Research Article

Image Super Resolution Using Fractal Coding and Residual Network

Zhen Hua ^{1,2} Haicheng Zhang ^{2,3} and Jinjiang Li ^{2,3}

¹School of Information and Electronic Engineering, Shandong Technology and Business University, Yantai, 264005, China

²Co-innovation Center of Shandong Colleges and Universities: Future Intelligent Computing, Shandong Technology and Business University, Yantai, 264005, China

³School of Computer Science and Technology, Shandong Technology and Business University, Yantai, 264005, China

Correspondence should be addressed to Zhen Hua; huazhen66@foxmail.com

Received 2 June 2019; Revised 7 August 2019; Accepted 6 November 2019; Published 28 November 2019

Academic Editor: Lingzhong Guo

Copyright © 2019 Zhen Hua et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Fractal coding techniques are an effective tool for describing image textures. Considering the shortcomings of the existing image super-resolution (SR) method, the large-scale factor reconstruction performance is poor and the texture details are incomplete. In this paper, we propose an SR method based on error compensation and fractal coding. First, quadtree coding is performed on the image, and the similarity between the range block and the domain block is established to determine the fractal code. Then, through this similarity relationship, the attractor is reconstructed by super-resolution fractal decoding to obtain an interpolated image. Finally, the fractal error of the fractal code is estimated by the depth residual network, and the estimated version of the error image is added as an error compensation term to the interpolation image to obtain the final reconstructed image. The network structure is jointly trained by a deep network and a shallow network. Residual learning is introduced to greatly improve the convergence speed and reconstruction accuracy of the network. Experiments with other state-of-the-art methods on the benchmark datasets Set5, Set14, B100, and Urban100 show that our algorithm achieves competitive performance quantitatively and qualitatively, with subtle edges and vivid textures. Large-scale factor images can also be reconstructed better.

1. Introduction

The image imaging process is often affected by factors such as downsampling, noise, and blur. Image SR reconstruction is used to reconstruct a poor-quality low-resolution (LR) image into a high-resolution (HR) image close to the real image. SR is widely used in various fields, such as video surveillance, remote sensing, and medical imaging. Due to the uncertainty of the image degradation model and the nonuniqueness of the reconstruction constraints, SR is essentially an ill-posed problem [1]. There are three main approaches to solving the SR problem: interpolation-based, reconstruction-based, and learning-based approaches.

Interpolation-based methods estimate unknown pixels in HR images from pixels in their known domain, such as bilinear interpolation and bicubic interpolation. They are most widely used for their high speed and simplicity.

However, they are prone to blur and jagged details. To compensate for the shortcomings of traditional methods, an edge-directed interpolation method [2] is proposed. Although the edge structure is reconstructed, noise and distortion are prone to occur [3]. The reconstruction-based approach solves this problem by a priori knowledge, including edge a priori [4], gradient prior [5], total variation, and Bayesian models. However, due to the heavy dependence on a priori, the a priori is ineffective for large-scale amplification factors. The basic idea of the learning-based method is to obtain a mapping of LR images to HR images by training samples, thereby predicting HR images. The trained sample library can be either from within the LR image [6–8] or from an external image [9–13]. A large number of learning algorithms based on the Markov network [9], local linear embedding method [10], sparse coding [11–13], and anchor neighbors [14, 15] have been proposed. However,

these methods have limited ability to extract and express features during the learning process. In recent years, the method of deep learning is applied to many fields [16, 17], and many researchers have introduced the convolutional neural network (CNN) into SR reconstruction and learning end-to-end mapping between LR and HR images by relying on external data sets; for example, SRCNN [18], VDSR [19], and EDSR [20] show superiority. In particular, the proposed residual network makes the data transmission between the networks smoother, which makes the depth of the network increase and the reconstruction effect better. However, they still have considerable drawbacks. For example, when the image content does not match the sample data set, edge irregularities are observed in the results [21]. Moreover, this approach tends to produce blurred and overly smooth outputs, lacking detailed information. The current work is only suitable for some small and specific scaling factors, and the reconstruction of large-scale factors will have the disadvantages of ambiguity and lack of detail. To make the predicted image match itself as much as possible, a self-similar-driven SR algorithm (selfExSR) [7] is proposed, which can extend the search space of internal image blocks by geometric changes, thus improving the visual effect. Many researchers have also begun to combine multiple methods to try to reconstruct SR, and Dong et al. [22] apply machine learning methods to image interpolation and embed nonlocal autoregressive modeling (NARM) into sparse representation models. The visual effect is further improved, but as the magnification scale increases, the texture and edge details are not well reconstructed.

Fractal is an effective tool for describing image textures and is widely used in image segmentation [23], classification [24], SR, and other fields. The fractal-based SR method utilizes the locality of the similarity and transitivity of a single image to search for similar image blocks. Using the analogy method, the information of the LR image block is merged to reconstruct the HR image blocks [25]. The similarity can be considered as a contractive fractal transformation operator ω that performs shrinking and gray-level modifying operations on the image. The attractive fixed point of this operator can estimate the target image by the collage theorem. Therefore, only the transformed ω is stored in the computer, which requires less storage space than the original image. Wee and Shin [26] proposed a new fast fractal super-resolution algorithm using the feature type orthogonal fractal coding method, which produced good details and low computational complexity but lacked flexibility and adaptability. Later, the combination of fractal technology and other technologies produced good results. Xu et al. [27] proposed a texture enhancement algorithm that uses local fractal analysis to improve the SR effect of images and solves the problem of image SR and enhancement. Yu et al. [28] combined fractal technology with an instance-based approach to propose a super-resolution algorithm that preserves vivid texture details. However, partial shape features (such as fractal scale factor and fractal dimension) and image similarity have not been well considered and some unexpected

artifacts may occur. In recent years, the SR method based on fractal technology has only an effective number of documents, all of which are studies on how to preserve texture and structure information. Zhang et al. [3] proposed an SR reconstruction method based on rational fractal interpolation. First, the image is divided into texture regions and nontexture regions, and then the images are interpolated according to the characteristics of the local structures. Yao et al. [29] proposed an adaptive rational fractal interpolation model that reconstructs the relationship between the partial shape dimension and the vertical scale factor. These methods have some improvement in the recovery of the edges, but the fractal dimension does not accurately represent the texture details.

To better improve the reconstruction effect of texture and edge regions and further improve the reconstruction performance of large-scale factors, we applied fractal geometry and CNN to the study of SR. In this paper, we propose an SR method based on fractal coding and residual networks. Fractal image coding can use the spatial information of the image and self-similar structural information to achieve super-resolution of the image. The basic idea is to estimate the fractal code of the original image from its degraded version and decode it at a higher resolution. Natural images generally have self-similar characteristics [27] that are not as strict. The fractal code is only an approximate estimate of the original low-resolution image. The use of fractal codes to achieve image SR recovery inevitably introduces errors and leads to the loss of blockiness and partial detail. To improve this phenomenon, we input the encoded and decoded error image into the depth residual network for estimation and use this as a compensation term to correct the interpolation image to further improve the reconstruction accuracy. Since most of the error images are high-frequency details lost after fractal interpolation, to better learn the details in this part, we propose a method of the convolutional neural network for training. We constructed a deep residual network for joint training of deep networks and shallow networks, increasing the width of the network and learning different features. The introduction of the residual block increases the depth of the network, further improves the information flow gradient, facilitates the recovery of high-frequency information, and improves the reconstruction performance. Through the combination of fractal technology and residual network, the texture of SR results is richer and more realistic.

Our main contributions can be summarized as follows. (1) We propose a fractal coding method based on error compensation. (2) We propose a method to estimate the fractal error of fractal coding using the CNN method. (3) Compared with the state-of-the-art methods, our method does not lead to excessive smoothing and artifacts and exhibits superior performance as the scale factor increases.

The rest of this paper is organized as follows. Section 2 describes the related work. In Section 3, our algorithm is presented. Experiments and discussions are used to evaluate the effectiveness of the algorithm in Section 4. Finally, Section 5 summarizes this article.

2. Related Work

Fractal image coding was originally used for image compression and was later applied to image denoising, classification, super-resolution, and so on. Since the process of fractal decoding is independent of resolution and the fractal code contains not only spatial information but also self-similar structural information of the image, fractal image coding can realize image resolution reconstruction [24]. Before discussing the proposed method in depth, the basic principles of traditional fractal image coding are briefly introduced.

2.1. Fractal Image Coding. The process of fractal image compression coding is based on the collage theorem. A set of compression maps is obtained by a given image so that the attractor of the iterated function system is approximated to a given image, and then the corresponding parameters are recorded. Let (S, t) be the measure space formed by the digital image and t be the measure of the distance between the two images in (S, t) space. Assume there is a compression map $\omega = \{\omega_1, \omega_2, \dots, \omega_c\}$ and an image $F \in (S, t)$. If

$$F = \omega(F) = \bigcup_{i=1}^c \omega_i(F), \quad (1)$$

then F is the only attractor determined by ω . For any given initial image $\xi \in (S, t)$, F can be iteratively solved by the following equation:

$$F_\omega = \lim_{i \rightarrow \infty} (\omega^i(\xi)) = \lim_{i \rightarrow \infty} (\omega^{i-1}(\xi)). \quad (2)$$

The distance between the image $\omega^i(\xi)$ and F after the i th iteration satisfies

$$t(F, \omega^i(\xi)) \leq \frac{1}{S} \cdot t(F, \omega(F)) + s^n \cdot t(F, \xi), \quad (3)$$

where S is the compression factor of the compression map, $s = \max(s_1, s_2, \dots, s_n)$.

Let F denote a given image of size $N \times N$ to be encoded, where N is usually a power of two. First, F is divided into nonoverlapping sub-blocks of size $n \times n$, which are called range blocks R_i . Thus, image F can be expressed as

$$F = \bigcup_{i=1}^{N_r=(N/n)^2} R_i, \quad (4)$$

where N_r is the number of range blocks, and all range blocks R_i form the range R . Then, F can be divided into a group of domain blocks $D_i (j = 1, 2, \dots, N_d)$ of size $m \times m$ by a certain step size δ along the vertical and horizontal levels. The step size is generally 1, 2, or 3, which can overlap in the neighborhood and does not need to cover the entire image F . Usually, the size of the domain block is twice the size of the range block, that is, $m = 2n$; therefore, the number of domain blocks is $N_d = (((N - 2n)/\delta) + 1)^2$, and all domain blocks D_i constitute domain D .

In the image encoding phase, for each range block R_i , the best matching domain block D_i needs to be searched in the domain block. As shown in Figure 1, R_i can be approximated by D_i through a contractive transformation operator ω_i , which

FIGURE 1: Block mapping of fractal coding.

can be considered a collection of shrinking transformations and gray-level transformations. The shrinking transformation ϕ_i^k is a combination of geometric mappings. We need to convert D_i to a block of the same size as R_i , and the shrinking domain block is represented as $\hat{D}_i = \phi_i^k(D_i)$. There are eight such transformations, i.e., $1 \leq k \leq 8$, including four rotations, one horizontal movement, one vertical movement, and two diagonal movements. The gray-level transformation transforms the contracted domain block into an approximate range block by $\phi(t) = \alpha t + \beta$, where α is the contrast factor and β is the luminance factor. The adjustment parameters can be determined by minimizing the following collage error over R_i :

$$e_i = \|\alpha_i \cdot \phi_i^{(k)}(D_i) + \beta_i - R_i\|_2^2 = \|\omega_i(D_i) - R_i\|_2^2. \quad (5)$$

The parameters α_i and β_i can be solved by the least squares method and can be obtained as

$$\alpha_i = \frac{(1/b) \sum_{j=1}^b r_j d_j - ((1/b) \sum_{j=1}^b r_j)((1/b) \sum_{j=1}^b d_j)}{(1/b) \sum_{j=1}^b r_j^2 - ((1/b) \sum_{j=1}^b d_j)^2}, \quad (6)$$

$$\beta_i = \frac{1}{b} \sum_{j=1}^b r_j - \alpha_i \left(\frac{1}{b} \sum_{j=1}^b d_j \right),$$

where r_j and d_j are the j th pixel values of R_i and D_i , respectively, and b represents the number of all pixels in a range block. To obtain the best match domain block, we use an exhaustive way to search. Therefore, we obtained the fractal code $W_i = \{\alpha_i, \beta_i, k, i\}$ of the range block R_i , where k is the optimal parameter of the geometric mapping and i is the position index of the optimal domain block. The set of fractal codes for all range blocks is called the iterative function system for a given image F .

2.2. Fractal Image Decoding. Fractal decoding is a process of generating the original image's attractor by using the fractal information of the original image and the initial image. In the fractal decoding stage, we can select any initial image ξ as

the initial image, which is generally a blank image. Then, we use the iterative function for decoding until it converges to approximate the original image, as shown in equation (2). Since the similarity between ξ and the original image is often small, there is an error between the decoded image and the original image. When enlarging the image F by the scale factor h , we need to scale the initial image to the size of the image to be reconstructed. It is worth noting that range blocks and domain blocks also need to be scaled using the same fractal code.

3. Proposed Algorithm

In this section, we discuss the details of the proposed algorithm. The proposed algorithm is carried out in the framework of image fractal coding. The flowchart is shown in Figure 2. It is mainly divided into the following steps: first, fractal coding is performed on the LR image, the corresponding fractal code is determined, and the similarity relationship between the range block and the domain block is established. Then, according to the characteristics that the fractal code is independent of the image resolution, the attractor is reconstructed by super-resolution fractal decoding, and then the interpolated image is obtained. Finally, the fractal coded error image is input into the depth residual network for upsampling estimation and the HR error image is used as a compensation term to correct the interpolated image, obtaining the final HR image.

3.1. Quadtree Fractal Coding. The collage error of the traditional fractal coding is shown by equation (2). To reduce the error, a quadtree fractal coding strategy with variable block size can be adopted. The basic idea is to first use a larger size range block to search for each optimal domain block. If the corresponding tile error is greater than a given threshold, the range block is decomposed into four smaller blocks R_a , R_b , R_c , and R_d , which have a transformed relationship with the corresponding domain blocks D_a , D_b , D_c , and D_d , respectively, as shown in Figure 3. Each block is then researched recursively for optimal domain blocks and encoded until all image blocks have been encoded.

3.2. Super-Resolution Fractal Decoding. By fractal coding the LR image, we obtained the transformation relationship between the blocks R_i and D_i with similarity in the image. This similarity between such blocks is independent of the resolution of the image, so when the original LR image F is magnified to h times, the similarity between R_i and D_i remains unchanged. The image that is magnified by h times is recorded as F^h , so it satisfies

$$I(F) = F^h = \bigcup_{i=1}^{N_r} R_i^h = \bigcup_{i=1}^{N_r} \omega_i(D_i^h), \quad (7)$$

where I is the interpolation operator, R_i^h and D_i^h are the corresponding blocks of R_i and D_i in the HR image F^h , and the sizes are $nh \times nh$ and $mh \times mh$, respectively.

Super-resolution fractal decoding is similar to fractal decoding. Given an arbitrary initial image ξ^h of the same size

as F^h , the initial image is iteratively iterated by the following equation. The fractal interpolation estimate F_ω^h of the F image can be approximated by L iterations:

$$F^h \approx F_\omega^h = \lim_{L \rightarrow \infty} \omega^L(\xi^h). \quad (8)$$

The decoding process of the above equation is called the super-resolution fractal decoding process. Fractal image coding is a lossy coding because natural images do not have a strict local self-similar structure. Thus, there is an error E between the interpolated image F_ω and the original image F , which satisfies

$$F = F_\omega + E. \quad (9)$$

Obviously, F_ω is a fractal part with strict local self-similarity in F and E is the residual part (error image). To obtain an interpolated image with a magnification of h times, an interpolation operator is applied to both ends of equation (9) to obtain

$$F^h = I(F_\omega + E) = I(F_\omega) + I(E) = F_\omega^h + E^h, \quad (10)$$

where E^h is the error between the HR image F^h and the interpolated image F_ω^h . The smaller the error, the more accurate the reconstruction accuracy.

3.3. Error Image Estimation for CNN. To effectively estimate E^h , the traditional method uses a bicubic interpolation algorithm. There are artifacts and blurring that seriously affect the final reconstruction effect. To solve this problem, we propose a CNN-based method to estimate the error image. To reduce the interpolation error, the error estimate $\hat{E}^h$ is finally added as a compensation term to F_ω^h to obtain a better estimate of the HR image, i.e., $\hat{F}^h = F_\omega^h + \hat{E}^h$.

3.3.1. Network Structure. The proposed method is based on fractal technology to achieve SR image reconstruction. To better estimate the estimation error compensation term more accurately, the network architecture with depth residual is used to estimate. ResNet, the residual learning framework, was first proposed by He et al. [30] to solve the phenomenon of gradient explosion or gradient disappearance [31] due to the increase in network depth. Because of the fast connection of the residual network, the data transmission between the networks is smoother and the underfitting phenomenon caused by the disappearance of the gradient is also improved. Residual networks are heavily applied to reconstruction work, for example, VDSR [19], SRResNet [32], EDSR [20], and RDN [33], through residual learning to increase the depth of the network, improving the effect of image reconstruction. Based on residual learning, we propose a network architecture for estimating error images, as shown in Figure 4. The figure shows a collection architecture with deep and shallow networks with common inputs. A two-way network structure increases the width of the network, can capture different effective features, and improves reconstruction performance. Residual learning in deep networks increases the convergence speed while increasing the depth of the network, effectively retaining the

FIGURE 2: The proposed algorithm flowchart. The input LR image obtains the reconstructed HR image in three steps.

FIGURE 3: Block mapping of quadtree coding.

FIGURE 4: Network architecture diagram of error estimation. It is jointly trained by a deep network and a shallow network. Each layer network includes a feature extraction phase and a reconstruction phase.

high-frequency characteristics of the network. The shallow network adopts a simple network structure, which improves the training difficulty and helps to restore more features of

the image. Through the joint training of the two networks, the detailed HR error image can be recovered more accurately and the reconstruction result can be improved.

The deep network includes a feature extraction phase and a reconstruction phase. The traditional feature extraction methods mostly use first-order and two-step methods to filter the input image, while the deep learning method does not need to manually design the filter but automatically learns from the training data. In the feature extraction stage, take the LR error image as input and forward it through a neural network as a series of feature maps. The feature extraction network consists of a convolutional layer and a plurality of residual blocks. First, we use a convolutional layer to learn shallow features and join the rectified linear unit (ReLU) for nonlinear mapping. The convolutional layer can be expressed as

$$L_0 = \max(0, w_0 * E + g_0), \quad (11)$$

where w_0 represents the weight of the convolutional layer, g_0 represents the bias of the layer, the arithmetic symbol $*$ represents the convolution operation, E is the input LR error image, and finally, the feature map of the layer is obtained by the ReLU function. The convolutional layer generates a feature map of 64 channels.

Then, the shallow feature L_0 is used as input to multiple residual blocks to learn more high-level features. Each residual block consists of two convolutions with the same kernel size and number of filters. The residual structure is shown in Figure 5. The residual structure is a connection to a standard feedforward convolutional network with a jump around some layers. Each time a layer is bypassed, a residual block is generated, and each layer learns the residual function instead of the original function by inputting the image. The output is composed of the input and the result of two successive convolutions. Each residual block is used as a unit, and the output of each unit is passed to the next unit. Assuming that there are I residual blocks, the i th residual block can be obtained by equation (12), i.e.,

$$L_i = O_i(L_{i-1}) = O_i(O_{i-1}(\cdots (O_1(L_0)))), \quad (12)$$

where Q_i is the i th residual block operation, equivalent to the convolution operation and the ReLU operation in equation (11). L_i is the i th residual block, generated entirely by each convolutional layer within the block, and the previous layer can access subsequent layers.

As the number of deep feature maps of the network increases, to reduce the number of feature maps, the outputs of all units are input to the fusion layer of the convolution kernel as 1×1 to control the output information. The introduction of residual blocks improves the convergence speed of the network, enhances the flow between information, and extracts more image features. Finally, a convolutional layer is added whose output is the input to the reconstruction module. All convolutional layers have the same convolution kernel size 3×3 . To ensure that the size of the feature map is the same as the size of the input, we set both the stride and the edge pad to 1.

The image reconstruction stage uses pixel shuffle [34] (also known as subpixel convolution) for upsampling of image resolution, which is to enlarge to the size of the HR

FIGURE 5: ResBlock architecture consists of two Conv layers and one ReLU layer.

space. Compared to deconvolutional layers (such as FSRCNN [35]), it introduces fewer checkerboard artifacts [36]. Additionally, our upsampling method does not insert one or more convolutional layers after upsampling, as in the previous technique [20, 33], but instead directly uses an upsampling layer, which would not affect the rebuild performance but would increase the speed significantly. Unlike a conventional convolutional layer, the number of characteristic channels it outputs is r^2 , where r is the scale factor.

The shallow network also includes feature extraction and reconstruction phases. EDSR, RDN, etc. use the global residual path to effectively extract shallow and deep information, but this global residual path is a linear stack of several convolutional layers, which is computationally intensive. Yu et al. [37] proves that those linear convolutions are redundant and can be absorbed to some extent in the residual body. Therefore, we combine the idea of [37] and use two networks to jointly train. A shallow Conv layer with a convolution kernel size of 5×5 is used, and the number of filters is 64. Then, directly input into the pixel shuffle layer for upsampling to obtain the reconstructed image.

3.3.2. Training. The deep network and the shallow network are independent, and the two do not affect each other. The LR image is used as input for SR reconstruction. The final HR error image is obtained by equation (13), i.e.,

$$\hat{E}^h = H_D(E, \theta_D) + H_S(E, \theta_S), \quad (13)$$

where E represents the input LR error image, $H_D(\cdot, \theta_D)$ and $H_S(\cdot, \theta_S)$ represent the HR output of the deep and shallow networks parameterized by θ_D and θ_S , respectively, and $\hat{E}^h$ is the final HR error image jointly trained.

Given M training image pairs $\{E^{(i)}, E_h^{(i)}\}_{i=1}^M$, where E represents the input LR error image and $E_h^{(i)}$ represents the corresponding HR error image. Our network is jointly trained by minimizing the Euclidean loss of the predicted HR image $\hat{E}^h$ and the real HR error image E^h , i.e.,

$$\theta_D^*, \theta_S^* = \arg \min_{\theta_D, \theta_S} \frac{1}{2M} \sum_{i=1}^M \|\hat{E}_h^{(i)} - E^{(i)}\|_2^2 + \lambda \Phi(\theta_D, \theta_S), \quad (14)$$

where $\Phi(\theta_D, \theta_S)$ represents the weight attenuation applied to the network parameters and λ represents the compromise parameter.

In summary, the specific steps of the SR method based on fractal interpolation and residual network are as follows (see Algorithm 1).

Input: LR image: F , threshold: ϑ , scale factor: h , N_{start} , N_{stop} .

Output: Reconstructed image: $\hat{F}^h$.

- (1) Construct range blocks for LR images $F = \cup_{i=1}^{N_r} R_i$, each range block size is $N_{\text{start}} \times N_{\text{start}}$, $R_i \cap R_j = \emptyset$;
- (2) Mark all R_i as “unencoded” and add them to the encoding queue C ;
- (3) **for** $i = 1$ to N_r **do**
- (4) Take an “uncoded” R_i from C , size is $N \times N$, exhaustively find the optimal domain block D_i in all image blocks of size $2N \times 2N$ and make it the smallest patch error e under the approximation of equation (5);
- (5) **if** $e \leq \vartheta$ and $N \times N \leq N_{\text{stop}} \times N_{\text{stop}}$ **do**
- (6) Record the size of the range block and the fractal code W ;
- (7) Mark as “encoded”;
- (8) **else**
- (9) Decompose R_i into four smaller sub-blocks R_a , R_b , R_c , and R_d , all marked as “uncoded” and added to C ;
- (10) **end if**
- (11) Delete the original sub-block R_i from C ;
- (12) **end for**
- (13) Fractal decode the image F according to the fractal code W , and calculate the error E between the decoded image and the original image;
- (14) Multiply the coordinates and size of the upper left corner of all range blocks and domain blocks by h times in queue C to obtain a new range R' and domain block D'_i corresponding to each range block R'_i ;
- (15) Optionally, magnify an initial image of the same size as F by h times and perform super-resolution decoding by equation (6) to obtain fractal interpolation estimation F_ω^h of image F ;
- (16) Input the error image E into the depth residual network for h -fold reconstruction to obtain an error compensation term $\hat{E}^h$;
- (17) Obtain the best estimate of the reconstructed image: $\hat{F}^h = F_\omega^h + \hat{E}^h$.

ALGORITHM 1: SR using fractal coding and residual network.

4. Experiments

4.1. Datasets. To demonstrate the effectiveness of the proposed method, quantitative and qualitative comparisons were made between different scale factors. In the training depth residual network, the external dataset DIV2K [38] issued by Timofte et al. was used. The DIV2K dataset consists of 800 training images, 100 validation images, and 100 test images. Since we use the depth residual network to estimate the error image, the datasets used by the training network are the fractal coded error image. During the experiment, we performed fractal coding processing on 800 training images. The error image pairs of the HR and LR image pairs are used as training samples, and 6 verification images are selected for verification. We used four standard benchmark datasets: Set5 [39], Set14 [40], B100 [41], and Urban100 [7] to compare the performance.

4.2. Parameter Settings. The experimental environment includes hardware devices and software configurations. The test computer is configured as an Intel Core i7-5820K CPU@ 3.30 GHz x12, a NVIDIA GeForce TITAN X GPU, 16 GB of RAM, a Win10 operating system, and the compile software is MATLAB 2018a.

In the process of training the residual network, Caffe [42] was used to build the network model and optimized by the Adam optimizer. The batch size was set to 256, the momentum was 0.9, and the weight decay parameters were 0.009 (L_2 penalty multiplied by 0.009). For weight initialization, we refer to the method proposed by He et al. [43]. The filter of the convolutional layer in the network was randomly initialized by a zero-mean Gaussian distribution with a

standard deviation of 0.01. The filter in the deconvolutional layer was initialized by a bilinear interpolation kernel. The initial learning rate for all layers was set to $1e-3$, which was reduced by half for every 100 epochs. The time required to train the network was approximately 7 hours.

For each scale factor, the HR error image was randomly cropped from the training image into image blocks of size 96×96 , and then the HR image was generated using the bicubic interpolation algorithm to generate corresponding LR training samples. To avoid overfitting and further improve accuracy, the dataset was amplified by horizontal, vertical flip and 90° rotation. Since humans are more sensitive to detailed changes than colors, as with other methods, we also applied our algorithm to the luminance channel in the YCbCr color space and bicubic interpolation to the other two color channels.

The parameters in the fractal coding algorithm were: $N_{\text{start}} = 16$, $N_{\text{stop}} = 4$, and $\vartheta = 100$. As shown in Table 1, the deeper the network, the better the performance. As the number of residual blocks and the number of convolutional layers increased, better performance was easily obtained. Therefore, designing a deep network into a deeper, broader network allows for more layered features. The number of residual blocks was set to 10, and each convolutional layer produced 64 feature maps.

4.3. Comparison with State-of-The-Art Models. To verify the validity of the proposed algorithm, we compared our method with the six state-of-the-art methods qualitatively and quantitatively with three different scale factors (2, 3, and 4). The methods compared include two traditional learning methods: A+ [15] and SelfexSR [21] and five deep learning

TABLE 1: Choice of depth at each level.

Deep	B100		Urban100	
	PSNR (dB)	Time (s)	PSNR (dB)	Time (s)
3	31.46	0.053	31.48	0.058
5	31.52	0.061	31.58	0.063
8	31.61	0.063	31.63	0.065
10	31.69	0.073	31.72	0.072

methods: SRCNN [16], VDSR [17], EDSR [18], RDN [31], and PASSR [44].

To effectively evaluate the benefits and drawbacks of the reconstruction effect, the peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) were used to measure the image quality. PSNR is a full reference evaluation method. The larger the value of the PSNR, the better the reconstruction effect of the image. SSIM refers to the similarity of structural information between two images and is also a full reference image quality evaluation index, which measures image similarity from three aspects of brightness, contrast, and structure. The SSIM value range [0, 1], and the larger the value, the better the reconstruction effect:

$$\text{PSNR} = 10 \lg \frac{\text{MAX}^2 PQ}{\|y - \hat{y}\|^2},$$

$$\text{SSIM} = \frac{(2\mu_y \mu_{\hat{y}} + C_1)(\sigma_{\hat{y}y} + C_2)}{(\mu_y^2 + \mu_{\hat{y}}^2 + C_1)(\sigma_y^2 + \sigma_{\hat{y}}^2 + C_2)}, \quad (15)$$

where MAX represents the gray level of the image, generally 255; P and Q are the width and height of the image, respectively; y is the original HR image, $\hat{y}$ is the reconstructed HR image; μ_y and σ_y are the average gray value and variance in the original HR image, respectively; $\mu_{\hat{y}}$ and $\sigma_{\hat{y}}$ are the grayscale mean and variance in the reconstructed image, respectively; $\sigma_{\hat{y}y}$ is the covariance of the original HR image and the reconstructed HR image; and C_1 and C_2 are constant.

Table 2 shows the quantitative comparison of all methods for scale factors of 2, 3, and 4 under four datasets. The effect of each method on SR is summarized by the average PSNR and SSIM. During the reconstruction process, the original LR image of any size is taken as input, and the corresponding HR image is directly reconstructed. We experiment with the code provided by the authors of these methods.

It can be seen from the table that the PSNR and SSIM of the proposed method achieved the best results in the comparison method. Bicubic uses a simple interpolation method and did not achieve good performance. The A+ algorithm uses the method of anchor domain regression, and the reconstruction effect is improved. The PSNR is increased by 1~2 dB compared with bicubic, and the SSIM is improved by approximately 0.03~0.06. For the first time, the SRCNN introduced deep learning into the SR field, which showed superior performance compared to traditional methods such as bicubic, A+, and SelfExSR. Compared with

A+, PSNR increased by approximately 0.1~0.2 dB. However, SelfExSR was better than the SRCNN in the Urban100 dataset, which uses the self-similarity of the image. Therefore, it is better for Urban100, which mostly contains self-similar structures. VDSR, EDSR, RDN, and PASSR all introduce residual learning, which makes the structure of the network deeper, learns more rich features, and achieves good results. RDN uses a residual-dense block (RDB) to retain a rich set of features, and when the scale factor was 2, it performed best on most datasets. However, when the scale factor was larger (such as 3 and 4), RDN did not have an advantage. PASSR is an SR reconstruction method for stereoscopic images that uses stereo image datasets for training. Therefore, its PSNR and SSIM are optimal on the dataset of Urban100, which contains a large number of stereo images.

As the scale factor increased, the performances of all methods decreased, but our method always maintained the best result when the scale factor was 4, which shows that our method shows better performance for large-scale factors because we use fractal coding to obtain more image information with less data than other methods. Moreover, for Urban100, which mostly contains self-similar structures, the use of larger training layers and image blocks can make better use of the receptive field to obtain more information. Although the proposed method did not show advantages when magnified 2 times, it showed pleasing results for large-scale factors. In general, our method is still competitive compared to state-of-the-art methods.

PSNR and SSIM only evaluate the difference between image pixels, and the quality of reconstruction cannot be fully evaluated. Therefore, based on the quantitative comparison, we combined the visual effects to qualitatively analyze the image quality. Figures 6–10 show qualitative comparisons of this paper with other advanced methods.

Figures 6–10 show qualitative comparisons of our method with state-of-the-art methods. Figures 6 and 10 are diagrams of the effects of various methods at a scale factor of 2. We tested the texture retention of “sculptures,” “text,” and “windows”. It can be seen from this that the proposed algorithm is visually competitive and reconstructs more textures. Bicubic is based on the principle of the smoothing hypothesis, the image is very blurred and the reconstruction is not good. Although the A+ algorithm has been improved, there is still a blurring phenomenon. SelfExSR, SRCNN, and VDSR have little difference in the reconstruction of “sculpture” and “text”. However, as can be clearly seen from Figure 10, the texture of the SRCNN image has significant distortion. SelfExSR is slightly better than the SRCNN, but

TABLE 2: Average PSNR/SSIM for scale factor 2, 3, and 4 on datasets Set5, Set14, B100, and Urban100. Bold indicates the best performance.

Algorithm	Scale	Set5		Set14		B100		Urban100	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Bicubic	2	33.66	0.9299	31.24	0.8688	29.56	0.8431	26.88	0.8403
A+		36.54	0.9544	32.28	0.9056	31.21	0.8863	29.20	0.8938
SRCNN		36.66	0.9542	32.42	0.9063	31.36	0.8879	29.50	0.8946
SelfexSR		36.49	0.9537	32.22	0.9034	31.18	0.8855	29.54	0.8967
VDSR		37.53	0.9587	33.03	0.9124	31.90	0.8960	30.76	0.9140
EDSR		38.11	0.9602	33.92	0.9195	32.32	0.9013	32.93	0.9351
RDN		38.24	0.9614	34.01	0.9212	32.34	0.9017	32.89	0.9353
PASSR		38.20	0.9612	33.99	0.9214	32.35	0.9020	32.96	0.9355
Ours		38.16	0.9611	33.98	0.9210	32.31	0.9013	32.96	0.9353
Bicubic	3	30.39	0.8682	27.55	0.7742	27.21	0.7349	24.46	0.7349
A+		32.58	0.9088	29.13	0.8188	28.29	0.7835	26.03	0.7973
SRCNN		32.75	0.9090	29.28	0.8209	28.41	0.7863	26.24	0.7989
SelfexSR		32.58	0.9093	29.16	0.8196	28.29	0.7840	26.44	0.8088
VDSR		33.66	0.9213	29.77	0.8314	28.82	0.7976	27.14	0.8279
EDSR		34.66	0.9280	30.52	0.8462	29.25	0.8093	28.80	0.8653
RDN		34.70	0.9283	30.57	0.8468	29.26	0.8093	28.80	0.8653
PASSR		—	—	—	—	—	—	—	—
Ours		34.70	0.9288	30.57	0.8472	29.26	0.8095	28.81	0.8655
Bicubic	4	28.42	0.8104	26.00	0.7027	25.96	0.6675	23.14	0.6577
A+		30.28	0.8603	27.32	0.7491	26.82	0.7087	24.32	0.7183
SRCNN		30.48	0.8628	27.49	0.7503	26.90	0.7101	24.52	0.7221
SelfexSR		30.31	0.8619	27.40	0.7518	26.84	0.7106	24.79	0.7374
VDSR		31.35	0.8838	28.01	0.7674	27.29	0.7251	25.18	0.7524
EDSR		32.46	0.8968	28.80	0.7876	27.71	0.7420	26.64	0.8033
RDN		32.47	0.8990	28.81	0.7871	27.72	0.7419	26.61	0.8028
PASSR		32.46	0.8988	28.82	0.7873	27.68	0.7419	26.57	26.57
Ours		32.48	0.8973	28.84	0.7878	27.72	0.7423	26.64	0.8031

both have artifacts at the edges. VDSR works best, with a more complete line texture. EDSR reconstructed more details in terms of “sculpture” and “text,” with a clearer effect. However, the line texture of the window did not achieve the same effect as VDSR. The proposed algorithm is slightly better than EDSR, more details are extracted, and the distortion is improved.

Figures 7 and 8 are visual representations with a scale factor of 3. The reconstruction performance of all algorithms is degraded. In Figure 7, A+ appears jagged and the SRCNN exhibits more distortion and edge blurring. This phenomenon improved with the increase in network depth in VDSR. Since both the SRCNN and VDSR are preprocessed using bicubic interpolation, significant artifacts are produced. The pattern on the “clothes” in SelfExSR is clearer, but the effect of the collar is not good. The EDSR and the proposed algorithm reconstruct the obvious edge details, which can prove the superiority of the deep network. In Figure 8, neither the bicubic nor the A+ method reconstructed the “baby’s” eyebrows. EDSR reconstructed a more complete eyebrow line, and the proposed algorithm has more eyebrow texture. In the tail of the “fish,” Bicubic, A+, SRCNN, SelfExSR, and RDN have different degrees of artifacts, while the effects of VDSR and Ours are more complete and clearer. For complex textures such as “coral,” our algorithm shows more fine edge information.

Figure 9 shows the effect diagrams with a scale factor of 4. With the increase in the scale factor, the reconstruction

ability of the algorithm is greatly reduced. Compared with the original image, the gap is larger and larger, and edge blur and artifacts are more likely to occur. The effect between the SRCNN and VDSR is not much different. The “wall,” “cactus,” and hair of “baboon” are blurred, the edge details are incomplete, and the texture is distorted. The “baboon” image generated by the SelfExSR method has a significant block effect because it utilizes the self-similarity feature inside the image block to group the image blocks. Images with more complex textures, such as img092, Bicubic, A+, SRCNN, and SelfExSR, are more severely jagged. The PASSR is generally clearer than the VDSR, but the textures on the walls are distorted and artifacts. Although our method also divides the image by fractal technology, it uses the deep network to effectively compensate for the error. Therefore, it has achieved considerable visual effects. For example, the thorns on the edge of the “cactus” are better presented, and the “baboon’s” hair not only has no blockiness but is also more complete. In img092, the texture is more realistic and not jagged.

In general, the bicubic method uses a simple interpolation algorithm, and the reconstruction effect is very blurry and smooth. A+ combines sparse representation and anchoring methods to calculate the correlation of dictionary atoms. Although the effect is better than the traditional sparse representation method, it still appears jagged. The SRCNN uses a simple convolutional neural network to improve compared to the traditional method, but the

FIGURE 6: Super-resolution results for scale factor 2. The two pictures show the SR results of the “sculpture” and “ppt,” respectively. Partially enlarged view of the picture in the red box on the right.

FIGURE 7: Super-resolution results of “img005” from Set14 with scale factor 3. Partially enlarged view of the picture in the red box on the right.

individual images still cannot match the SelfExSR method using the internal example. The VDSR method uses a global residual method to deepen the network structure, which is

slightly better than the SRCNN. However, there are artifacts in the edge details, and the texture details cannot be reproduced well. EDSR uses a deeper network structure to

FIGURE 8: Super-resolution results for scale factor 3. Below each image is a partial enlargement.

learn more image features and reduce edge blurring. RDN uses RDB blocks to fully extract the layered features of the image and restore more textures. PASSR introduces the parallax attention mechanism of the global receiving field, which is convenient for processing stereoscopic images with large visual changes, so it has certain limitations in the application range. The proposed method is based on the advantages of fractal technology in texture description and uses fractal super-resolution technology for magnification reconstruction. Because of the blockiness that may occur due to the self-similarity characteristics inside the image, the error caused by fractal coding is compensated by constructing a deep residual network. The reconstruction performance is greatly improved and more texture details are presented, making the estimated image more realistic and complete.

4.4. Ablation Analysis. To further understand the contribution of the proposed algorithm, we evaluate the different components in the model through experiments. The parameter factors in the experiment are the same as those in Section 4.2. Since the results of different scale factors are similar, only the result of a scale factor of 3 is reported, and the dataset is the Set14.

The traditional fractal-based SR method generally uses a bicubic interpolation algorithm for estimating error compensation. From the analysis of Section 4.3, it can be seen that the bicubic algorithm adopts the principle based on smoothing and cannot effectively estimate a more complete error image. Similarly, if the estimated error image is not effective, the ideal reconstructed image cannot be obtained when it is added to the fractal interpolation image. Table 3 shows the average PSNR obtained using the traditional

bicubic method and the depth residual network. It can be seen that reconstructing the error image by constructing the residual network greatly improves the reconstruction quality.

To prove the validity of the parallel structure in the deep residual network, we split the network into a deep network and a shallow network. Table 3 shows an ablation study of different components on the Set14 dataset with scale factor 3. The joint training of the two networks increases the PSNR by 0.18 dB. Because the two-network joint training is equivalent to widening the network, increases the parameters of the network and the number of features, so that the network learns more different features, which can effectively improve the visual effect of reconstruction.

To prove the advantage of ResBlocks, which are deleted to observe its effect, in the experiment, all ResBlocks were replaced with the same number of Conv, and the subsequent fusion layers were removed. The result of the last convolutional layer is then fed directly into the reconstruction network to obtain the final prediction. As shown in Table 3, after the ResBlock is removed, the SR reconstruction performance is degraded because residual learning reduces the weight of the network, improves the information flow and gradient, greatly speeds up the convergence, and achieves good performance.

4.5. Evaluation in Running Time. Finally, we compare the running time with the state-of-the-art methods on the same machine. We use these algorithms to super-resolution 100 images in Urban100 and then record the average PSNR and consumption time of each algorithm. As shown in Figure 11, the X coordinate represents the running time and the Y coordinate represents the average PSNR. The average PSNR of our algorithm is the largest, and its consumption time is at a medium level, which is smaller than EDSR, RDN, SelfexSR, and A+.

FIGURE 9: Super-resolution results for scale factor 4. Below each image is a partial enlargement.

FIGURE 10: Super-resolution results for scale factor 2. Below each image is a partial enlargement.

TABLE 3: Ablation study of the proposed component in reconstruction quality. Bold indicates the best performance.

Bicubic	Deep network	Shallow network	ResBlock	PSNR
✓	×	×	×	29.74
×	✓	×	×	30.13
×	✓	×	✓	30.39
×	✓	✓	×	30.2
×	✓	✓	✓	30.57

FIGURE 11: Runtime versus performance. The results are evaluated on the Urban100 dataset for scale factor 4. The proposed strides a balance between reconstruction accuracy and execution time.

5. Conclusions

In this paper, we propose a super-resolution method based on fractal image interpolation and depth residual network, which can effectively recover texture structure details, improve edge blur and distortion problems, and have high SR reconstruction performance. First, the image is fractal coded to establish the similarity between the range block and the domain block. Then, we use the similarity relationship to SR reconstruction of the image with SR fractal decoding. In the fractal decoding process, since the fractal image encoding is lossy compression, there is an error between the decoded image and the original image. To estimate the error compensation term more accurately, the residual network is used to train the mapping relationship between high- and low-resolution error images. In future work, the reconstruction accuracy of the interpolation method will be further improved.

Data Availability

All data in this article are derived from publicly available datasets on the Internet.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

The authors acknowledge the National Natural Science Foundation of China (Grant: 61772319, 61976125, 61976124, 61773244, and 61873177), Shandong Natural Science Foundation of China (Grant: ZR2017MF049), and Yantai Key Research and Development Program of China (Grant: 2017ZH065 and 2019XDHZ081).

References

- [1] S. Baker and T. Kanade, "Limits on super-resolution and how to break them," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 9, pp. 1167–1183, 2002.
- [2] D. Zhou, W. Dong, and W. Shen, "Image zooming using directional cubic convolution interpolation," *IET Image Processing*, vol. 6, no. 6, pp. 627–634, 2012.
- [3] Y. Zhang, Q. Fan, F. Bao, Y. Liu, and C. Zhang, "Single-image super-resolution based on rational fractal interpolation," *IEEE Transactions on Image Processing*, vol. 27, no. 8, pp. 3782–3797, 2018.
- [4] Y. Tai, S. Liu, M. S. Brown, and S. Lin, "Super resolution using edge prior and single image detail synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2400–2407, San Francisco, CA, USA, May 2010.
- [5] J. Sun, Z. Xu, and H. Shum, "Gradient profile prior and its applications in image super-resolution and enhancement," *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1529–1542, 2011.
- [6] G. Freedman and R. Fattal, "Image and video upscaling from local self-examples," *ACM Transactions on Graphics*, vol. 30, no. 2, pp. 1–11, 2011.
- [7] J. B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 5197–5206, Boston, MA, USA, June 2015.
- [8] Z. Y. Wang, Y. Z. Yang, Z. W. Wang et al., "Self-tuned deep super resolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1–8, Boston, MA, USA, June 2015.
- [9] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael, "Learning low-level vision," *International Journal of Computer Vision*, vol. 40, no. 1, pp. 25–47, 2000.
- [10] H. Chang, Y. M. Xiong, and D. Y. Yeung, "Super-resolution through neighbor embedding," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 275–282, Washington, DC, USA, June–July 2004.
- [11] J. H. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Transactions on Image Processing*, vol. 19, no. 11, pp. 2861–2873, 2010.
- [12] L. Fang, H. Zhuo, and S. Li, "Super-resolution of hyperspectral image via superpixel-based sparse representation," *Neurocomputing*, vol. 273, pp. 171–177, 2018.
- [13] C. Y. Yang and M. H. Yang, "Fast direct super-resolution by simple functions," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 561–568, Sydney, NSW, Australia, December 2013.
- [14] R. Timofte, V. De, and G. L. Van, "Anchored neighborhood regression for fast example-based super-resolution," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1920–1927, Sydney, NSW, Australia, December 2013.
- [15] R. Timofte, V. D. Smet, and L. V. Goo, "A+: adjusted anchored neighborhood regression for fast super-resolution," in *Computer Vision—ACCV 2014*, pp. 111–126, Springer, Cham, Switzerland, 2014.
- [16] J. J. Li, G. J. Yuan, and H. Fan, "Multifocus image fusion using wavelet-domain-based deep CNN," *Computational Intelligence and Neuroscience*, vol. 2019, Article ID 4179397, 23 pages, 2019.
- [17] J. Li, G. Li, and H. Fan, "Image dehazing using residual-based deep CNN," *IEEE Access*, vol. 6, pp. 26831–26842, 2018.
- [18] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Computer Vision—ECCV 2014*, pp. 184–199, Springer, Cham, Switzerland, 2014.

- [19] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1646–1654, Las Vegas, NV, USA, June 2016.
- [20] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the Computer Vision and Pattern Recognition Workshops*, pp. 1132–1140, Honolulu, HI, USA, July 2017.
- [21] Y. HaCohen, R. Fattal, and D. Lischinski, "Image upsampling via texture hallucination," in *Proceedings of the IEEE International Conference on Computational Photography*, pp. 1–8, Cambridge, MA, USA, March 2010.
- [22] W. Dong, L. Zhang, R. Lukac, and G. Shi, "Sparse representation based image interpolation with nonlocal autoregressive modeling," *IEEE Transactions on Image Processing*, vol. 22, no. 4, pp. 1382–1394, 2013.
- [23] L. Wei, L. Xizhe, Y. Zhengming et al., "A new improved threshold segmentation method for scanning images of reservoir rocks considering pore fractal characteristics," *Fractals*, vol. 26, no. 2, Article ID 1840003, 2018.
- [24] Y. Li, S. Zhang, and X. Zhang, "Classification and fractal characteristics of coal rock fragments under uniaxial cyclic loading conditions," *Arabian Journal of Geosciences*, vol. 11, no. 9, p. 201, 2018.
- [25] W. Xie, J. Liu, L. Shao, and F. Jing, "Multi-scale fractal coding for single image super-resolution," *Intelligent Computing Theory*, vol. 8588, pp. 425–434, 2014.
- [26] Y. Wee and H. Shin, "A novel fast fractal super resolution technique," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 3, pp. 1537–1541, 2010.
- [27] H. Xu, G. Zhai, and X. Yang, "Single image super-resolution with detail enhancement based on local fractal analysis of gradient," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 10, pp. 1740–1754, 2013.
- [28] L. Yu, Y. Xu, H. Xu, and X. Yang, "Self-example based super-resolution with fractal-based gradient enhancement," in *Proceedings of the IEEE International Conference on Multimedia and Expo Workshops*, pp. 1–6, San Jose, CA, USA, July 2013.
- [29] X. Yao, Q. Wu, P. Zhang, and F. Bao, "Adaptive rational fractal interpolation function for image super-resolution via local fractal analysis," *Image and Vision Computing*, vol. 82, pp. 39–49, 2019.
- [30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, Las Vegas, NV, USA, June 2015.
- [31] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994.
- [32] C. Ledig, L. Theis, F. Huszár et al., "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 105–114, Honolulu, HI, USA, July 2017.
- [33] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," 2018, <https://arxiv.org/abs/1802.08797>.
- [34] W. Shi, J. Caballero, F. Huszár et al., "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1874–1883, Las Vegas, NV, USA, June 2016.
- [35] D. Chao, C. L. Chen, and T. Xiaoou, "Accelerating the super-resolution convolutional neural network," in *European Conference on Computer Vision*, pp. 391–407, Springer, Cham, Switzerland, 2016.
- [36] O. Augustus, D. Vincent, and O. Chris, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, 2016.
- [37] J. Yu, Y. Fan, J. Yang et al., "Wide activation for efficient and accurate image super-resolution," 2018, <https://arxiv.org/abs/1808.08718>.
- [38] R. Timofte, E. Agustsson, L. V. Gool et al., "NTIRE 2017 challenge on single image super-resolution: methods and results," in *Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1110–1121, Honolulu, HI, USA, August 2017.
- [39] M. Bevilacqua, A. Roumy, C. Guillemot et al., "Low-complexity single image super-resolution based on nonnegative neighbor embedding," in *Proceedings of the British Machine Vision Conference*, pp. 135.1–135.10, Guildford, UK, May 2012.
- [40] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *Curves and Surfaces*, pp. 711–730, Springer, Berlin, Germany, 2010.
- [41] D. R. 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 *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1110–1121, Vancouver, BC, Canada, July 2002.
- [42] Y. Jia, E. Shelhamer, J. Donahue et al., "Caffe: convolutional architecture for fast feature embedding," 2014, <https://arxiv.org/abs/1408.5093>.
- [43] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: surpassing human-level performance on ImageNet classification," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1026–1034, Santiago, Chile, December 2015.
- [44] L. Wang, Y. Wang, Z. Liang et al., "Learning parallax attention for stereo image super-resolution," 2019, <https://arxiv.org/abs/1903.05784>.

Submit your manuscripts at
www.hindawi.com