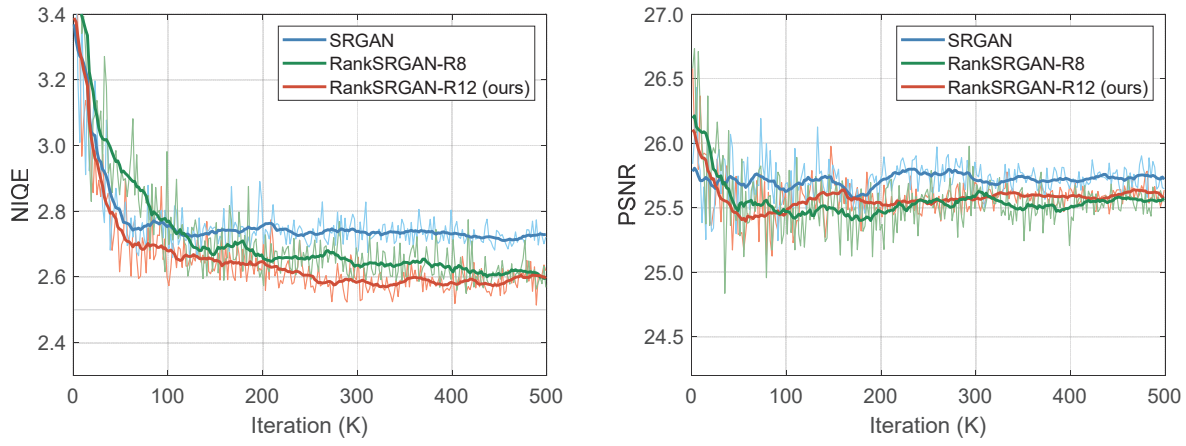


Figure 2. The convergence curves of RankSRGAN with Ranker-VGG-8 and Ranker-VGG-12 in NIQE and PSNR.

1.2. Network Architecture

The architecture of Ranker is based on the VGG network [8]. We train three VGG networks varying from shallow to deep ones: VGG-8, VGG-12 and VGG-16. Table 2 shows the architecture, the number of parameters, and the performance in different models. Since the VGG-12 can achieve the same accuracy as VGG-16, we apply the VGG-8 and VGG-12 on RankSRGAN. Figure 2 shows the performance of RankSRGAN with different Rankers. The Ranker with higher value of SROCC can achieve better performance when applied on RankSRGAN.

Model	VGG-8	VGG-12 (Ours)	VGG-16
Architecture	Conv3S1-64 Conv4S2-64, BN, LReLU	Conv3S1-64 Conv4S2-64, BN, LReLU	Conv3S1-64, LReLU Conv4S2-64, BN, LReLU
			Conv3S1-128, BN, LReLU
		Conv3S1-128, BN, LReLU	Conv3S1-128, BN, LReLU
	Conv4S2-128, BN, LReLU	Conv4S2-128, BN, LReLU	Conv4S2-128, BN, LReLU
			Conv3S1-256, BN, LReLU
		Conv3S1-256, BN, LReLU	Conv3S1-256, BN, LReLU
	Conv4S2-256, BN, LReLU	Conv4S2-256, BN, LReLU	Conv4S2-256, BN, LReLU
			Conv3S1-512, BN, LReLU
		Conv3S1-512, BN, LReLU	Conv3S1-512, BN, LReLU
	Conv4S2-512, BN, LReLU	Conv4S2-512, BN, LReLU	Conv4S2-512, BN, LReLU
			Conv3S1-512, BN, LReLU
		Conv3S1-512, BN, LReLU	Conv3S1-512, BN, LReLU
	Conv4S2-512, BN, LReLU	Conv4S2-512, BN, LReLU	Conv4S2-512, BN, LReLU
	Average pooling		
	FC-100		
	FC-1		
Number of params (K)	7,069	13,734	19,194
SROCC	0.83	0.88	0.88

Table 2. The network architecture of Rankers with different depths. The network design draws inspiration from VGG [8] but uses Leaky ReLU activations [5] and strided convolutions instead of pooling layers [7]. Conv3S1-64: Convolutional layer with kernel size 3×3, stride 1 and channel 64. BN: Batch Normalization. LReLU: Leaky ReLU.

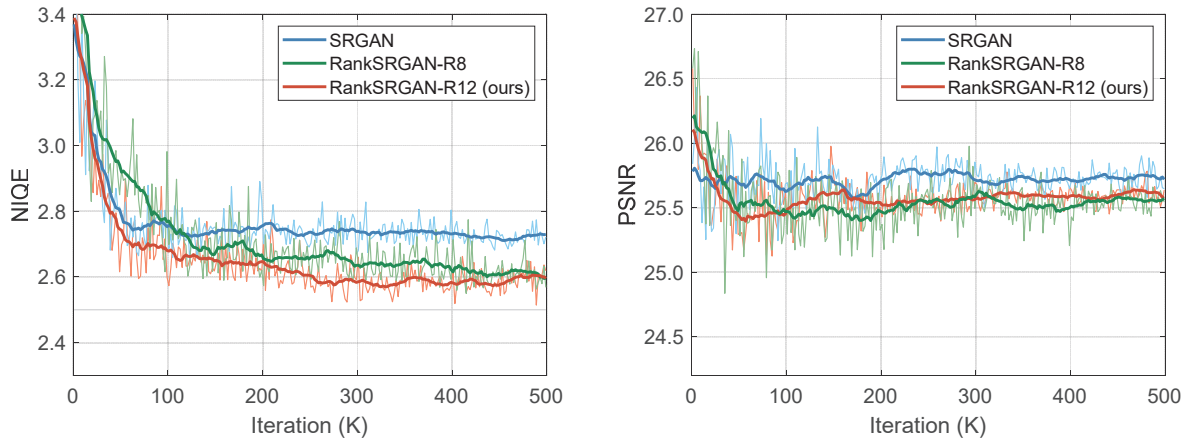


Figure 2. The convergence curves of RankSRGAN with Ranker-VGG-8 and Ranker-VGG-12 in NIQE and PSNR.

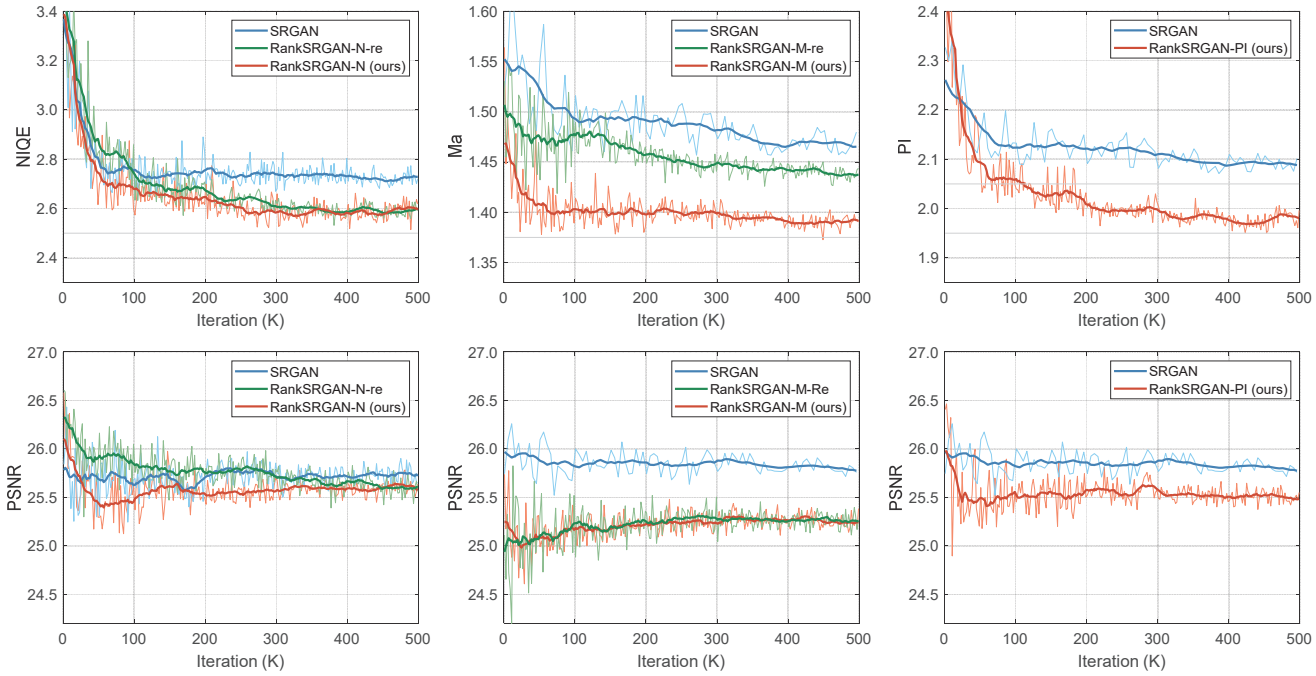


Figure 5. The convergence curves of RankSRGAN$_N$, RankSRGAN$_M$ and RankSRGAN$_{PI}$.

2.2. Convergence curves for RankSRGAN-HR

As shown in Figure 6, we present the curves of RankSRGAN and RankSRGAN-HR. To improve the performance of NIQE evaluation, we use (SRResNet, SRGAN, ESRGAN) to generate rank dataset to train Ranker in RankSRGAN. Figure 6 shows that RankSRGAN is consistently better than SRGAN by a large margin. Furthermore, we directly use the ground truth HR to replace ESRGAN . We train our Ranker with the rank dataset (SRResNet, SRGAN, HR) and obtain the new model RankSRGAN-HR. In Figure 6, RankSRGAN-HR achieves better NIQE values than SRGAN. But at the same time, RankSRGAN-HR also constantly improves the PSNR. It achieves a good balance between the perceptual metric and PSNR.

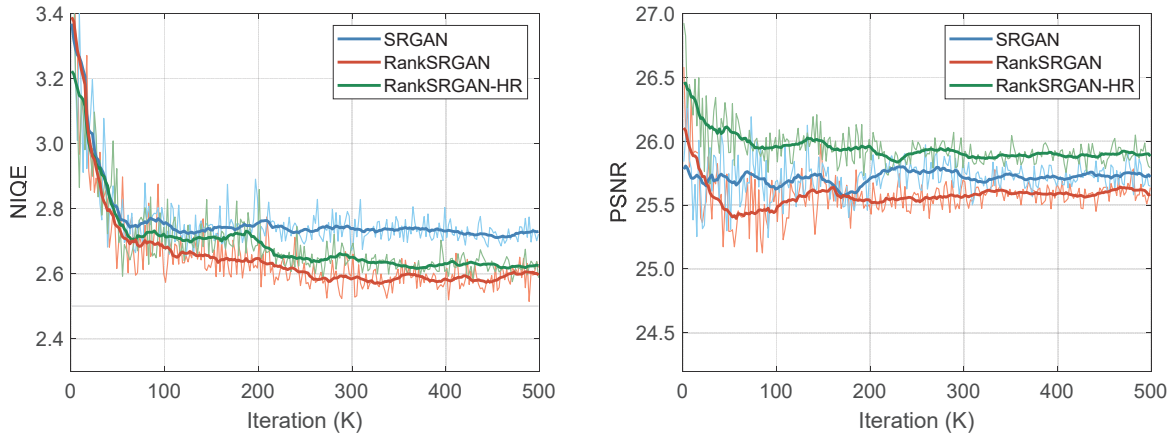


Figure 6. The convergence curves of RankSRGAN-HR in PSNR and NIQE.

3. More Qualitative Results

In this section, we provide additional qualitative results ($\times 4$ enlargement) to clearly show the effectiveness of our RankSRGAN. We compare the proposed RankSRGAN with the state-of-the-art perceptual SR methods SRGAN [3] / ESRGAN [9] and PSNR-oriented method SRResNet [3]. We employ NIQE and PSNR to evaluate those SR methods. Lower NIQE value indicates better perceptual quality while higher PSNR indicates that there is less distortion with the Ground-Truth image.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, volume 3, page 2, 2017.

[2] Yochai Blau, Roey Mechrez, Radu Timofte, Tomer Michaeli, and Lihi Zelnik-Manor. 2018 pirm challenge on perceptual image super-resolution. *arXiv preprint arXiv:1809.07517*, 2018.

[3] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew P Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, volume 2, page 4, 2017.

[4] Chao Ma, Chih-Yuan Yang, Xiaokang Yang, and Ming-Hsuan Yang. Learning a no-reference quality metric for single-image super-resolution. *Computer Vision and Image Understanding*, 158:1–16, 2017.

[5] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3, 2013.

[6] Anish Mittal, Rajiv Soundararajan, and Alan C Bovik. Making a" completely blind" image quality analyzer. *IEEE Signal Process. Lett.*, 20(3):209–212, 2013.

[7] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *Computer Science*, 2015.

[8] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

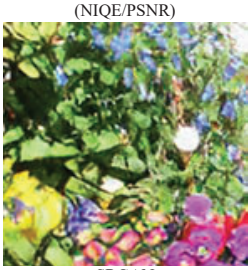
[9] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *The European Conference on Computer Vision (ECCV) Workshops*, September 2018.

Img 206

GT
(NIQE/PSNR)

Bicubic
(7.49/20.94)

SRResNet
(5.30/21.98)



SRGAN
(2.33/19.20)

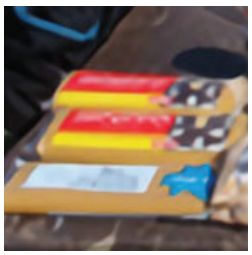
ESRGAN
(2.51/19.73)

RankSRGAN (Ours)
(**2.17**/19.34)

Img 207

GT
(NIQE/PSNR)

Bicubic
(7.22/26.84)



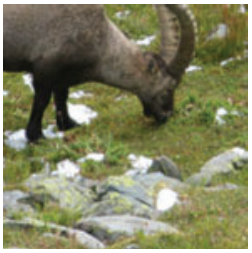
SRResNet
(5.12/28.71)

SRGAN
(1.99/26.13)

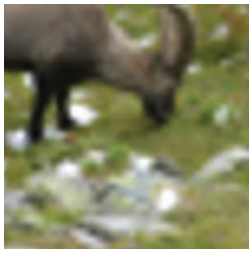
ESRGAN
(1.77/25.28)

RankSRGAN (Ours)
(**1.77**/26.07)

Img 210

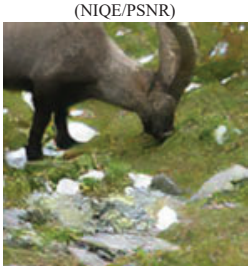


GT
(NIQE/PSNR)



Bicubic
(7.79/25.38)

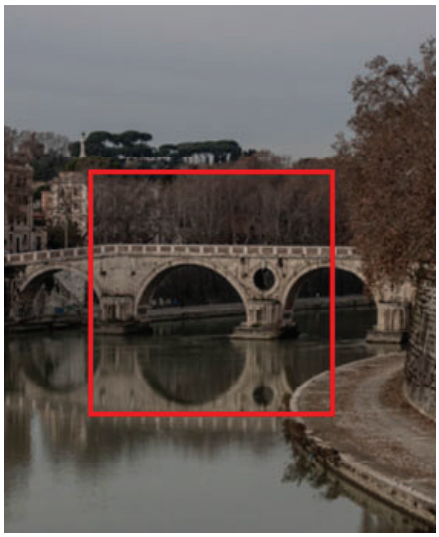
SRResNet
(6.69/26.38)



SRGAN
(2.70/23.49)

ESRGAN
(2.82/23.12)

RankSRGAN (Ours)
(**2.66**/23.42)

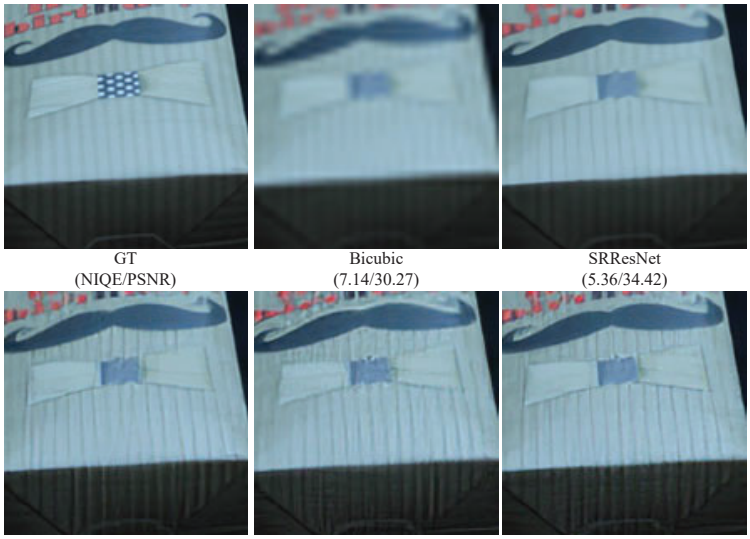


Img 216

GT (NIQE/PSNR) | Bicubic (7.33/29.25) | SRResNet (5.77/30.27)

SRGAN (2.31/27.52) | ESRGAN (2.53/27.98) | RankSRGAN (Ours) (**2.22**/27.43)

Img 225



GT (NIQE/PSNR) | Bicubic (7.14/30.27) | SRResNet (5.36/34.42)

SRGAN (2.75/31.88) | ESRGAN (2.47/31.14) | RankSRGAN (Ours) (**2.27**/31.48)

Img 238

GT (NIQE/PSNR) | Bicubic (7.82/26.51) | SRResNet (6.07/31.00)

SRGAN (3.66/29.40) | ESRGAN (3.49/28.26) | RankSRGAN (Ours) (**3.14**/29.10)

Img 243

GT (NIQE/PSNR) | Bicubic (7.45/26.37) | SRResNet (5.12/29.83)

SRGAN (2.42/27.38) | ESRGAN (2.32/26.71) | RankSRGAN (Ours) (**2.11**/26.96)

Img 246

GT (NIQE/PSNR) | Bicubic (7.02/30.60) | SRResNet (6.11/32.64)

SRGAN (2.73/30.32) | ESRGAN (2.91/30.16) | RankSRGAN (Ours) (**2.47**/30.29)

Img 256

GT (NIQE/PSNR) | Bicubic (7.15/26.83) | SRResNet (6.55/27.69)

SRGAN (2.88/24.18) | ESRGAN (2.68/24.75) | RankSRGAN (Ours) (**2.44**/24.85)