Lightweight and Accurate Image Super-Resolution Using Progressive Mode

Anonymous CVPR submission

Paper ID ****

Abstract

Recently, models based on deep neural network have achieved great success in Single-Image Super-Resolution (SISR), while a key problem remain unsolved: how to build a general super-resolution model which can accommodate different upscaling factors and recover image texture detail? In this paper, we introduce a simple, flexible, and general mode for SISR: progressive mode. We present the Progressive Super-Resolution Network (ProSRN), a simplified version of the LapSRN model. Our model using the progressive mode to achieve large-scale amplification, rather than directly rebuild large-scale images. At each level, our model takes an un-preprocessed low-resolution image as input, reconstructs the sub-SR image and uses it as a new input image for the next level. Despite ProSRN shows superior performance over the state-of-the-art methods on benchmark datasets, we still find that the recovered images often lack texture details. To solve this problem, we extend this model to GANs architecture and present ProSRGAN, which is designed to differentiate between the generated residual images and the real residual images. The visual comparison shows that our ProSRGAN model can rebuild more realistic images.

1. Introduction

Single-image super-resolution **(SISR)**, a famous computer vision problem which aims to reconstruct a super-resolution (SR) image from a single low-resolution (LR) input image, has gained more attention in recent years. Now, the majority of methods for solving the SISR problem is to learn the mapping functions between LR and high-resolution (HR) images.

The development of deep neural networks especially convolution neural network **(CNN)** [16] has greatly promoted the process of solving the SISR problem. In 2014, Dong et al. [5] proposed Super-Resolution Convolutional Neural Network **(SRCNN)**, an efficient network which can learn a kind of end-to-end mapping between the LR and HR images. This is the first successful model adopting CNN to the SISR problem, the model uses pre-processed LR images as input which are upscaled to HR space using an upsampling operator like bicubic. However, Shi et al. [25] has proved that the use of preprocessed LR images will add computational complexity and produces visible artifacts in large scaling factors. To avoid this, some new methods have been proposed, e.g., Fast Super-Resolution Convolutional Neural Networks **(FSRCNN)** [6] and Efficient sub-pixel convolution networks **(ESPCN)** [25]. These two methods use untreated LR images as input, thus can significantly improve the effectiveness and efficiency. The mentioned models have a common attribute: less than 5 convolutional layers, because it's hard to train deeper networks.

Figure 1. An example of visual result for our models (ProSRN and ProSRGAN) . The ProSRGAN model can reconstruct more realistic images. [4x upscaling]

Inspired by the residual neural network proposed by He et al. [9], we can easily train deeper networks. Following this, Kim et al. [13] proposed Very Deep Convolutional Network **(VDSR)** for SISR which has 20 convolutional layers, the network learns a residual image in order to reconstruct natural images. Now, the super-resolution models has been extended to deeper and more complex structures, e.g., Deeply-recursive Convolutional Network **(DRCN)** [14] which trains a deeply recursive layers; Deep Recursive Residual Network **(DRRN)** [28] which adapts global residual learning to the identity branch containing 25 residual units (52 convolutional layers); Deep Residual

Method	Input	Layer	Mode
SRCNN [5]	bicubic LR	3	Direct
VDSR [13]	bicubic LR	20	Direct
DRCN [14]	bicubic LR	20	Direct
DRRN [28]	bicubic LR	52	Direct
ESPCN [25]	LR	3	Direct
FSRCNN [6]	LR	8	Direct
SRResNet [19]	LR	39	Direct
LapSRN [17]	LR	27	Progressive
ProSRN (Ours)	LR	14	Progressive

Table 1. Comparison of the structure of SR methods: SRCNN, FSRCNN, ESPCN, VDSR, DRCN, DRRN, SRResNet and LapSRNN. The number of layer include convolutional layer and transposed convolutional layer [4x upscaling].

Network **(SRResNet)** [9] which uses VGG [26] network to extract high-level feature maps; Laplacian Pyramid Super-Resolution Network **(LapSRN)** [17] which reconstructs the sub-band residuals of HR images. Although, these models have been improved by indicators such as **PSNR** and **SSIM** [30], they do not have significant improvements in visual performance. To better illustrate the differences between these models, we compared them in table 1. Meanwhile, most mentioned SR models usually train a specialized network for different upscaling factors. Several models that can accommodate different upscaling factors by combine multiple sub-networks into a large network. As the upscaling factor increases, the network structure becomes larger.

In the past studies, super-resolution was regarded as the progress of learning a mapping between LR and HR images. However, it is difficult to reconstruct realistic images at large upscaling factors (e.g, 4x, 8x) because LR images lose too much high-frequency information. In 2015, Goodfellow et al. [8] proposed Generative Adversarial Nets **(GANs)**. Following this, Mirza et al. [22] proposed Conditional Generative Adversarial Nets **(cGANs)** which believe GAN model can be extend to a conditional model if both the generator and discriminator have some extra information; Isola et al. [12] presented **Pix2Pix** which based on cGANs and was used to solve image-to-image translation problems. Inspired by GANs, Ledig et al. [19] proposed SRGAN which aims to train the generator to reconstruct SR images with the goal of fooling the discriminator that is trained to distinguish SR images from real images. Although the SRGAN model can reconstruct realistic images, careful observation reveals that the reconstructed image contains lots of noise that does not exist in the original image.

Over all, there are two main issues. First, how to build a general SR model that can be applied to different upscaling factors without increasing the complexity of the network? Second, how to recover the finer texture details at large upscaling factors? To resolve these issues, we introduce the Progressive Mode, and present the Progressive Super-Resolution Network **(ProSRN)**. Our model takes an un-preprocessed LR image as input and directly predicts the sub-SR image. Follow LapSRN, at each level, we first extract the feature map and apply a transposed convolutional layers for upsampling the feature maps to a finer level. Then we fuse the feature maps to predict the residual image. Finally, the residual image and upsampled LR image are used to reconstruct sub-SR image by addition operations. Unlike LapSRN, we propose a simple memory block to extract image features, and we do not send the upsampled feature maps into another sub-network for larger feature maps prediction. Instead, the same network is used, and the image generated at the previous level (sub-SR image) is used as a new input image to predict the finer SR image. ProSRN performs mixed training through a cascading loss function to achieve better results at large upscaling factors. Further more, we extend this network to GANs architecture and present **ProSRGAN**, a generative adversarial Network for SR. This model contain a generator (G) network and a discriminator (D) network which can reconstruct realistic images. To achieve this, we design a new loss formulations which consists of a adversarial loss and a residual loss. The G network is used to generate the residual images, the D network is trained to distinguish the generated residual images from the real residual images. Meanwhile, our D network use the patchGAN method proposed by pix2pix [12], which can achieve accurate judgments. An example of realistic image reconstructed by ProSRGAN model is shown in Figure 1. Our main contributions are:

- With the powerful learning ability of neural networks, we present a simple, flexible, and general mode for SISR: progressive mode. It breaks down complex challenges into small, controllable sub-tasks. This idea can also be extended to other areas to solve complex problems.

- We present two models, the ProSRN model can achieve fast and accurate image super-resolution, using only a simple network structure; the ProSRGAN is a GANs model which optimized for a new compound loss formulation, and can reconstruct more realistic images.

2. Proposed Methods

In this section, we first introduce the progressive mode, and then describe proposed models (ProSRN and ProSRGAN), including network architecture, memory block and loss functions. Finally, we describe the mixed training method which can improve the robustness of the network.

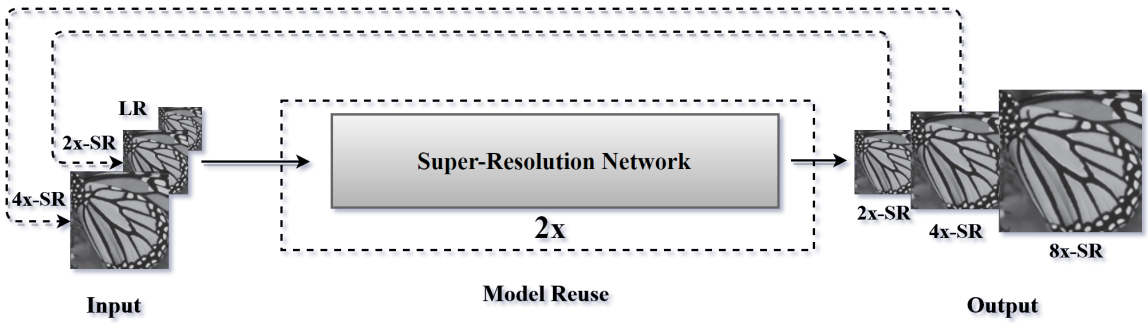

Figure 2. **Progressive mode:** At each level, our model takes an low-resolution image as input to predicts the sub-SR image (2x-SR). Then the model takes the 2x-SR image as new input to predicts 4x-SR by reuse the model. Through this method, our model can theoretically achieve unlimited amplification.

2.1. Progressive mode

In LapSRN [17], Lai et al. proposed the deep laplacian pyramid networks for SR. Take 8x as an example, LapSRN contains three different sub-networks. The model take LR image as input and progressively predicts the sub-band residuals in a coarse-to-fine fashion, each sub-network is responsible for predicting a sub-residual. In order to reduce network parameters, Lai et al recently proposed MS-LapSRN [18]. Comparing these two models, we find that the principle is similar. By weight sharing, MS-LapSRN reduces multiple sub-network in LapSRN to one. The core idea of these two models is consistent, the network takes an LR image as input, using a cascade of convolutional layers to extract feature maps and upsampling the feature maps to a finer level. Then, these feature maps are fed into another network (MS-LapSRN uses the same network) for the same operation.

As we know, SISR is an ill-posed problem, it is difficult to rebuild HR image from heavily downsampled LR image. However, it is simple to reconstruct small-scale images by deep neural networks. So, we break down complex challenges into small, controllable sub-tasks, presenting a simple, flexible, and general mode for SISR: progressive mode. As shown in Figure 2, at each level our model reconstruct a small-scale (e.g., 2x) image. Take 8x as an example, we divided it into three levels, at each level our model only rebuild a 2x sub-SR image, and this image is used as a new input image to rebuild the finer SR image.

Our model takes an LR image as input and progressively reconstructs the SR image in a coarse-to-fine fashion. At each level, our model takes the sub-SR image generated by the previous level as a new input, rather than use the upsampled feature maps as input. Our method is more simple and flexible, and use sub-SR images as input can get more useful features. Furthermore, the sub-SR image increases the diversity of training samples in training, which can improve the robustness of the network. The result shows that our method can accommodate different upscaling factors without increasing the complexity of the network, and our model can get a good balance between different upscaling factors.

2.2. CNN-based model: ProSRN

The solution to the SISR problem is estimate a super-resolution image SR from a low-resolution image LR. The LR in training dataset is obtained by the bicubic operation of the high-resolution image HR. We denote the LR image by x, the corresponding HR image by y and the reconstructed image by $y^{'}$.

2.2.1 Objective

Our ultimate goal is to establish a network G to estimate the corresponding output image y for a given input image x. The objective can define as:

$$Ł(G) = \mathbb{E}_{x,y\sim Pdata(x,y)}\rho(y - G(x)). \quad (1)$$

where $G(x)$ represents $y^{'}$ which reconstructed by the network G. The loss function ρ is used to minimize the difference between y and $y^{'}$.

The most widely-used image objective optimization function is the **MSE** function. Although this method can achieving high PSNR/SSIM, solutions of MSE optimization problems often produce overly smooth textures. Now, there are also many different loss functions have been used to learn the mapping from images to images (e.g., **VGG** function [19], L1 function [12]). Considering the effectiveness and complexity of these functions, We choose the Charbonnier Penalty Function [4, 17] which can accelerate the convergence of the network:

$$\rho(z) = \sqrt{z^2 + \varepsilon^2}, \quad (2)$$

where ρ is the Charbonnier Penalty Function, a differentiable variant of L1 norm. During the experiment, we set ε to 10^{-6}, and we aim to solve:

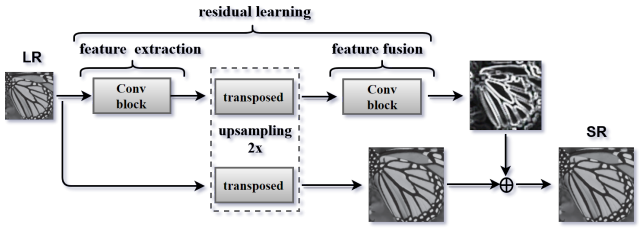

Figure 3. **ProSRN Architecture** Our model takes an unpreprocessed low-resolution image as input, using the feature extraction block, upsampling block and feature block to generate fine residual image.

$$G^* = arg \min_G Ł(G). \tag{3}$$

2.2.2 Network architecture

For SR image reconstruction, we use a simple convolutional network inspired by VDSR [13] and LapSRN [17]. As shown in Figure 3, our network contains two stages: residual learning and image reconstruction. At the same time, there are three different modules in the residual learning stages: (1) feature extraction, (2) upsampling and (3) feature fusion.

In residual learning stage, our model takes an unpreprocessed LR images as input and uses the following operations to generate the residual image.

Feature extraction. LapSRN [17] model uses a cascade of convolutional layers to extract feature maps. However, with the increase of network depth, the features of the image will gradually disappear in the process of transmission, so that these features can not be well utilized and the network becomes difficult to train. Recent studies [9, 10, 28] have shown that skip connections and local residual learning can greatly improve the accuracy of the network. But these models contain multiple recursive blocks or residual blocks, which greatly increases the complexity of the model. In order to balance network performance and network complexity, we propose a simplified version of the recursive network [28], and we call it memory blocks. As shown in Figure 4, our memory block adopted the skip connections, which connects source layer to every other layer in a feed-forward fashion.

Upsampling. We use a transposed convolutional layer [7] for upsampling the feature maps learned by memory block to high-resolution space.

Feature fusion. How do the feature maps learned by the memory block transformed into residual image ? What size convolutional kernel should be use ? Inspired by **GoogleNet** [27], we introduce an inception block to allow the network to learn how to combine these feature maps and transform

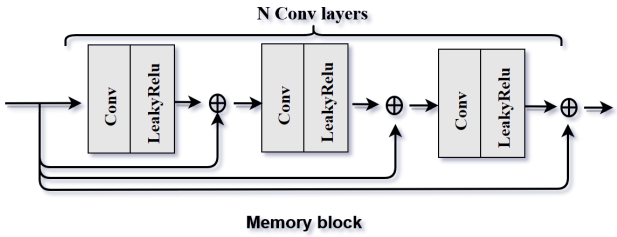

Figure 4. The **Memory block** used to extract image features in the residual learning stage.

them into an accurate residual image. Although this approach proved to be effective in our experiments, it leads to an increase in network complexity. In this work we want to explore a simple SR model, so we finally use a convolution layer instead of it.

In image reconstruction stage, the SR image is generated by adding the LR image processed by a transposed convolutional layer and the residual image.

2.3. GANs-based model: ProSRGAN

Recently, GANs has been applied in many areas which is a generative model aim to learns a mapping from random noise vector z to output image y, and conditional GANs learns a mapping from observed image x and random noise vector z to output image y. All these models contain two parts, the generator (G) and the discriminator (D). The discriminator is used to determine whether two objects are identical, the feedback information of the discriminator can be use to optimize the generator.

2.3.1 Objective

Our ProSRN model contains two stages: residual learning and image reconstruction. The learned residual image affects the quality of the reconstructed SR image. So we extend this model to GANs architecture and present ProSRGAN. Aa shown in Figure 5, we design a discriminators (D) network. which is not trained to distinguish between super-resolution images and real images, but is trained to distinguish between generated residual images and true residual images. At the same time we use PatchGAN [12] to achieve fine discrimination.

We denote the LR image by x, the upscaled LR image by $x^{'}$, the corresponding HR image by y and the residual image by r. The output residual images is modeled by $r = y - x^{'}$. We express the objective as:

$$\begin{aligned} Ł_{GAN}(G, D) =& \mathbb{E}_{r\sim Pdata(r)}[\log D(r)]+ \\ & \mathbb{E}_{x,x'\sim Pdata(x,x')}[\log(1 - D(G(x) - x^{'}))], \end{aligned} \tag{4}$$

where G is use the previously proposed ProSRN model, and G tries to minimize this objective against an adversarial D that tries to maximize the objective $G^* = arg \min_G \max_D Ł_{GAN}(G, D)$.

At the same time, the loss function of the G network can be redefined as:

$$Ł_c(G) = \mathbb{E}_{x,x',r\sim Pdata(x,x',r)}\rho(r - (G(x) - x')), \quad (5)$$

$$\rho(z) = \sqrt{z^2 + \varepsilon^2}. \quad (6)$$

Our full objective is:

$$Ł(G, D) = Ł_{GAN}(G, D) + \lambda Ł_c(G), \quad (7)$$

where λ controls the relative importance of these two objectives. We aim to solve the full objective:

$$G^* = arg \min_G \max_D Ł(G, D). \quad (8)$$

2.3.2 Network architecture

We adapt our discriminator architectures from SRGAN [19], the difference is that we use the patchGAN architecture proposed by pix2pix [12]. This discriminator tries to classify if each N * N patch in an image is real or fake,rather than directly judging the entire image. This architecture has been proved that has fewer parameters, runs faster, and can be applied on arbitrarily large images. Most importantly, the method of judging the patch can take into account the local structural information of the image, so the model can chieve accurate judgments and rebuild more realistic images.

2.4. Mixed training method

The multi-scales training method have been shown more effective than single-scale training method in VDSR [13] and LapSRN [17]. In our work, we introduce mixed training to improve the accuracy of the network. Unlike LapSRN, all of our SR images of different upsampling scales are generated from the same model. At each level, our model takes the sub-SR image generated at the previous level as a new input, which greatly increases the diversity of training samples, allowing the network to be able to handle more complex downsampled images. In essence, our network contains only one scale (2x), but mixed training of different levels images can increase the robustness of our network, the cascade loss function can be defined as follows:

ProSRN:

$$Ł_{Mixed}(G) = \sum_{i=1}^{N} Ł_i(G), \quad (9)$$

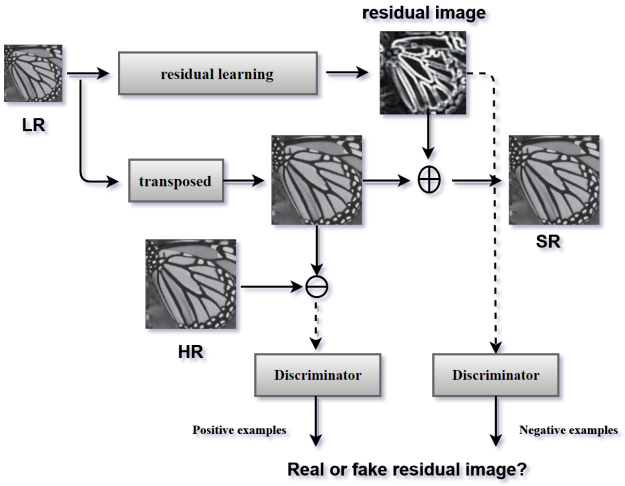

Figure 5. The discriminator D, learns to classify between real and fake residual images. The generator learns to fool the discriminator, and the generator reuse ProSRN model.

ProSRGAN:

$$Ł_{Mixed}(G, D) = \sum_{i=1}^{N} Ł_i(G, D), \quad (10)$$

where i stands for the level and $N = log_2 S$, S stands for the upscaling factor. Take 8x as an example, $N = log_2 8 = 3$, so our model requires three level to reconstruct the final SR image. We add the losses at each level, and aim to solve the $G^*_{Mixed} = arg \min_G Ł_{Mixed}(G)$ or $G^*_{Mixed} = arg \min_G \max_D Ł_{Mixed}(G, D)$.

3. Analysis

In this section, we first analyze the advantages of progressive mode. Then we verify the validity of the proposed feature extraction structure and compare it with other structures. Finally, we show the effectiveness of mixed training method.

3.1. Progressive mode

In Section 2.1, we propose the progressive mode, this is a simple SR mode which can suitable for different upscaling factors without change the network. In order to prove its effectiveness, we trained a ProSRN model for 2x scale without using the mixed training method. Then, use the asymptotic amplification method, we tested the accuracy of our ProSRN model at three different upscaling scales (2x, 4x, 8x). At the same time, we trained three LapSRN models for different upscaling scales. All models are completed under the same equipment and the same training settings,

	Train	Test Scale	SET5 PSNR / SSIM	SET14 PSNR / SSIM	BSDS100 PSNR / SSIM	URBAN100 PSNR / SSIM	MANGA109 PSNR / SSIM
LapSRN	2x	2	37.52 / 0.9581	33.08 / 0.9109	31.80 / 0.8949	30.41 / 0.9112	37.27 / 0.9725
	4x	4	31.54 / 0.8811	28.19 / 0.7635	27.32 / 0.7162	25.21 / 0.7564	29.09 / 0.8845
	8x	8	26.15 / 0.7028	24.45 / 0.5792	24.54 / 0.5293	21.81 / 0.5555	23.39 / 0.7068
ProSRN	2x	2	37.54 / 0.9584	33.18 / 0.9113	31.86 / 0.8967	30.81 / 0.9177	37.29 / 0.9737
	2x	4	31.44 / 0.8821	28.21 / 0.7684	27.29 / 0.7217	25.25 / 0.7648	28.95 / 0.8875
	2x	8	26.19 / 0.7047	24.47 / 0.5878	24.57 / 0.5360	21.94 / 0.5725	23.54 / 0.7273

Table 2. Quantitative comparisons of the LapSRN and our ProSRN: the LapSRN training different networks at different upscaling scales, the ProSRN model is only trained on a single upsampling scale (2x). Red text indicates the best performance.

Structure	Scale	SET5 PSNR / SSIM	SET14 PSNR / SSIM	BSDS100 PSNR / SSIM	URBAN100 PSNR / SSIM	MANGA109 PSNR / SSIM
LapSRN (A)	2	37.52 / 0.9581	33.08 / 0.9109	31.80 / 0.8949	30.41 / 0.9112	37.27 / 0.9725
Residual (B)	2	37.52 / 0.9882	33.10 / 0.9110	31.82 / 0.8952	30.60 / 0.9155	37.27 / 0.9730
Memory (C)	2	37.54 / 0.9584	33.18 / 0.9113	31.86 / 0.8967	30.81 / 0.9177	37.29 / 0.9737

Table 3. Quantitative comparisons of three different structures on public benchmark datasets. Red text indicates the best performance. The memory structure achieves best performance.

and we use the same number of convolution layers as LapSRN. In Table 2, the results shows that our model achieves the best results on every scale (2x, 4x, 8x). This fully illustrates that this method is effective at large upscaling factors, and worth mentioning that this method can be flexible to deal with larger upscaling factors like 16x.

3.2. Feature extraction structure

With the increase of network depth, the problem of gradient disappearance or explosion will occur during training. As shown in Figure 6, we explored three network structures for feature extraction:

General Structure: A widely used network structure is also used in the LapSRN model.

Residual Structure: The network structure designed by ResNet can effectively prevent gradient disappearance or explosion problem.

Memory Structure: A simplified version of the recursive block [28], which connects source layer to every other layer.

We designed a set of comparative experiments, using the above structure to test the performance of the model. All experiments were carried out under the same equipment and training settings. In Table 3, we show the results of these structure tests on five benchmark datasets. The proposed simple memory block achieves best performance without increase the complexity of the network.

3.3. Network depth

As we know, as the depth of the network increases, the model parameters will increase. This will cause the network hard to train, and will increase the time for image reconstruction. Therefore, we designed an experiment to determine the depth of the network. The convolutional layer of our model is mainly concentrated in the memory block, so we change the number of layers of memory block and train these models on the same equipment and training settings. Experimental results shown in Figure 7, we found that the accuracy of the network increases with the depth of the network. That does not mean that the deeper the better, with the network depth increasing, the convergence rate

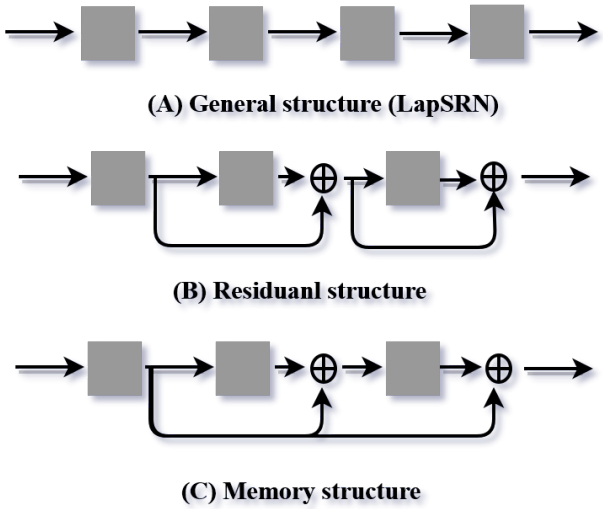

Figure 6. Three different network structures are used for image feature extraction.

Mixed training	Test Scale	SET5 PSNR / SSIM	SET14 PSNR / SSIM	BSDS100 PSNR / SSIM	URBAN100 PSNR / SSIM	MANGA109 PSNR / SSIM
ProSRN (2x)	2	37.54 / 0.9584	33.18 / 0.9113	31.86 / 0.8967	30.81 / 0.9177	37.29 / 0.9737
ProSRN (2x, 4x)	2	37.48 / 0.9582	33.04 / 0.9110	31.81 / 0.8963	30.56 / 0.9157	37.25 / 0.9737
ProSRN (2x, 4x, 8x)	2	36.94 / 0.9560	32.65 / 0.9085	31.50 / 0.8933	29.72 / 0.9061	36.28 / 0.9705
ProSRN (2x)	4	31.44 / 0.8821	28.21 / 0.7684	27.29 / 0.7217	25.25 / 0.7648	28.95 / 0.8875
ProSRN (2x, 4x)	4	31.58 / 0.8842	28.24 / 0.7688	27.32 / 0.7217	25.32 / 0.7665	29.19 / 0.8906
ProSRN (2x, 4x, 8x)	4	31.54 / 0.8828	28.25 / 0.7694	27.29 / 0.7219	25.25 / 0.7650	29.20 / 0.8903
ProSRN (2x)	8	26.19 / 0.7047	24.47 / 0.5878	24.57 / 0.5360	21.94 / 0.5725	23.54 / 0.7273
ProSRN (2x, 4x)	8	26.28 / 0.7126	24.50 / 0.5895	24.60 / 0.5376	21.97 / 0.5765	23.69 / 0.7331
ProSRN (2x, 4x, 8x)	8	26.26 / 0.7117	24.58 / 0.5897	24.60 / 0.5376	21.99 / 0.5760	23.70 / 0.7325

Table 4. Quantitative comparisons of single-scale training method and mixed training method. The mixed training method gains better results at large upscaling factors.

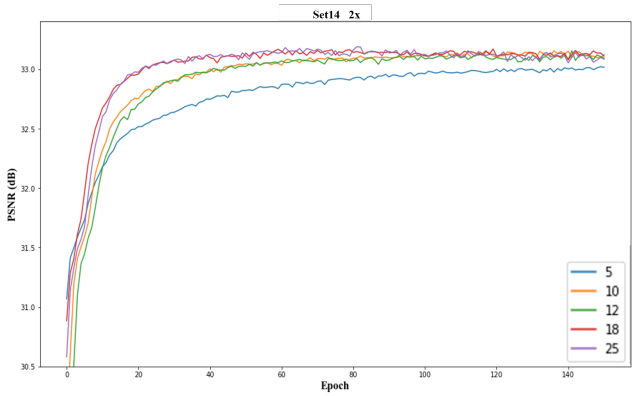

Figure 7. Different color curves represent the use of different numbers of convolutional layers in memory block. The deeper, the faster the network converges.

of the network becomes faster, but the accuracy is already saturated and not changing.

3.4. Mixed training method

We train our ProSRN model using the mixed training method. Our model is very simple, only need to add the loss at all levels. In order to show its effect, we use three different levels (2x, 4x, 8x) for training. As shown in Table 4, we can find that train (2x) can achieve best results on 2x SR. Similarly, train (2x, 4x) can achieve best results on 4x SR, train (2x, 4x, 8x) will achieve best results on 8x SR. This shows that mixed training method can improve the effect of the network at large upscaling factors.

4. Experiment Results

In this section, we first introduce the datasets used for training/testing and give our detail training steps. Then, we compare our models (ProSRN / ProSRGAN) with some state-of-the-art SR mothods on 5 benchmark datasets and show the visual comparison of these methods. Finally, we show the comparison of model parameters and run time.

4.1. Datasets

In previous studies, some methods take ImageNet [23] as training dataset, and other methods use training dataset including 291 images, of which 91 images from Yang et al. [31] another 200 images from Berkeley Segmentation Dataset [20]. In our work, we carry out large upscaling factors, so we use higher quality training datasets DIV2K [1] (800 training images) as our training dataset, a new high-quality image dataset for image restoration tasks. As for the test, we choose five widely used benchmark datasets: Set5 [3], Set14 [32], BSDS100 [2], Urban100 [11] and Manga109 [21]. These five datasets contain natural scenes, urban scenes and Japanese manga, which can sufficiently test the generalization abilities of the network.

4.2. Training details

We train our models on four NVIDIA Titan X GPUs using a random sample of 125 thousand images cuts form DIV2K. We generate the LR training images using the bicubic down sampling from HR images, and HR images only observed on training. Our models finally uses our proposed memory structure to extract LR image feature maps, and the memory structure uses 10 convolution layers. Following previous works, we only use the luminance channel in YCbCr colour space during our experiments, because the paper [24] pointed out that human are more sensitive to luminance changes. In this work, all training and testing are based on luminance channel.

We trained our model with ADAM optimizer [15], and set momentum parameter to 0.9. The ProSRN model was trained with a initial learning rate of 10^{-4}. When training ProSRGAN model, we employ the trained ProSRN as initialized generator to avoid undesired local optimal. ProSRGAN were trained at a low learning rate of 10^{-5}. We

alternately updates the generator and discriminator network as the same in Goodfellow et al. [8] and Ledig et al. [19]. The additional mapping loss function helps stabilize training which affects the quality of the generated images. In practice, we found that λ setting too big would lead to sharper results, and λ setting too small would cause blurred images, so we empirically set $\lambda = 10^{-2}$ which can get best results.

4.3. Comparisons with state-of-the-art methods

In this section, we evaluate our model from two aspects: image quality and model parameters.

4.3.1 Image quality assessment

We compare our ProSRN model with 9 state-of-the-art super-resolution algorithms, including Bicubic, A+ [29], RFC [24], SRCNN [5], FSRCNN [6], SelfExSR [25], VDSR [13], DRCN [14], LapSRN [17]. All these SR models are based on CNN architecture, except for Bicubic and A+. We also compared our ProSRGAN model with the SRGAN [19] model, which are all based on GAN architecture.

For fair comparison, we evaluate the SR images with two commonly-used image quality metrics: PSNR and SSIM [30], compareing performance on 2x, 4x and 8x SR. Moreover, all reported PSNR/SSIM measures were calculated on the luminance channel, and removed of a scale-pixel wide strip from each border. (N scale remove N-pixel wide). We re-train the existing method using the source code or the code provided by the GitHub community. But due to limited ability, we can't restore SRGAN model very well, so we use the reconstructed SR image provided by SRGAN [19] paper (this paper provides a download link, including the reconstructed images of the three datasets of Set5, Set14 and BSDS100 [4x upscaling]).

In Table 5, we show the comparison between our ProSRGAN model and the SRGAN model. Our ProSRGAN model achieves the best results, whether PSRN or SSIM. Table 6 shows the quantitative results of SR models based on the CNN architecture include our ProSRN model. Our ProSRN model achieves best results especially on scale factors 4x and 8x. It's worth noting that we didn't specifically build a deep network for large upscaling factors, with the asymptotic amplification and multi-scale mixed training methods, ProSRN uses only a simple SR model but sets a new state-of-the-art on public benchmark datasets. We show visual comparisons in figure **??**, and Figure **??** shows the reconstructed images of our models (ProSRN and ProSRGAN) on different upscaling factors. For better viewing, we provide our SR results on our website at http://www.prosrn.com.

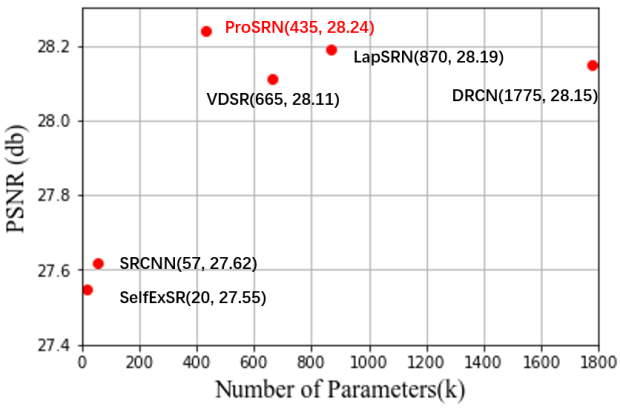

Figure 8. Comparision of model parameters and performance. The results are evaluated on the Set14 dataset for. The proposed DPSRN strides a balance between reconstruction accuracy and parameters number. [4x upscaling]

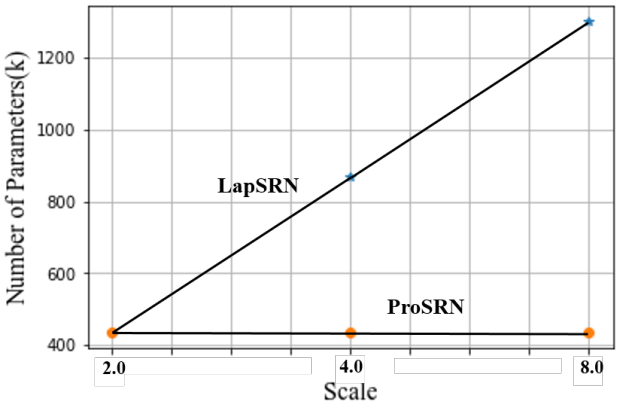

Figure 9. Comparision of model parameters at different upscaling factors. Our ProSRN model parameters do not change with the upscaling factor.

4.3.2 Comparison of model parameters

As shown in Figure 8, we compared the CNN-based SR model for the reconstruction performance and the parameters of the model. Our ProSRN model gets the best results an achieves best balance between the number of parameters and reconstruction performance. In addition, we compared the parameters of these models (PreSRN and LapSRN) at different upscaling factors in Figure 9, the results show that our model parameters do not change with the upscaling factor, this will allow our model to be applied in more scenarios.

Dataset	Scale	Bicubic PSNR / SSIM	SRGAN-VGG54 PSNR / SSIM	**ProSRGAN (Ours)** PSNR / SSIM
Set5	4	28.43 / 0.8022	29.43 / 0.8296	29.19 / 0.8471
Set14	4	26.10 / 0.6936	26.12 / 0.6882	26.31 / 0.7296
BSDS100	4	25.97 / 0.6517	25.18 / 0.6288	25.43 / 0.6711

Table 5. Quantitative comparisons of Bicubic, SRGAN-VGG54 and ProSRGAN. Red text indicates the best performance

Algorithm	Scale	SET5 PSNR / SSIM	SET14 PSNR / SSIM	BSDS100 PSNR / SSIM	URBAN100 PSNR / SSIM	MANGA109 PSNR / SSIM
Bicubic	2	33.69 / 0.9284	30.34 / 0.8675	29.57 / 0.8434	26.88 / 0.8438	30.82 / 0.9332
A+ [29]	2	36.60 / 0.9542	32.42 / 0.9059	31.24 / 0.8870	29.25 / 0.8955	35.37 / 0.9663
RFL [24]	2	36.59 / 0.9534	32.38 / 0.9035	31.18 / 0.8843	29.14 / 0.8920	35.12 / 0.9647
SRCNN [5]	2	36.71 / 0.9536	32.32 / 0.9052	31.36 / 0.8880	29.54 / 0.8962	35.74 / 0.9661
FSRCNN [6]	2	37.06 / 0.9554	32.76 / 0.9078	31.53 / 0.8912	29.88 / 0.9024	36.67 / 0.9694
SelfExSR [25]	2	36.60 / 0.9537	32.46 / 0.9051	31.20 / 0.8863	29.55 / 0.8983	35.82 / 0.9671
VDSR [13]	2	37.53 / 0.9583	33.05 / 0.9107	31.92 / 0.8965	30.79 / 0.9157	37.22 / 0.9729
DRCN [14]	2	37.63 / 0.9584	33.06 / 0.9108	31.85 / 0.8947	30.76 / 0.9147	37.63 / 0.9723
LapSRN [17]	2	37.52 / 0.9581	33.08 / 0.9109	31.80 / 0.8949	30.41 / 0.9112	37.27 / 0.9725
ProSRN (Ours 2x)	2	37.54 / 0.9584	33.18 / 0.9113	31.86 / 0.8967	30.81 / 0.9177	37.29 / 0.9737
Bicubic	4	28.43 / 0.8022	26.10 / 0.6936	25.97 / 0.6517	23.14 / 0.6599	24.91 / 0.7826
A+ [29]	4	30.33 / 0.8565	27.44 / 0.7450	26.83 / 0.6999	24.34 / 0.7211	27.03 / 0.8439
RFL [24]	4	30.17 / 0.8501	27.33 / 0.7407	26.76 / 0.6958	24.20 / 0.7119	26.80 / 0.0.8332
SRCNN [5]	4	30.50 / 0.8573	27.62 / 0.7453	26.91 / 0.6994	24.53 / 0.7236	27.66 / 0.8505
FSRCNN [6]	4	30.73 / 0.8601	27.71 / 0.7488	26.98 / 0.7029	24.62 / 0.7272	27.90 / 0.8517
SelfExSR [25]	4	30.34 / 0.8593	27.55 / 0.7511	26.84 / 0.7032	24.83 / 0.7403	27.83 / 0.8598
VDSR [13]	4	31.36 / 0.8796	28.11 / 0.7624	27.29 / 0.7167	25.18 / 0.7543	28.83 / 0.8809
DRCN [14]	4	31.56 / 0.8810	28.15 / 0.7627	27.24 / 0.7150	25.15 / 0.7530	28.98 / 0.8816
LapSRN [17]	4	31.54 / 0.8811	28.19 / 0.7635	27.32 / 0.7162	25.21 / 0.7564	29.09 / 0.8845
ProSRN (Ours 2x)	4	31.44 / 0.8821	28.21 / 0.7684	27.29 / 0.7217	25.25 / 0.7648	28.95 / 0.8875
ProSRN (Ours 2x,4x)	4	31.58 / 0.8842	28.24 / 0.7688	27.32 / 0.7217	25.32 / 0.7665	29.19 / 0.8906
Bicubic	8	24.40 / 0.6045	23.19 / 0.5110	23.67 / 0.4808	20.74 / 0.4841	21.46 / 0.6138
A+ [29]	8	25.53 / 0.6548	23.99 / 0.5535	24.21 / 0.6115	21.37 / 0.5193	22.39 / 0.6454
RFL [24]	8	25.38 / 0.6392	23.88 / 0.5442	24.13 / 0.5047	21.27 / 0.5075	22.28 / 0.6304
SRCNN [5]	8	25.34 / 0.6471	23.86 / 0.5443	24.14 / 0.5043	21.29 / 0.5133	22.46 / 0.6606
FSRCNN [6]	8	25.42 / 0.6440	23.94 / 0.5482	24.21 / 0.5112	21.32 / 0.5090	22.39 / 0.6357
SelfExSR [25]	8	25.49 / 0.6733	24.02 / 0.5650	24.19 / 0.5146	21.81 / 0.5536	22.99 / 0.6907
VDSR [13]	8	25.73 / 0.6743	23.20 / 0.5110	24.34 / 0.5169	21.48 / 0.5289	22.73 / 0.6688
LapSRN [17]	8	26.15 / 0.7028	24.45 / 0.5792	24.54 / 0.5293	21.81 / 0.5555	23.39 / 0.7068
ProSRN (Ours 2x)	8	26.19 / 0.7047	24.47 / 0.5878	24.57 / 0.5360	21.94 / 0.5725	23.54 / 0.7273
ProSRN (Ours 2x,4x,8x)	8	26.26 / 0.7117	24.58 / 0.5897	24.60 / 0.5376	21.99 / 0.5760	23.70 / 0.7325

Table 6. Quantitative comparisons of state-of-the-art methods: average PSNR/SSIM for scale factors 2x, 4x and 8x. Red text indicates the best and blue text indicates the second best performance. There are two ways for VDSR and LapSRN training, we select their best results.

5. Limitation and Future work

Although our ProSRN model sets a new state-of-the-art on large upscaling factors, e.g., 4, 8x, we found that the reconstructed images still lacked texture details. The introduce of the GAN architecture makes it possible for the network to reconstruct realistic images by generating missing texture details. However, this method makes the network more difficult to train. In the future, we will focus on the image texture detail reconstruction on a large upscaling factor.

6. Conclusions

In this work, we present a simple and flexible method for SISR: the large upscaling factor is decomposed into multiple small upscaling factors, and reconstructed images using the progressive models. We have described a simple super-resolution network ProSRN that sets a new state-of-the-art on public benchmark datasets especially at large upscaling factors (4x, 8x). In order to generate more realistic images, we propose the ProSRGAN model which based on GANs architecture. Our model suitable for different upscaling factors without increasing the complexity of networks. We believe that breaking down complex challenges into small, controllable sub-tasks can be promoted in other areas, and we believe that reducing the complexity of the model will enable it to be applied in more scenarios.

References

[1] E. Agustsson and R. Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017. 7

[2] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898, 2011. 7

[3] M. Bevilacqua, A. Roumy, C. Guillemot, and A. Morel. Low-complexity single image super-resolution based on nonnegative neighbor embedding. *Bmvc*, 2012. 7

[4] A. Bruhn, J. Weickert, and C. Schnrr. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, 61(3):211–231, 2005. 3

[5] C. Dong, C. L. Chen, K. He, and X. Tang. Learning a deep convolutional network for image super-resolution. 8692:184–199, 2014. 1, 2, 8, 9

[6] C. Dong, C. L. Chen, and X. Tang. Accelerating the super-resolution convolutional neural network. In *European Conference on Computer Vision*, pages 391–407, 2016. 1, 2, 8, 9

[7] V. Dumoulin and F. Visin. A guide to convolution arithmetic for deep learning. 2016. 4

[8] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *International Conference on Neural Information Processing Systems*, pages 2672–2680, 2014. 2, 8

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. pages 770–778, 2015. 1, 2, 4

[10] G. Huang, Z. Liu, K. Q. Weinberger, and V. D. M. Laurens. Densely connected convolutional networks. 2016. 4

[11] J. B. Huang, A. Singh, and N. Ahuja. Single image super-resolution from transformed self-exemplars. In *Computer Vision and Pattern Recognition*, pages 5197–5206, 2015. 7

[12] P. Isola, J. Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. 2016. 2, 3, 4, 5

[13] J. Kim, J. K. Lee, and K. M. Lee. Accurate image super-resolution using very deep convolutional networks. pages 1646–1654, 2015. 1, 2, 4, 5, 8, 9

[14] J. Kim, J. K. Lee, and K. M. Lee. Deeply-recursive convolutional network for image super-resolution. pages 1637–1645, 2015. 1, 2, 8, 9

[15] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *Computer Science*, 2014. 7

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *International Conference on Neural Information Processing Systems*, pages 1097–1105, 2012. 1

[17] W. S. Lai, J. B. Huang, N. Ahuja, and M. H. Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2, 3, 4, 5, 8, 9

[18] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang. Fast and accurate image super-resolution with deep laplacian pyramid networks. *arXiv:1710.01992*, 2017. 3

[19] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, and Z. Wang. Photo-realistic single image super-resolution using a generative adversarial network. 2016. 2, 3, 5, 8

[20] D. Martin, C. Fowlkes, D. Tal, and J. 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*, pages 416–423 vol.2, 2002. 7

[21] Y. Matsui, K. Ito, Y. Aramaki, A. Fujimoto, T. Ogawa, T. Yamasaki, and K. Aizawa. Sketch-based manga retrieval using manga109 dataset. *Multimedia Tools Applications*, pages 1–28, 2015. 7

[22] M. Mirza and S. Osindero. Conditional generative adversarial nets. *Computer Science*, pages 2672–2680, 2014. 2

[23] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 7

[24] S. Schulter, C. Leistner, and H. Bischof. Fast and accurate image upscaling with super-resolution forests. In *Computer Vision and Pattern Recognition*, pages 3791–3799, 2015. 7, 8, 9

[25] W. Shi, J. Caballero, F. Huszr, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. pages 1874–1883, 2016. 1, 2, 8, 9

[26] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *Computer Science*, 2014. 2

[27] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. pages 1–9, 2014. 4

[28] Y. Tai, J. Yang, and X. Liu. Image super-resolution via deep recursive residual network. In *IEEE Computer Vision and Pattern Recognition*, 2017. 1, 2, 4, 6

[29] R. Timofte, V. D. Smet, and L. V. Gool. A+: Adjusted anchored neighborhood regression for fast super-resolution. In *Asian Conference on Computer Vision*, pages 111–126, 2014. 8, 9

[30] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 13(4):600, 2004. 2, 8

[31] J. Yang, J. Wright, T. S. Huang, and Y. Ma. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, 2010. 7

[32] R. Zeyde, M. Elad, and M. Protter. On single image scale-up using sparse-representations. In *International Conference on Curves and Surfaces*, pages 711–730, 2010. 7