Research Article

An Improved Feedback Network Superresolution on Camera Lens Images for Blind Superresolution

Yuhao Liu

College of Information Engineering, Shanghai Maritime University, 1550 Haigang Ave., Shanghai 201306, China

Correspondence should be addressed to Yuhao Liu; liuyuhao@shmtu.edu.cn

Received 14 January 2021; Revised 2 March 2021; Accepted 6 March 2021; Published 20 March 2021

Academic Editor: Yang Li

Copyright © 2021 Yuhao Liu. 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.

Most of the recent advances in image superresolution (SR) assume that the blur kernel during downsampling is predefined (e.g., Bicubic or Gaussian kernel), but it is a difficult task to make it suitable for all the realistic images. In this paper, we propose an Improved Superresolution Feedback Network (ISRFN) which is designed free to predefine the downsampling blur kernel by dealing with real-world HR-LR image pairs directly without downsampling process. We propose ISRFN by modifying the layers and network structures of the famous Superresolution Feedback Network (SRFBN). We trained the ISRFN with the Camera Lens Database named City100, which produced the HR and LR on the same lens, respectively, free for downsampling, so our proposed ISRFN is free to estimate the blur kernel. Due to different camera lens (smartphone and DSLR) databases, we perform two series of experiments under two camera lenses-based City100 databases, respectively, to choose the optimum network structures; experiments make it clear that different camera lens-based databases have different optimum network structures. We also compare our two ISRFNs with the state-of-the-art algorithms on performance; experiments show that our proposed ISRFN outperforms other state-of-the-art algorithms.

1. Introduction

Single-Image Superresolution (SISR) aims to reconstruct a high-resolution image from a single low-resolution (LR) input image which has gained wide attention in many areas [1]. In general, this problem is very challenging and inherently ill-posed since there are always multiple high-resolution (HR) images corresponding to a single LR image [2]. In practice, we always generate a fake LR image from the given HR image with the blur kernel during downsampling which is predefined and then perform SR from fake LR images. This means that certain SR model which is trained under fixed downsampling methods can achieve better performance under the same downsampling methods based on valid data, but does not perform well on the realistic LR image if the downsampling blur kernel is incorrect. Mostly, in the real world, the blur kernels are unknown, so single downsampling blur kernel is always a misfit. This problem is called the Blind Superresolution problem which is a research hot point for SR.

Addressing Blind Superresolution problem, Gu et al. [3] proposed an Iterative Kernel Correction (IKC) method for blur kernel estimation in Blind Superresolution problem where the blur kernels are unknown; the proposed IKC method can correct inaccurate blur kernels. They further proposed the Spatial Feature Transform (SFT) layers-based SR network architecture named Spatial Feature Transform for Multiple Degradations (SFTMD); the IKC method with SFTMD can provide visually favorable SR results. Zhang et al. [4] proposed a deep plug-and-play framework to handle LR images with arbitrary blur kernels; they extended Bicubic degradation-based deep SISR to a new degradation method which can handle LR images with arbitrary blur kernels. Wang et al. [5] introduced kernel estimation mechanism along with dual back-projection under two networks. All above methods are forcing on estimating the blur kernel to simulate the real-world LR images.

Chen et al. [6] gave us a simple but effective way to solve the Blind Superresolution problem; they investigate super-resolution from the perspective of camera lenses and

proposed a new database named City100 database. City100 database was composed by HR-LR pairs for all the scenes. First, they captured each HR image on a tripod with a zoom lens, and then they zoom in the zoom lens to capture the LR image. At last, they align the realistic HR-LR image pair after rectification. Different from previous work, the LR images of City100 database are all taken from the real world, which have nature 'blur kernel' without downsampling; we can make use of City100 database without estimating the blur kernel. Because the LR image is real image which was shot in the same distance as the HR image, the LR image not only has the real blur kernel but also has the field-of-view problem in different focal length for a zoom lens; this means that SR from City100 database is a difficult task.

Another famous algorithm that focused on Blind Superresolution problem is Xu et al.'s method [7]; they proposed a dual convolutional neural network which works on both raw channel and 8-bit JPEG channel. The LR image was generated by simulating the imaging process of digital cameras from HR image. Xu's method is an excellent and common approach but neglects the difference character of different camera lens. So, in this paper, we chose Chen's [6] method to solve the Blind Superresolution problem. This method is effective, but we must notice the limitation that different camera lenses have different character, so we need to rebuild the model if the camera lens is changed.

In this paper, we propose Improved Superresolution Feedback Network (ISRFN) which is aimed at solving the Blind Superresolution problem and outperforms other state-of-the-art methods on performance (PSNR/SSIM) as illustrated in Table 1. Our contribution in this paper is listed as follows:

- (1) We propose the ISRFN to solve the Blind Superresolution problem. We absorb Chen's [6] framework and database, but our proposed ISRFN outperforms Chen's work (VDSR-based) in performance (PSNR/SSIM).
- (2) Our proposed ISRFN is based on the famous SRFBN [8] with some improvements to adapt the City100 database. Due to the same dimension of HR-LR for City100 database, the SRFBN cannot directly handle City100 database which needs improvement.
- (3) Our work makes it clear that different camera lens (smartphone or DSLR on City100 databases) has different optimum network structure. It is well known the deeper network structures can achieve better performance [9, 10] but costly so we achieve two different optimum network structures for smartphone and DSLR camera lens with the same total number of network parameters.

2. Relate Work

2.1. The Deep Convolution-Based SR Methods. In this section, we will give a brief introduction on the deep convolution layers-based SR methods. The Superresolution Convolutional Neural Network (SRCNN) [11] is a well-known SR method, which is famous as the first deep convolution

layers-based SR method. SRCNN directly learns an end-to-end mapping between the LR image and HR image; the input LR images need to be upsampled by Bicubic interpolation; then the upsampled LR images are fed into the convolutional network. The convolutional network model of SRCNN is simple with 3 convolutional layers and 2 ReLU layers between them; SRCNN chose Mean Squared Error (MSE) as the loss function. SRCNN is an excellent SR method, and many improvements are performed based on SRCNN.

The very deep convolutional networks [10] make use of 20 convolutional layers and 19 ReLU layers between them. By cascading small filters many times in a deep network structure, the contextual information extends to large regions. Due to its very deep network structure, VDSR introduce residuals to reduce the gradient vanishing problem and introduce gradient clipping to solve the gradient exploding problem. Farrugia [12] proposed an outstanding light field SR method, by introducing deep convolutional network into light field combined with low rank, illustrating that the convolutional network can achieve better performance if combined with other classic algorithms.

Residuals connection is critical for SR. The SR image can be achieved by adding the output of the model and the residuals (the LR images), where the residuals connection can transfer the low-frequency information of the LR image to the end of the model directly [13, 14], while the model only needs to generate the small amount of high-frequency information. There are two advantages for the residuals-based model: 1. the model only needs to generate a few of information, so the performance will be improved greatly. 2. Most features after applying rectified linear units (ReLU) are zero, so the model is easy to converge.

Feedback network is a famous network structure which is widely used in SR field. The advantage of feedback network is that the network can go very deep by increasing recursion depth without increasing the number of network parameters. For this, many feedback network-based algorithms were proposed. Kim et al. [15] introduced feedback network into SR proposing Deeply Recursive Convolutional Network (DRCN) for SR with 16 recursions, because learning a DRCN is very difficult due to exploding/vanishing gradients, so skip connection is introduced into DRCN. Jiang et al. [16] introduced hierarchical dense residual block into SR, proposing the hierarchical dense connection network (HDN) for SR.

DenseNet [17] is a famous network structure which connects each layer to every other layer in a feedforward fashion. The DenseNet can alleviate the vanishing-gradient problem; although the DenseNet can go very deep (more than 100 layers), the gradient values during training step can deliver to previous layers due to the skip connection. By introducing feedback and DenseNet into SR, Li et al. [8] proposed Superresolution Feedback Network (SRFBN).

2.2. Superresolution Feedback Network (SRFBN). SRFBN [8] is a newly proposed SR algorithm; by introducing feedback mechanism into SRFBN, SRFBN can go deeper without introducing the number of net parameters. The Feedback Block of SRFBN is with G projection groups sequentially;

TABLE 1: The performances (PSNR/SSIM) of the selected state-of-the-art algorithms.

Database of City100\algorithms	SRCNN	VDSR	DRCN	DRRN	RED	MemNet	ISRFN (ours)
Smartphone	25.36/0.7918	26.11/0.8227	25.72/0.8076	25.90/0.8150	26.07/ 0.8228	24.93/0.777	<i>26.13/0.8281</i>
DSLR	28.98/0.8326	30.05/0.8630	29.19/0.8415	29.97/0.8606	30.19/0.8662	28.16/0.8123	<i>30.20/0.8663</i>

each projection group includes a deconvolutional layer followed by a convolution layer; the deconvolutional layer can enlarge the features of LR image while the convolution layer can get rid of useless features. G projection groups are sequentially with dense skip connections among them; the dense connections on each layer can interact between all inputs with 1×1 convolution layer. Notice that the SRFBN extracts features directly from the input LR images to $2\times$, $4\times$ or $8\times$ SR but cannot handle $1\times$ SR with LR images.

Figure 1 gives us a brief illustration on the feedback mechanism of SRFBN (LRFB stands for the LR Feature Extraction Block, FB stands for the Feedback Block, and RB stands for the Reconstruction Block), where the LR image is the input of the SRFBN and the SR image is the output. First, the input LR image is fed into the LR Feature Extraction Block (LRFB) to extract its feature maps. Then, the output of LRFB (denoted by $f_{LR_b}()$) is fed into the Feedback Block (FB) to extract the details of the residual image, due to the feedback mechanism of FB; the output of FB (denoted by $f_{Fb}()$) is fed into two branches: one branch feeds into the next iteration as the input along with $f_{LR_b}()$; another branch is fed into the Reconstruction Block (RB) to compute the total loss and SR image. At last, the RB reconstructs the feature maps of $f_{Fb}()$ into residual image which is denoted by $f_{Rb}()$, so the output of current iteration is obtained by adding the residual image $f_{Rb}()$ and enlarged LR image. If there are N iterations, we will achieve N output images. The output of the i iteration is

$$SR^i(X_{LR}) = f_{Rb}(f_{Fb}(f_{LR_b}(X_{LR}), f_{Fb}^{(i-1)})) + \text{Upsampled}(LR). \quad (1)$$

For each iteration, the loss is chosen as L1 loss to optimize the difference between $SR^i(X_{LR})$ and the corresponding HR image. The total loss of SRFBN is defined as $L = (1/N) \times \sum_{i=1}^N W^i \|SR^i(X_{LR}) - HR\|_1$ where W^i means the weight of i -th loss, generally, $W^i = 1$. All the iteration outputs are taken and used for loss function, but only the last iteration output is considered as the SR image.

3. Improved Superresolution Feedback Network (ISRFN)

In this section, we describe the design detail of our proposed Improved Superresolution Feedback Network (ISRFN). Our ISRFN takes an LR image as input and generates SR image as output; the LR image and output SR image are of the same dimensions; this is quite different from the SRFBN, so modification and improvement are needed. Figure 2 illustrates the detailed structure of the proposed ISRFN. The modification and improvement are as follows.

As illustrated in Figure 2, the ISRFN is composed of 3 blocks: the LR Feature Extraction Block (LRFB), the

Feedback Block (FB), and the Reconstruction Block (RB); the details and improvement are as follows (for all layers in our ISRFN, m is the basic feature number, generally 32 or 64).

3.1. LR Feature Extraction Block (LRFB). Due to the feedback mechanism, the weights of FB are the same during interactions, which means that the LR image was not suitable to feed into FB directly. So, the LRFB is used to extract the features from original LR image, generating acceptable feature maps for FB. The LRFB is composed of 3 parts: input feature extraction layer (input LRFB), middle feature extraction layers (middle LRFB), and output feature extraction layer (output LRFB). This is different from SRFBN, which only contains the input feature extraction layer and the output feature extraction layer.

The details of LRFB are as follows:

- (1) The input feature extraction layer (input LRFB) is one convolution layer following nonlinear layer, with its in-channel as 3 for RGB images (1 for gray or YCbCr images); this layer is used as the input transformation, extracting $4 \times m$ basic features to feed into the middle feature extraction layers.
- (2) Middle feature extraction layer is cascading a pair of layers (convolutional and nonlinear, totally L pairs) repeatedly, with the number of both in-channel and out-channel being $4 \times m$. The middle feature extraction layer is used to extract deeper feature maps. Deeper feature maps can be extracted if the middle feature extraction layer goes deeper.
- (3) The output feature extraction layer is one convolution layer following nonlinear layer with the output channel as m with kernel size $= 1 \times 1$. By refined feature maps from $4 \times m$ to m , this layer aims to get rid of the useless features.

3.2. Feedback Block (FB). Figure 3 is the Feedback Block, which is the same as the Feedback Block of SRFBN [8]; we did not modify the Feedback Block because it was well designed. We only change some parameters to balance the cost and performance due to the improvements on the LRFB and RB blocks changing the network structure.

As shown in Figure 3, the Feedback Block contains G groups ($G=3$ in Figure 3), each group contains one deconvolutional followed by one convolution, then all the deconvolutional input and output are connected with dense skip (there is no connection from L_0^i to output), and all the convolutional input is dense and is skip connected (lines on bottom of Figure 3); notice that the dense connection is concatenated, so 1×1 convolutions are used to compress the

FIGURE 3: The Feedback Block structure with 3 groups.

the same RB layers, which means that RB assembles features of all iterations on the same network parameters to compute the loss, and the last iteration output for RB is the SR image. The RB is composed of 2 parts: assembled layers (assembled RB) and output reconstruction layer (output RB). The details and improvement are as follows:

- (1) Assembled layers (assembled RB) is cascading a pair of layers (convolutional and nonlinear, totally R pairs) repeatedly with the numbers of in-channels and out-channels being both $4 \times m$. In order to connect assembled RB between previous layer (FB) and output RB, the first convolutional layer's in-channel is m which is the same as the FB's outputs, and the last convolutional layer's out-channel is m which is the same as the output RB's in-channel. There are 2 particular cases: if the number of assembled layers is 0, which means no assembled layers, the output of FB is fed into output RB directly; if the number of assembled layers is 1, which means only one convolutional layer, so the in-channel and out-channel are both m to match the previous (FB) output and next (output RB) input. This layer is different from SRFBN's ($m = 64$ for SRFBN's FB which is costly). The mean running time on each iteration is 12s and total parameters are 1.61 M for ISFRN with $m = 32$, while they are 25s and 6.42 M for ISFRN with $m = 64$ with other parameters unchanged ($L = 4$, $R = 5$, $G = 6$, and $T = 3$) (we cannot fairly compare ISFRN with SRFBN since SRFBN needs preprocessed downsampling which is not suitable for our Blind Superresolution framework). So, we reduce $m = 32$ which means relative slim but deeper FB. We expect that the assembled RB can share workload from FB block, so the FB can only need to extract the alterable features during all the iterations.
- (2) The output feature extraction layer is one convolutional layer with the input channel as m and the output channel as 3 for RGB images (1 for gray or YCbCr images). This layer is used to assemble the feature maps, generating the residual image output. We gain the final output of current iteration by adding the residual image with the corresponding LR input.

3.4. Other Implementation Details. Following are the other implementation details not mentioned above:

- (1) The upsampling block: we remove the upsampling block, due to the same dimension of HR and LR images, so the LR input is directly added with the corresponding output residual image.
- (2) Network structure-related parameters: we will discuss the parameters on Section 4.2 (the depth of middle LRFB and middle RB, the FB related parameters). We use PReLU [18] as the activation function for all the nonlinear functions mentioned above. We reduce the base filter number of the Feedback Blocks (FB) to 32 to reduce the system cost.
- (3) The SRFBN's training skills: we follow these skills and choose L1 loss to optimize our proposed ISFRN, Adam optimizer to optimize the parameters of the network with initial learning rate 0.0001; we reduce the learning rate by multiplying 0.5 for every 200 epochs; the algorithm is implemented with PyTorch framework.

4. Experiments

In this section, we choose the City100 database [6] as the training data and valid data. There are 2 subdatabases on City100 database: smartphone cameras-based City100 database and DSLR-based City100 database; both are the same sceneries under different camera lens. In this paper, different from Chen's work [6], we perform two series of experiments, each for one subdatabase, generating two different SR networks. There are 100 images in each City100 database (both smartphone and DSLR), each in the form of HR-LR pair; the HR and LR are of the same dimensions; the resolution of the HR images is $2.4\times$ of the LR ones for smartphone cameras-based City100 database and $2.9\times$ for DSLR-based database. In Chen's work [6], 5 images were chosen from each City100 database, so the training data contains 95 images, with the valid data containing 5 images; we follow this strategy. Notice that the training data is too small to train deep network, so reused strategy is performed, during each iteration; 10 random patches of one image are extracted; the train patch is 60×60 . We drop some training skills (noise, Gaussian blur kernel, etc.) because LR images are real-world images.

We will first perform a set of experiments to demonstrate the performance (PSNR and SSIM) of our proposed ISRFN under different parameters. Second, we perform two series of experiments to choose the optimum network structures for each City100 database (smartphone and DSLR). At last, we compare our proposed ISRFN with other selected state-of-the-art algorithms under different subdatabases, respectively.

4.1. City100 Database Analysis and Benchmarks. In this section, we will give a brief analysis on both smartphone cameras and DSLR-based City100 database. Because the resolution of the HR images is $2.4\times$ ($2.9\times$) of the LR ones, we perform Bicubic and SRCNN [11] to analysis benchmark (PSNR and SSIM) on two City100 databases. The direct benchmark is performed with HR-LR pairs computing the PSNR/SSIM directly. The Bicubic benchmark is performed by $2.4\times$ and $2.9\times$; downscale the LR of City100 databases; then resize it with Bicubic to original size. The SRCNN is performed in the end-to-end study with HR-LR pairs for City100 databases. The results (PSNR/SSIM) are shown in Table 2.

Table 2 demonstrates that different image has different PSNR and SSIM due to different SR difficulty, but both subdatabases performed weaker than Set5 (we perform downsampling on HR images with $1/2.4$ scale and $1/2.9$ scale and then resize them to original size by Bicubic interpolation on Set5; the performance (PSNR/SSIM) is 32.12/0.9052 for $2.4\times$ and 30.66/0.8745 for $2.9\times$), so we expect that the City100 valid set is difficult to SR compared to Set5. The reason that we guess is as follows: 1. the LR images of City100 valid set are performed, not merely having the lower resolution, but containing the filed-of-view in realistic imaging system, which will lead to some distortion. 2. The LR images of smartphone cameras-based City100 are generated by smartphone cameras, whose CCD sensor is smaller than tradition DSLR so LR images contain heavier real-world noise, which cannot be simulated by Bicubic or Gaussian blur kernel. Real LR-based SR is harder than simulated LR-based SR; lower performance (PSNR and SSIM) is reasonable.

4.2. Parameters for Our Proposed ISRFN. In this section, we discuss the parameters selected for our proposed Improved Superresolution Feedback Network (ISRFN) on two City100 databases (smartphone and DSLR). The numbers of middle layers for LRFB (denoted by L , the same as that in Figure 2) and assembled layers for RB (denoted by R , the same as that in Figure 2) are critical parameters for our proposed ISRFN, while the groups (denoted by G , the same as [6]) and steps (denoted by T , the same as [6]) for FB are critical parameters for the FB block-based network. So, we will discuss the proposed ISRFN under both smartphone and DSLR-based City100 databases. Due to 4 parameters, we will discuss two of them with other two fixed ones. For all tables in this section, the best performance is shown in bold and the selected performance is shown in italics.

4.2.1. Performances for L (Middle LRFB) and R (Assembled RB) on Smartphone-Based City100 Database. In this section, we will discuss the numbers of L (middle LRFB) and R (assembled RB) with groups (G) and steps (T) fixed; we fix $G=6$ and $T=3$ under a large number of experiments. We perform the proposed ISRFN under L from 0 to 5 and R from 0 to 5 ($L=0$ or $R=0$ means no convolutional layer for middle LRFB or assembled RB). During the training step, we performed 1000 iterations for each parameters' group; the performances (PSNR/SSIM) are listed in Table 3.

Table 3 demonstrates that both L (middle LRFB) and R (assembled RB) are critical for the proposed ISRFN; large L and R will improve the performance (PSNR/SSIM), but the increase amplitude of PSNR/SSIM is reduced if L and R are large enough, so there is a balance between the cost and performance for the proposed ISRFN. Considering that large $L+R$ is difficult to converge, we choose $L=4$ and $R=5$ ($L+R=9$) as our proposed ISRFN under smartphone-based City100 database.

Table 3 demonstrates that the performance increases greatly with R increase; the reason we guess is that the valid database of smartphone-based City100 database is difficult to SR (the same as the result of Section 4.1), so large Re-construction Block will generate acceptable SR image from FB. Notice that this is quite different from the VDSR-based City100 database (will be discussed in Section 4.2.3), which means that different camera lens with different character will lead to different network structure.

4.2.2. Performances with Different Group (G) and Step (T) on Smartphone-Based City100 Database. In this section, we will discuss group (G) and step (T) on FB, with L (middle LRFB) and R (assembled RB) fixed; we chose $L=4$ and $R=5$ which is discussed in Section 4.2.1. We perform our proposed ISRFN under G from 1 to 6 and T from 1 to 6. During the training step, we performed 1000 iterations for each parameters' group.

Table 4 demonstrates that both T (step) and G (group) are critical for the proposed ISRFN; large T and G will improve the performance (PSNR/SSIM), but the increase amplitude of PSNR/SSIM is reduced if T and G are large enough, so there is a balance between the cost and performance for the proposed ISRFN. Because the FB block has $2\times G$ layers (G projection groups sequentially, each group is convolution and deconvolutional pair) and all the layers computed T iterations to enforce feedback, $2\times T\times G$ layers is need for the FB. We will choose an acceptable performance with $O(T\times G)$.

Table 4 demonstrates that G is more critical, the performance will increase with G increase, and large G is required, so we choose $G=6$ for performance because large G means more convolutional and deconvolutional layers. Large T cannot increase the performance significantly; if $T<3$, T is critical, but if $T\geq 4$, the performance increment will not be significant. The reason that we guess is the Feedback Block used to refine the superresolution image with the same network, so similar high-level features are suitable for Feedback Block. If the feedback iteration goes

TABLE 2: The performance benchmarks (PSNR/SSIM) on valid City100.

Method		Images					Means
HR-LRDirect	Smartphone (2.40×)	25.18/0.8263	27.09/0.8338	22.35/0.6533	26.77/0.8450	21.82/0.7064	24.64/0.7730
	DSLR (2.90×)	28.77/0.8645	30.20/0.8451	26.52/0.7319	30.66/0.8672	24.60/0.7521	28.15/0.8122
Bicubic	Smartphone (2.40×)	24.85/0.8074	26.88/0.8216	22.02/0.6167	26.38/0.8250	21.45/0.6729	24.31/0.7487
	DSLR (2.90×)	28.04/0.8441	29.63/0.8301	25.90/0.6969	29.99/0.8518	24.02/0.7223	27.51/0.7892
SRCNN	Smartphone (2.40×)	25.78/0.8392	27.80/0.8455	22.94/0.6748	27.79/0.8633	22.49/0.7361	25.36/0.7918
	DSLR (2.90×)	29.75/0.8812	30.87/0.8543	27.27/0.7663	31.35/0.8740	25.71/0.7870	28.98/0.8326

TABLE 3: The performances (PSNR/SSIM) under different L and R on smartphone-based City100 database ($G=6$ and $T=3$ fixed).

R	L					
	0	1	2	3	4	5
0	25.94/0.8212	25.99/0.8230	26.02/0.8244	26.04/0.8250	26.06/0.8254	26.07/0.8261
1	25.98/0.8224	26.00/0.8238	26.01/0.8239	26.05/0.8263	26.04/0.8252	26.08/0.8266
2	26.00/0.8240	26.00/0.8236	26.03/0.8247	26.05/0.8248	26.07/0.8261	26.05/0.8266
3	26.00/0.8224	26.04/0.8257	26.07/0.8269	26.08/0.8265	26.08/0.8259	26.10/0.8265
4	26.03/0.8252	26.06/0.8255	26.08/0.8265	26.09/0.8256	26.10/0.8270	26.11/0.8281
5	26.06/0.8258	26.09/0.8274	26.10/0.8271	26.10/0.8274	26.13/0.8285	26.14/0.8286

TABLE 4: The performances (PSNR/SSIM) under different G and T on smartphone-based City100 database ($L=4$ and $R=5$ fixed).

G	T					
	1	2	3	4	5	6
1	26.05/0.8255	26.06/0.8262	26.07/0.8262	26.07/0.8263	26.07/0.8264	26.08/0.8264
2	26.07/0.8262	26.07/0.8257	26.07/0.8271	26.07/0.8265	26.09/0.8260	26.09/0.8267
3	26.09/0.8263	26.08/0.8270	26.08/0.8265	26.09/0.8268	26.10/0.8267	26.09/0.8269
4	26.07/0.8264	26.10/0.8270	26.10/0.8269	26.11/0.8276	26.12/0.8277	26.10/0.8272
5	26.08/0.8261	26.11/0.8279	26.10/0.8276	26.12/0.8280	26.11/0.8274	26.11/0.8281
6	26.11/0.8270	26.10/0.8276	26.13/0.8285	26.11/0.8280	26.13/0.8284	26.13/0.8289

too much, because loss function takes use of all iterations' outputs, previous iteration outputs will balance the loss, leading to unexpected SR output (only the last iteration output). So, we choose $T=3$. This is different from SRFBN [8] because the SRFBN chooses the base number of convolutional filters $m=64$ which is larger than our proposed ISRFN (we reduce $m=32$ to reduce cost). Small base number of convolutional filters of ISRFN means narrow path way of feature, resulting sensitively in feedback iteration (T) and low performance, so depth LRFB and RB compensate for them.

4.2.3. Parameters on DSLR-Based City100 Database. In this section, we will discuss different parameters on DSLR-based City100 database. Table 5 lists the performances under different L (middle LRFB) and R (assembled RB) with groups ($G=6$) and steps ($T=3$) fixed, while Table 6 lists the performances under different G and T with L and R fixed ($L=5$ and $R=4$). This is quite the same with previous experiments; only the database changed into DSLR-based City100 database.

Table 5 demonstrates that (different from Table 4) the performance increases greatly with L increase; the reason we guess is that the valid database of VDSR-based City100 database is easy to SR (the same as the result of Section 4.1), so small Reconstruction Block will generate acceptable SR image from FB. Section 4.1 demonstrates that VDSR-based City100 database is easier (higher PSNR/SSIM under the same SR algorithms) than smartphone-based City100 database. Large L and R will improve the performance (PSNR/SSIM), but the increase amplitude of PSNR/SSIM is reduced if L and R are large enough, so we choose $L=5$ and $R=4$ ($L+R=9$) as our proposed ISRFN under VDSR-based City100 database to balance the cost and performance.

We follow the same strategy as that in Section 4.2.2 to choose the group (G) and step (T) with L and R fixed ($L=5$ and $R=4$ fixed) on DSLR-based City100 database. We choose $G=6$ and $T=3$ which is the same as smartphone-based City100 database.

In conclusion, we choose two network structures for two different databases (smartphone and DSLR) because two databases have different camera lens characters, so different network structure is needed. We choose the same group

TABLE 5: The performances (PSNR/SSIM) under different L and R on DSLR-based City100 database ($G=6$ and $T=3$ fixed).

R/L	0	1	2	3	4	5
0	30.03/0.8631	30.12/0.8651	30.14/0.8656	30.13/0.8657	30.15/0.8655	30.15/0.8661
1	29.91/0.8600	30.10/0.8653	30.14/0.8657	30.16/0.8660	30.17/0.8661	30.16/0.8661
2	30.09/0.8645	30.12/0.8651	30.13/0.8655	30.16/0.8659	30.15/0.8658	30.18/0.8659
3	30.09/0.8645	30.14/0.8656	30.14/0.8658	30.17/0.8662	30.18/0.8664	30.19/0.8667
4	30.01/0.8644	30.12/0.8657	30.16/0.8660	30.15/0.8655	30.18/0.8665	30.20/0.8663
5	30.09/0.8648	30.13/0.8657	30.15/0.8658	30.19/0.8665	30.18/0.8663	30.21/0.872

TABLE 6: The performances (PSNR/SSIM) under different G and T on DSLR-based City100 database ($L=5$ and $R=4$ fixed).

G/T	1	2	3	4	5	6
1	30.10/0.8649	30.11/0.8651	30.13/0.8648	30.12/0.8653	30.12/0.8653	30.12/0.8650
2	30.12/0.8651	30.14/0.8656	30.13/0.8655	30.15/0.8654	30.15/0.8658	30.14/0.8653
3	30.16/0.8659	30.15/0.8656	30.16/0.8653	30.17/0.8661	30.18/0.8664	30.16/0.8656
4	30.16/0.8657	30.17/0.8661	30.18/0.8659	30.18/0.8665	30.18/0.8664	30.18/0.8663
5	30.17/0.8664	30.18/0.8664	30.18/0.8666	30.18/0.8668	30.20/0.8668	30.19/0.8663
6	30.18/0.8666	30.19/0.8665	30.20/0.8663	30.21/0.8666	30.21/0.8672	30.22/0.8672

($G=6$) and step ($T=3$) for both smartphone and DSLR-based databases, meaning the same FB structure, while having different L (middle LRFB) and R (assembled RB). Due to difficulty of SR in smartphone-based City100 database, we choose $L=4$ and $R=5$ for smartphone-based City100 database; larger R means deeper RB, which gains better reconstruction character. We choose $L=5$ and $R=4$ for smartphone-based City100 database to gain better input feature maps for FB.

4.3. Comparison with the State-of-the-Art SR Algorithms. Because the LR images and the HR images are of the same dimensions for the City100 database, we choose the algorithms with its input LR images preprocessed for interpolation, which means that we can make use of the HR-LR image pair of City100 database directly, only removing the interpolation step without modifying the network structure, so we choose SRCNN [11], VDSR [10], DRCN [15], Deep Recursive Residual Network (DRRN) [19], the very deep Residual Encoder-Decoder Networks (RED) [20], and MemNet [21] as the state-of-the-art algorithms in this experiment. We did not choose SRFBN because SRFBN needs downsampling which is not suitable for our Blind Super-resolution framework. The City100 database is proposed to solve the Blind Superresolution without considering the blur kernel for downscaling process and LR images are the real-world images, so we drop some training skills (noise, Gaussian blur kernel, etc.) which are widely used in fake LR image generating SR methods. We follow the classic SRCNN parameters because they are well designed with 1500 iterations as their convergence is slower than other chosen algorithms; while other algorithms have 1000 iterations, our proposed ISRFN has $L=4$ and $R=5$ for smartphone-based City100 database and $L=5$ and $R=4$ for DSLR-based City100 database. The result is listed in Table 1 (the best

performance is shown in italics and the second-best performance is shown in bold).

Table 1 demonstrates that our proposed ISRFN outperforms other selected algorithms on both databases; due to different camera lens characters, smart choice of L and R gains the best performance without being too costly. The VDSR (PSNR) and RED (SSIM) are both the second-best performances for smartphone cameras-based City100 database while the RED is the second-best performance for DSLR-based City100 database, indicating different network structure suitable for different camera lens characters.

Figure 4 illustrates some visual comparison on smartphone cameras-based City100 database and DSLR-based City100 database. For all the shown examples, especially the images edges, our method perceptually outperforms other state-of-the-art ones. Compared with other methods, our proposed ISRFN could alleviate the distortions and generate more accurate detail in SR images especially in the boundary of elements in the picture. The VDSR performs the second-hand (PSNR) on smartphone cameras-based City100 database, while RED performs the second-hand (PSNR/SSIM) on DSLR-based City100 database, so SR patch images in Figure 4 also perform the second-hand for VDSR on smartphone cameras database and for RED on DSLR-based database.

We train our proposed ISRFN under single NVIDIA 3090 GPU, the mean running time on each iteration is 12 s with 1000 iterations, and total parameters are 1.61 M for ISRFN with $m=32$, so the computational cost of the ISRFN is about 3 hours and 20 minutes to train a network under City100 database. The most consumption of our proposed ISRFN for Blind Superresolution framework is to choose an optimum network structure for new camera lens which needs to run many times; fortunately the network structure selection process can parallel running on multiple GPUs in practical applications.

FIGURE 4: Visual comparison on smartphone cameras-based City100 database and DSLR-based City100 database.

5. Conclusion

In this paper, we propose an Improved Superresolution Feedback Network (ISRFN) which is derived from the famous SRFBN. Our proposed ISRFN is focused on the Blind Superresolution problem by making use of the City100 database and modifying network structure. We extend the input (LR Feature Extraction Block) layer, so our ISRFN can extract and use much more deeper features. We reduce the base filter number of the FB to 32, as a narrow but relative deeper Feedback Block, to gain deeper features. We extend the output (Reconstruction Block) layer, so deeper features can be assembled into SR image. Different from previous work, we train two ISRFNs on different City100 databases (smartphone and DSLR), respectively, with different network structure; experiments show that different camera lens (smartphone and DSLR) has different best network (balance between the performance and cost) and our proposed ISRFNs achieve the better performance in comparison with the state-of-the-art methods.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant number: U2006228).

References

- [1] K. Li, S. Yang, R. Dong, X. Wang, and J. Huang, "Survey of single image super-resolution reconstruction," *IET Image Processing*, vol. 14, no. 11, pp. 2273–2290, 2020.
- [2] Z. Wang, J. Chen, and S. C. Hoi, "Deep learning for image super-resolution: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–22, 2020.
- [3] J. Gu, H. Lu, W. Zuo, and C. Dong, "Blind super-resolution with iterative kernel correction," in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1604–1613, IEEE, Las Vegas, NV, USA, June 2019.
- [4] K. Zhang, W. Zuo, and L. Zhang, "Deep plug-and-play super-resolution for arbitrary blur kernels," in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision*

- and *Pattern Recognition*, pp. 1671–1681, IEEE, Las Vegas, NV, USA, June 2019.
- [5] Y. Wang, L. Wang, H. Wang, P. Li, and H. Lu, “Blind single image super-resolution with a mixture of deep networks,” *Pattern Recognition*, vol. 102, p. 107169, June 2020.
  - [6] C. Chen, Z. Xiong, X. Tian, Z. Zha, and F. Wu, “Camera lens super-resolution,” in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1652–1660, IEEE, Las Vegas, NV, USA, June 2019.
  - [7] X. Xu, Y. Ma, and W. Sun, “Towards real scene super-resolution with raw images,” in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1723–1731, IEEE, Long Beach, CA, USA, June 2019.
  - [8] Z. Li, J. Yang, Z. Liu, X. Yang, G. Jeon, and W. Wu, “Feedback network for image super-resolution,” in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3867–3876, IEEE, June 2019.
  - [9] C. Szegedy, W. Liu, Y. Jia et al., “Going deeper with convolutions,” in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1–9, IEEE, Boston, MA, USA, June 2019.
  - [10] J. Kim, J. K. Lee, and K. M. Lee, “Accurate image super-resolution using very deep convolutional networks,” in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1646–1654, IEEE, Las Vegas, NV, USA, June 2016.
  - [11] C. Dong, C. C. Loy, K. He, and X. Tang, “Learning a deep convolutional network for image super-resolution,” in *Proceedings of the Computer Vision – ECCV*, pp. 184–199, Glasgow, UK, September 2014.
  - [12] R. Farrugia and C. Guillemot, “Light field super-resolution using a low-rank prior and deep convolutional neural networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 5, p. 1, 2019.
  - [13] B. Li, Y. Zhang, B. Zhao, and H. Shao, “3D-ReConstnet: a single-view 3d-object point cloud reconstruction network,” *IEEE Access*, vol. 8, pp. 83782–83790, 2020.
  - [14] B. Li, Y. Zhang, and F. Sun, “Deep residual neural network based PointNet for 3d object part segmentation,” *Multimedia Tools and Applications Journal*, pp. 1–15, 2020.
  - [15] J. Kim, J. K. Lee, and K. M. Lee, “Deeply-recursive convolutional network for image super-resolution,” in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1637–1645, IEEE, Providence, RI, USA, June 2016.
  - [16] K. Jiang, Z. Wang, P. Yi, and J. Jiang, “Hierarchical dense recursive network for image super-resolution,” *Pattern Recognition*, vol. 107, p. 107475, 2020.
  - [17] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, IEEE, Honolulu, HI, USA, July 2017.
  - [18] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: surpassing human-level performance on imagenet classification,” in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1026–1034, IEEE, San Diego, CA, USA, December 2015.
  - [19] Y. Tai, J. Yang, and X. Liu, “Image super-resolution via deep recursive residual network,” in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3147–3155, IEEE, Honolulu, HI, USA, July 2017.
  - [20] X. Mao, C. Shen, and Y. Yang, “Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections,” in *Proceedings of the Advances in Neural Information Processing Systems 29 (NIPS 2016)*, Lake Tahoe, NV, USA, June 2016.
  - [21] Y. Tai, J. Yang, X. Liu, and C. Xu, “MemNet: a persistent memory network for image restoration,” in *Proceedings of the CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4549–4557, IEEE, Venice, Italy, October 2017.