

Sapienza University of Rome

Vision and Perception

Deep Learning Project

ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks

Problem Statement

The Super-Resolution Generative Adversarial is capable of generating realistic textures during single image super-resolution. However, the **hallucinated details** are often accompanied with **unpleasant artifacts**.

In this paper the **Enhanced SRGAN (ESRGAN)** was introduced to further **enhance** the visual quality by **studying** and **improving** three key components of SRGAN – **network architecture**, **adversarial loss** and **perceptual loss**.

In particular, the novelties include :

- **Residual-in-Residual Dense Block (RRDB)** without batch normalization as the basic network building unit.
- **Relativistic GAN** which consists on a discriminator that predicts relative realness instead of the absolute value.
- Finally, the **perceptual loss** is improved by using the features before activation, which could provide stronger supervision for brightness consistency and texture recovery.

Benefiting from these improvements, the proposed ESRGAN achieves consistently **better visual quality** with more realistic and natural textures than SRGAN.

Short state of the art

To solve the Super Resolution problems, we mainly focus on deep neural network approaches.

- In the beginning we had SRCNN architectures to learn the mapping from LR to HR images in an end-to-end manner.
- Then **deeper network with residual learning, Laplacian pyramid structure, residual blocks, recursive learning, densely connected network, deep back projection and residual dense network** were introduced.
- Several methods were proposed to **stabilize** training a very deep model. For instance, **residual path** is developed to stabilize the training and improve the performance.
- Then a robust initialization method for VGG-style networks without BN was introduced. To facilitate training a deeper network, a compact and effective residual-in-residual dense block was developed, which also helps to improve the perceptual quality.
- **Perceptual-driven approaches** have also been proposed to improve the visual quality of SR results.
- **Perceptual loss** was proposed to **enhance the visual quality** by minimizing the error in a feature space instead of pixel space.
- **Contextual loss** was developed to **generate images with natural image statistics** by using an objective that focuses on the feature distribution rather than merely comparing the appearance.
- **Relativistic discriminator** was developed not only **to increase the probability that generated data are real**, but also **to simultaneously decrease the probability that real data are real**.
- SR algorithms are typically evaluated by several widely used distortion measures, e.g., **PSNR** (Peak signal-to-noise ratio) and **SSIM** (structural similarity index measure). However, these metrics fundamentally disagree with the subjective evaluation of human observers.

The proposed method

- We will first describe our proposed **network architecture** and then discuss the improvements from the **discriminator** and **perceptual loss**.
- In the end, we will describe the **network interpolation** and **Image interpolation** strategies for balancing perceptual quality and PSNR.

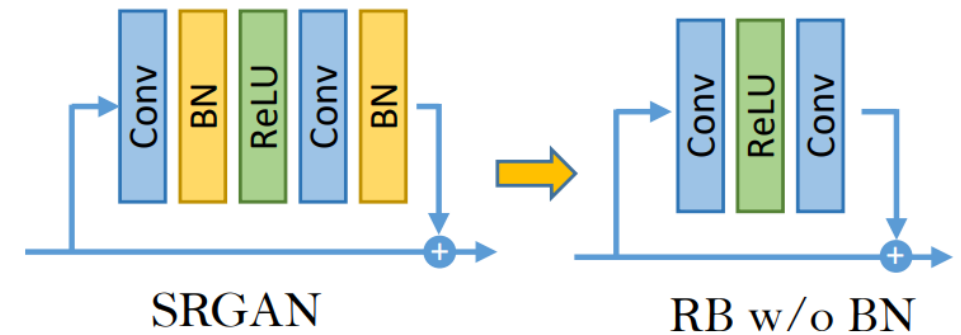
Network Architecture

- In order to further improve the recovered image quality of SRGAN, we mainly make two modifications to the structure of generator **G**:
 - 1) Remove all BN layers
 - 2) Replace the original **basic block** with the proposed **Residual-in-Residual Dense Block (RRDB)**, which combines multi-level residual network and dense connections.

BN removal

- Removing BN layers has proven to increase performance and reduce computational complexity.
- BN layers are more likely to bring artifacts when the network is deeper and trained under a GAN framework. These artifacts occasionally appear among iterations and different settings, violating the needs for a stable performance over training.
- We therefore remove BN layers for stable training and consistent performance. Furthermore, removing BN layers helps to **improve generalization ability, reduce computational complexity** and **memory usage**.

Residual Block (RB)



Residual-in-Residual Dense Block

- We keep the high-level architecture design of SRGAN and use a **novel** basic block named **RRDB** as in Fig. 4. The proposed RRDB employs a deeper and more complex structure than the original residual block in SRGAN and this lead to a **boost** in the performance.
- The proposed RRDB has a **residual-in-residual** structure, where **residual learning** is used in different levels.
- Our RRDB uses **dense block** in the **main path** where the network capacity becomes higher benefiting from the dense connections.

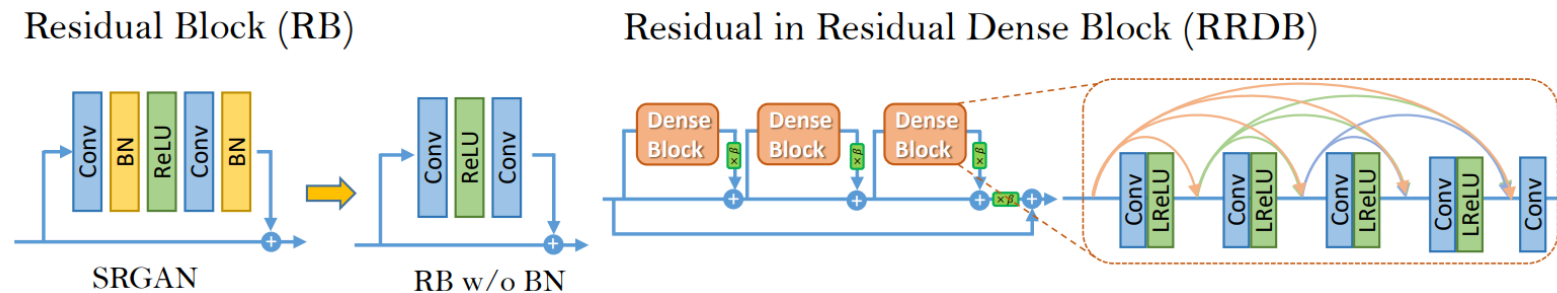


Fig. 4: **Left:** We remove the BN layers in residual block in SRGAN. **Right:** RRDB block is used in our deeper model and β is the residual scaling parameter.


More techniques used

- In addition to the **improved architecture**, several **techniques** are also exploited in order to facilitate the training of a very deep network:
 1. **Residual scaling** which scales down the residuals by multiplying a constant between 0 and 1 before adding them to the main path to prevent instability.
 2. **Smaller initialization**, as we empirically find residual architecture is easier to train when the initial parameter variance becomes smaller.

Relativistic Discriminator (1)

The discriminator based on the Relativistic GAN is enhanced. A **Relativistic Discriminator** tries to predict the probability that a real image x_r is relatively more realistic than a fake one x_f , as shown in Fig. 5.

- Specifically, we replace the standard discriminator with the Relativistic average Discriminator RaD, denoted as DRa. The standard discriminator in SRGAN can be expressed as $D(x) = \sigma(C(x))$, where σ is the sigmoid function and $C(x)$ is the non-transformed discriminator output.

$D(x_r) = \sigma(C(\text{Real})) \rightarrow 1$	Real?		$D_{Ra}(x_r, x_f) = \sigma(C(\text{Real}) - \mathbb{E}[C(\text{Fake})]) \rightarrow 1$	More realistic than fake data?
$D(x_f) = \sigma(C(\text{Fake})) \rightarrow 0$	Fake?		$D_{Ra}(x_f, x_r) = \sigma(C(\text{Fake}) - \mathbb{E}[C(\text{Real})]) \rightarrow 0$	Less realistic than real data?
a) Standard GAN			b) Relativistic GAN	

Relativistic Discriminator (2)

- Then the **RaD** is formulated as $\mathbf{D}_{\text{Ra}}(\mathbf{x}_r, \mathbf{x}_f) = \sigma(\mathbf{C}(\mathbf{x}_r) - \mathbf{E}_{\mathbf{x}_f} [\mathbf{C}(\mathbf{x}_f)])$, where $\mathbf{E}_{\mathbf{x}_f} [\cdot]$ represents the operation of taking average for all fake data in the mini-batch.
- The **discriminator loss** is then defined as:

$$L_D^{\text{Ra}} = -\mathbf{E}_{\mathbf{x}_r} [\log(\mathbf{D}_{\text{Ra}}(\mathbf{x}_r, \mathbf{x}_f))] - \mathbf{E}_{\mathbf{x}_f} [\log(1 - \mathbf{D}_{\text{Ra}}(\mathbf{x}_f, \mathbf{x}_r))].$$

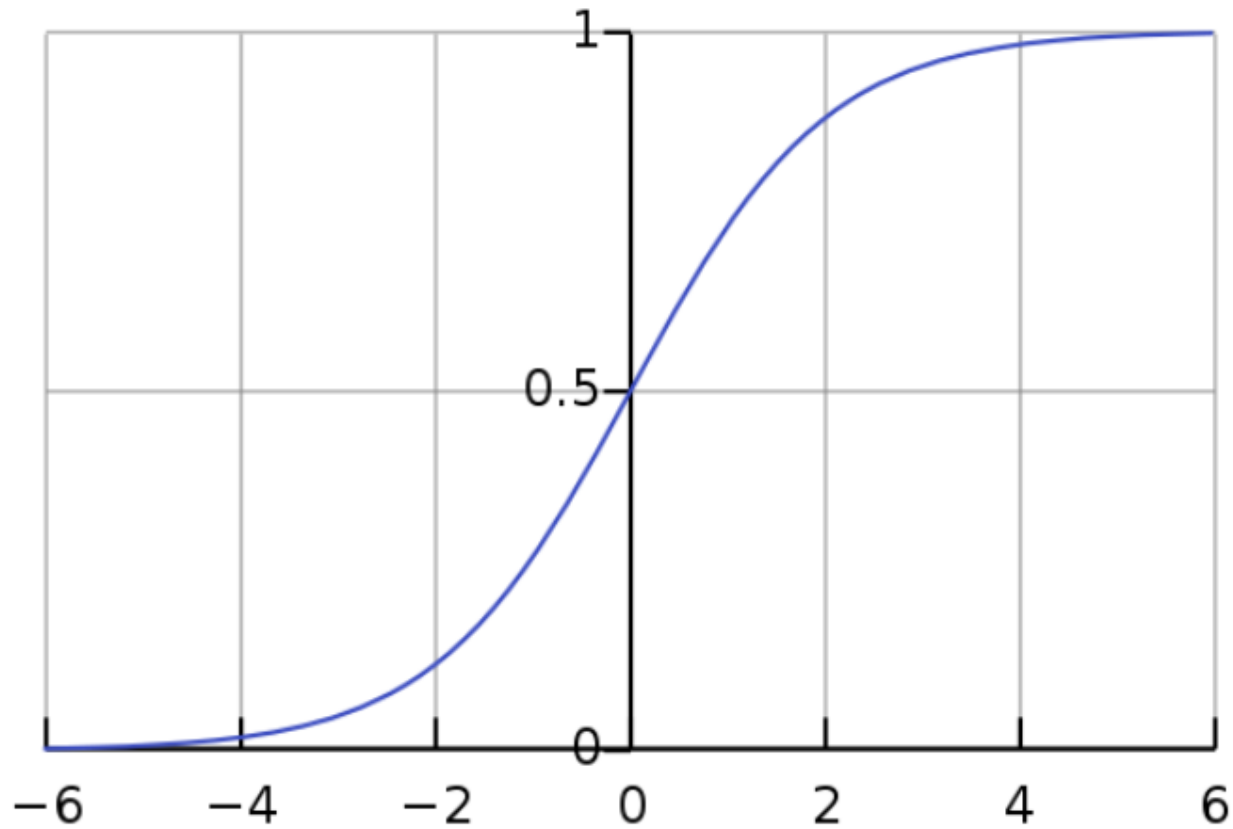
- The **adversarial loss** for generator is in a symmetrical form:

$$L_D^{\text{Ra}} = -\mathbf{E}_{\mathbf{x}_r} [\log(1 - \mathbf{D}_{\text{Ra}}(\mathbf{x}_r, \mathbf{x}_f))] - \mathbf{E}_{\mathbf{x}_f} [\log(\mathbf{D}_{\text{Ra}}(\mathbf{x}_f, \mathbf{x}_r))].$$

- Where $\mathbf{x}_f = G(\mathbf{x}_i)$ and \mathbf{x}_i stands for the input LR image. It is observed that the adversarial loss for generator contains both \mathbf{x}_r and \mathbf{x}_f . Therefore, our generator benefits from the gradients from both generated data and real data in adversarial training, while in SRGAN only generated part takes effect.

Functions used

Sigmoid activation function



Negative of a log function

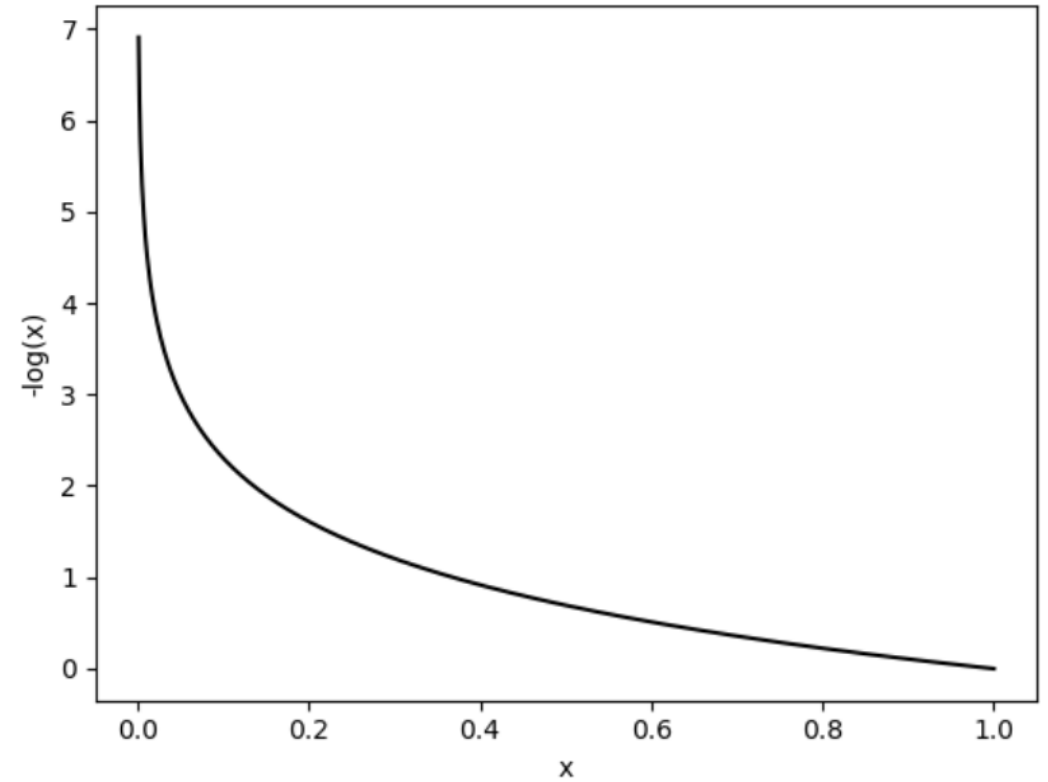


Figure: The loss function reaches infinity when input is 0, and reaches 0 when input is 1.

Perceptual Loss (1)

- We also develop a more effective perceptual loss $\mathbf{L}_{\text{percep}}$ by constraining on features before activation rather than after activation as practiced in SRGAN.
- Perceptual loss is previously defined on the activation layers of a pre-trained deep network, where the distance between two activated features is minimized.
- We propose to use features before the activation layers, which will overcome two drawbacks of the original design:
 1. **First**, the activated features are very sparse, especially after a very deep network. The sparse activation provides weak supervision and thus leads to inferior performance.
 2. **Second**, using features after activation also causes inconsistent reconstructed brightness compared with the ground-truth image.

Perceptual Loss (2)

Therefore, the total loss for the generator is:

$$\mathbf{L}_G = \mathbf{L}_{\text{percep}} + \lambda \mathbf{L}_G^{Ra} + \eta \mathbf{L}_1 \quad \text{Eq. (3)}$$

where $\mathbf{L}_1 = \mathbf{E}_{\mathbf{x}_i} \|\mathbf{G}(\mathbf{x}_i) - \mathbf{y}\|_1$ is the content loss (absolute loss) that evaluate the 1-norm distance between recovered image $\mathbf{G}(\mathbf{x}_i)$ and the ground-truth \mathbf{y} , and λ, η are the coefficients to balance different loss terms

In contrast to the commonly used perceptual loss that adopts a VGG network trained for image classification, we develop a more suitable perceptual loss for SR – MINC loss.

It is based on a fine-tuned VGG network for material recognition, which focuses on textures rather than object. Although the gain of perceptual index brought by MINC loss is marginal, we still believe that exploring perceptual loss that focuses on texture is critical for SR.

Network Interpolation (1)

- To remove unpleasant noise in GAN-based methods while maintain a good perceptual quality, we propose a flexible and effective strategy – **network interpolation**.
- Specifically, we **first train** a **PSNR-oriented network GPSNR** and **then** obtain a **GAN-based network GGAN** by fine-tuning. We interpolate all the corresponding parameters of these two networks to derive an interpolated model GINTERP, whose parameters are:

$\theta_G^{\text{INTERP}} = (1 - \alpha)\theta_G^{\text{PSNR}} + \alpha \theta_G^{\text{GAN}}$, where θ_G^{INTERP} , θ_G^{PSNR} and θ_G^{GAN} are the parameters of G_{INTERP} , G_{PSNR} and G_{GAN} , respectively, and $\alpha \in [0, 1]$ is the **interpolation parameter**.

- The proposed network interpolation brings two advantages:
 1. **First**, the interpolated model is able to produce meaningful results for any feasible α without introducing artifacts.
 2. **Second**, we can continuously balance perceptual quality and fidelity without re-training the model.

Network Interpolation (2)

Alternative methods (but not better)

- We also explore alternative methods to balance the effects of PSNR-oriented and GAN-based methods. For instance, we can **directly interpolate** their output images (pixel by pixel or **Image Interpolation**) rather than the network parameters.
- However, such an approach fails to achieve a good trade-off between noise and blur, i.e., the interpolated image is either too blurry or noisy with artifacts.
- Another method is to tune the weights of content loss and adversarial loss, i.e., the parameter λ and η in Eq. (3) $\Rightarrow (L_G = L_{\text{percep}} + \lambda L_G^{Ra} + \eta L_1)$. But this approach requires tuning loss weights and fine-tuning the network, and thus it is too costly to achieve continuous control of the image style.

Experiments setup

- Training Details
- Data

Training Details

- Following SRGAN, all experiments are performed with a scaling factor of $\times 4$ between LR and HR images.
- We downloaded LR images which are down-sampled HR images using the **MATLAB bicubic kernel function**.
- The mini-batch size is set to 16. The spatial size of cropped HR patch is **128×128** .
- We observe that training a deeper network benefits from a larger patch size, since an enlarged receptive field helps to capture more semantic information. However, it costs more training time and consumes more computing resources. This phenomenon is also observed in PSNR-oriented methods.
- The training process is divided into two stages:
 1. **First**, we train a **PSNR-oriented model** with the **L1** loss. The learning rate is initialized as 2×10^{-4} and decayed by a factor of **2** every \times -mini-batch updates(specified in config files).
 2. We then employ the trained PSNR-oriented model as an initialization for the generator. The generator is trained using the loss function in Eq. (3): $(\mathbf{L}_G = \mathbf{L}_{\text{percep}} + \lambda \mathbf{L}_G^{Ra} + \eta \mathbf{L}_1)$ with $\lambda = 5 \times 10^{-3}$ and $\eta = 1 \times 10^{-2}$. The learning rate is set to 1×10^{-4} and halved at [40k, 60k, 100k, 130k] iterations.
 3. *Pre-training with pixel-wise loss* helps GAN-based methods to obtain more visually pleasing results.
- The reasons are that:
 1. It can avoid undesired local optima for the generator.
 2. After pre-training, the discriminator receives relatively good super-resolved images instead of extreme fake ones (black or noisy images) at the very beginning, which helps it to focus more on texture discrimination.
- For optimization, we use Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$. We alternately update the generator and discriminator network until the model converges. We use this setting for our generator : A deep model with **23 RRDB** blocks.

Data

- For training, we mainly use the DIV2K dataset, which is a high-quality (2K resolution) dataset for image restoration tasks.
- Beyond the training set of DIV2K that contains 800 images, we also seek for other datasets with rich and diverse textures for our training.
- We empirically find that using a large dataset with richer textures helps the generator to produce more natural results, as shown in **Fig. 8**.
- We train our models(PSNR ESRGAN) in RGB channels and augment the training dataset with random horizontal flips and 90 degree rotations. We evaluate our models on widely used benchmark datasets – Set5, Set14, BSD100 and artificially made dataset with images from google-photos.

1 st	2 nd	3 rd	4 th	5 th	6 th	7 th
BN?	✓	✗	✗	✗	✗	✗
Activation?	After	After	Before	Before	Before	Before
GAN?	Standard GAN	Standard GAN	Standard GAN	RaGAN	RaGAN	RaGAN
Deeper with RRDB?	✗	✗	✗	✗	✓	✓
More data?	✗	✗	✗	✗	✗	✓

Fig. 8

Quantitative Results

- PSNR/SSIM
- [butterfly.png] Bic=22.25db/0.75, SR=26.38db/0.72
- [head.png] Bic=32.01db/0.76, SR=30.70db/0.72
- [baby.png] Bic=31.96db/0.85, SR=31.35db/0.83
- [woman.png] Bic=26.44db/0.83, SR=28.30db/0.88
- [bird.png] Bic=30.27db/0.87, SR=31.84db/0.90

PSNR/SSIM

[bridge.png] Bic=24.38db/0.56, SR=22.21db/0.49
[barbara.png] Bic=25.19db/0.69, SR=24.34db/0.70
[baboon.png] Bic=22.06db/0.45, SR=20.56db/0.43
[zebra.png] Bic=24.15db/0.68, SR=24.40db/0.65
[ppt3.png] Bic=21.76db/0.82, SR=24.72db/0.92
[foreman.png] Bic=27.65db/0.86, SR=29.33db/0.88
[coastguard.png] Bic=25.33db/0.52, SR=23.19db/0.42
[flowers.png] Bic=25.85db/0.72, SR=25.69db/0.73
[pepper.png] Bic=29.38db/0.88, SR=31.09db/0.81
[man.png] Bic=25.74db/0.68, SR=24.99db/0.67
[face.png] Bic=31.98db/0.76, SR=31.00db/0.73
[monarch.png] Bic=27.60db/0.88, SR=31.15db/0.92
[lenna.png] Bic=29.67db/0.80, SR=29.41db/0.79
[comic.png] Bic=21.69db/0.59, SR=20.92db/0.62

PSNR



[pepper.png] Bic=29.38db/0.88, SR=29.89db/0.84



[butterfly.png] Bic=22.25db/0.75, SR=25.38db/0.74

ERSGAN



[pepper.png] Bic=29.38db/0.88, SR=31.09db/0.81



[butterfly.png] Bic=22.25db/0.75, SR=26.38db/0.72

Interpolation results

Network Interpolation

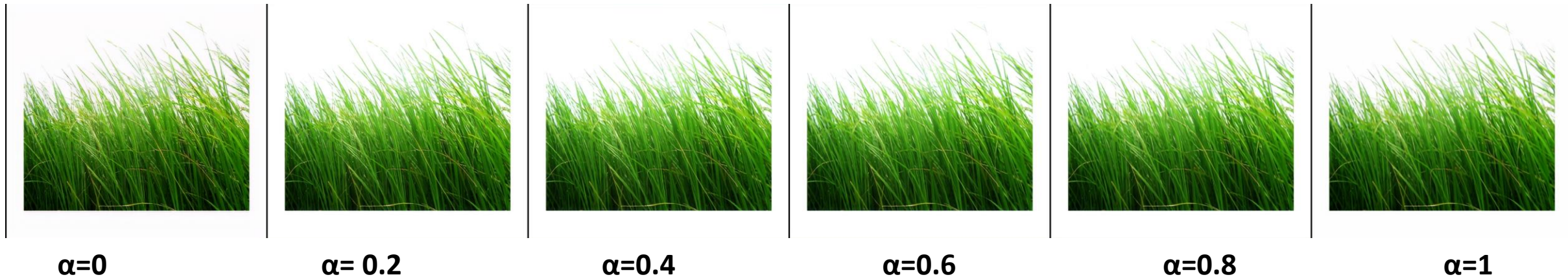
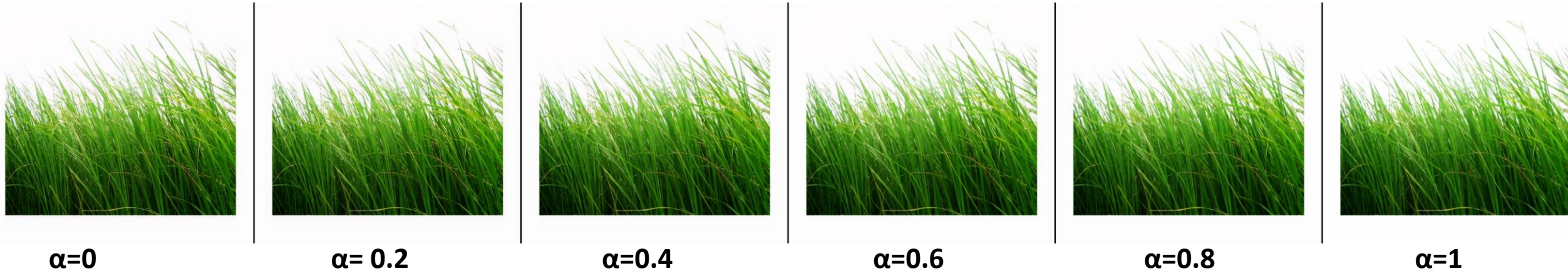


Image Interpolation



Qualitative Results

- We compare our final models on several public benchmark datasets with state-of-the-art **PSNR-oriented** methods including **SRCNN**, **EDSR** and **RCAN** and also with **perceptual-driven approaches** including **SRGAN** and **EnhanceNet**.
- It can be observed from Fig. 7 that our proposed **ESRGAN** outperforms previous approaches in both sharpness and details. For instance, **ESRGAN** can produce sharper and more natural baboon's whiskers and grass textures than **PSNR-oriented methods**, which tend to generate blurry results, and than previous GAN-based methods, whose textures are unnatural and contain displeasing noise. **ESRGAN** is capable of generating more detailed structures in building while other methods either fail to produce enough details (**SRGAN**) or add undesired textures.
- Moreover, previous GAN-based methods sometimes introduce unpleasant artifacts, e.g., **SRGAN** adds wrinkles to the face. Our **ESRGAN** gets rid of these artifacts and produces natural results.

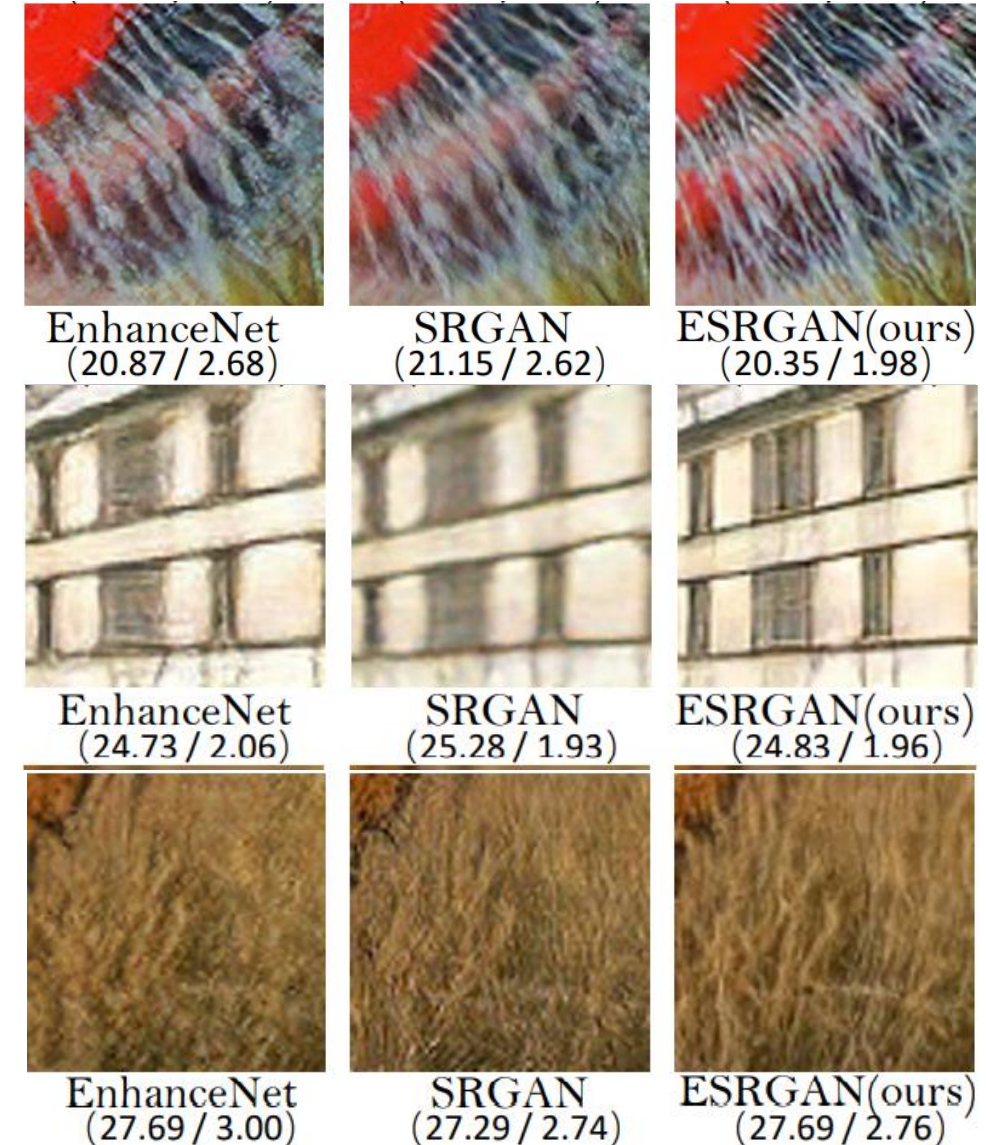


Fig. 7

Visual comparison

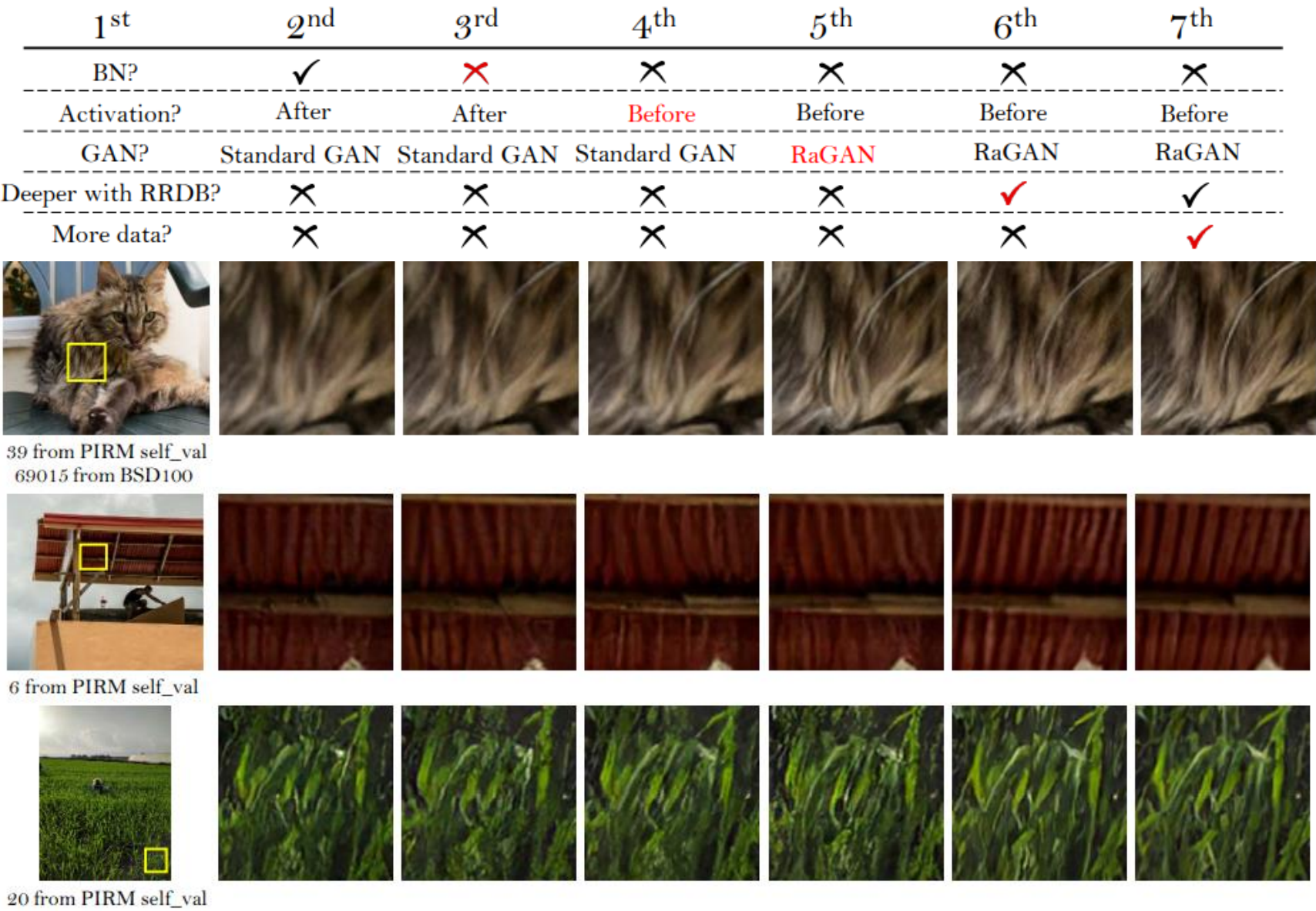


Fig. 8.

Supplementary information

RaGAN

- RaGAN uses an improved relativistic discriminator, which is shown to benefit learning sharper edges and more detailed textures.

Deeper network with RRDB

- **Deeper model** with the proposed **RRDB** can further improve the recovered textures, especially for the regular structures since the deep model has a strong representation capacity to capture semantic information.
- Also, we find that a deeper model can reduce unpleasing noises.
- In contrast to SRGAN, which claimed that deeper models are increasingly difficult to train, our deeper model shows its superior performance with easy training, thanks to the improvements mentioned above especially the proposed RRDB without BN layers.

Network Interpolation

Explanation

- We compare the effects of network interpolation and image interpolation strategies in balancing the results of a PSNR-oriented model and GAN-based method.
- We apply simple linear interpolation on both the schemes. The interpolation parameter α is chosen from 0 to 1 with an interval of 0.2. As depicted in Fig. 10 in the next slide, the pure GAN-based method produces sharp edges and richer textures but with some unpleasant artifacts, while the pure PSNR-oriented method outputs cartoon-style blurry images.
- By employing network interpolation, unpleasing artifacts are reduced while the textures are maintained. By contrast, image interpolation fails to remove these artifacts effectively. Interestingly, it is observed that the network interpolation strategy provides a smooth control of balancing perceptual quality and fidelity in Fig. 10.



Fig. 10: The comparison between network interpolation and image interpolation.

Conclusion

- We have presented an ESRGAN model that achieves consistently better perceptual quality than previous SR methods.
- We have formulated a novel architecture containing several RDDB blocks without BN layers.
- In addition, useful techniques including **residual scaling** and **smaller initialization** are employed to facilitate the training of the proposed deep model.
- We have also introduced the use of **relativistic GAN** as the **discriminator**, which learns to judge whether one image is more realistic than another, guiding the generator to recover more detailed textures.
- Moreover, we have **enhanced** the **perceptual loss** by using the **features before activation**, which offer stronger supervision and thus restore more accurate brightness and realistic textures.