Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Single MR image super-resolution via channel splitting and serial fusion network[☆]

Xiaole Zhao[a], Yulun Zhang[b], Yun Qin[c], Qian Wang[d], Tao Zhang[c,*], Tianrui Li[a,*]

[a] School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu, Sichuan 611756, China
[b] Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115, USA
[c] School of Life Science and Technology, University of Electronic Science and Technology of China, Chengdu, Sichuan 611731, China
[d] Tangshan Seismic Station of Hebei Earthquake Agency, Tangshan, Hebei 066300, China

ARTICLE INFO

Article history:
Received 19 October 2021
Received in revised form 23 March 2022
Accepted 24 March 2022
Available online 31 March 2022

Keywords:
Convolutional neural network
Magnetic resonance imaging
Channel splitting
Super-resolution
Serial fusion

ABSTRACT

In magnetic resonance imaging (MRI), spatial resolution is an important and critical imaging parameter that represents how much information is contained in a unit space. Acquiring high-resolution MRI data usually takes a long scanning time and is subject to motion artifacts due to hardware, physical, and physiological limitations. Single image super-resolution (SISR) based on deep learning is an effective and promising alternative technique to improve the native spatial resolution of magnetic resonance (MR) images. However, because of the simple diversity and single distribution of training samples, the effective training of deep models with medical training samples and improvement of the tradeoff between model performance and computing overhead are major challenges. In addition, deeper networks are more difficult to effectively train since the information is gradually weakened as the network deepens. In this paper, a novel channel splitting and serial fusion network (CSSFN) is presented for single MR image super-resolution. The proposed CSSFN splits hierarchical features into a series of subfeatures, which are then integrated together in a serial manner. Hence, the network becomes deeper and can discriminatively and reasonably deal with the subfeatures. Moreover, a dense global feature fusion (DGFF) is adopted to integrate the intermediate features, which further promotes the information flow in the network and helps to stabilize model training. Extensive experiments on several typical MR images show the superiority of our CSSFN models to other advanced SISR methods.

1. Introduction

Magnetic resonance imaging (MRI) is an essential and widely-used tool for diagnosis and image-guided therapeutics. Usually, high-resolution (HR) imaging is preferred in clinical practice due to clearer image structure and texture, as well as the benefits of subsequent analysis and processing [1]. However, the acquisition of HR images is constrained by hardware, physical and physiological factors, and improving the spatial resolution of magnetic resonance (MR) images typically reduces the signal-to-noise ratio (SNR) or prolongs the imaging time [2], which further increases the risk of MR images affected by motion artifacts.

Super-resolution (SR) is an effective and cost-efficient alternative technique to increase the spatial resolution of MR images, which aims at recovering an HR image from one or multiple low-resolution (LR) images. To date, a large number of SR methods have been investigated and proposed for both natural images and medical images, e.g., interpolation-based methods [3], edge-guided methods [4], modeling/reconstruction methods [5], example learning methods [6], and sparse representation methods [7]. However, the performance of these conventional methods is essentially limited since they apply inadequate extra information and models with limited representational capacity to solve the ill-posed inverse problem of image SR tasks [8].

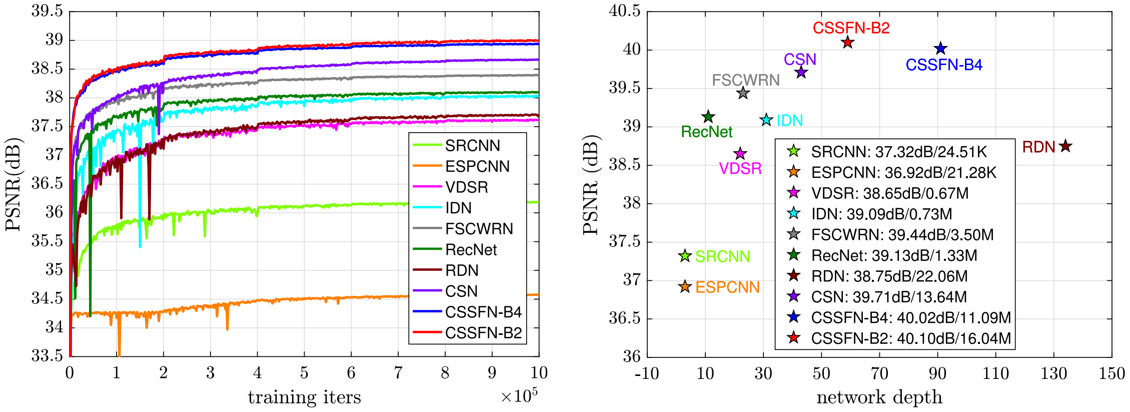
In recent years, single image super resolution (SISR) based on deep learning [9] has exhibited great superiority over conventional SR methods. A pioneering work that uses convolutional neural networks (CNNs) [10] to deal with SISR is the Super-Resolution Convolutional Neural Network (SRCNN) [11]. It implicitly learns an end-to-end mapping function between LR and HR images by utilizing a fully-convolutional network (FCN). Subsequently, more advanced SISR methods based on CNNs were proposed. Some typical examples are DRCN [12], TDAN [13], VDSR [14], MemNet [15], ESPCNN [16], SRResNet [17], EDSR [18], RDN [19], CMSCN [20], RCAN [21], EBRN [22], SAN [23], CSNLN [24] and NLSN [25] etc. These models have overwhelming

[☆] This work is supported in part by the National Natural Science Foundation of China under Grant Nos. 62102330, 61876158 and 61773324, and in part by the Fundamental Research Funds for the Central Universities under Grant No. 2682021CX040.

[*] Corresponding authors.

E-mail addresses: zxlation@foxmail.com (X. Zhao), yulun100@gmail.com (Y. Zhang), qyuner@163.com (Y. Qin), wangqianqhd123@163.com (Q. Wang), tao.zhang@uestc.edu.cn (T. Zhang), trli@swjtu.edu.cn (T. Li).

https://doi.org/10.1016/j.knosys.2022.108669
0950-7051/© 2022 Elsevier B.V. All rights reserved.



(a) Validation curves

(b) Testing results (PSNR/Params)

Fig. 1. Model performance versus model scale between several deep models. The results are evaluated on T2-weighted MR images of the IXI dataset for SR×2. B2 and B4 imply the number of branches when performing channel splitting.

advantages over traditional methods and greatly promote the best state of SISR performance. However, they are mainly targeted at natural images, instead of medical images (or more specifically, MR images). Therefore, they may be unsuitable for solving medical image SR tasks due to the degradation of training examples [8] although they have excellent performance on natural images.

Due to the tremendous success of deep learning techniques in computer vision and pattern recognition, some deep learning based methods specializing in the SISR tasks of medical images have also emerged [26,27]. These methods utilize relatively shallow networks to process medical images, e.g., Pham et al. [26] presented an algorithm for super-resolving single brain MR images according to SRCNN [11]. Despite their extension to 3D cases (named SRCNN3D), the entire network is very shallow and the representative capacity of the model is relatively limited, resulting in unsatisfactory performance.

The depth of CNN models is of crucial importance for image SR [21], and is usually defined as the longest path from the input to the output of the network [8,20]. However, deeper networks are typically more difficult to fully and effectively train, especially with medical images due to the degeneracy of training samples [8]. Actually, it is experimentally verified that the original EDSR model [18] with approximately 43M parameters and 70 layers of depth is difficult to be well-trained with 2D proton density (PD) images [8]. Although Zhao et al. [28] trained the EDSR [18] using T1-weighted magnetization-prepared rapid gradient echo (MP-RAGE) images, the reported results are not satisfactory, possibly due to over/underfitting. In particular, this is probably because the EDSR, which has enormous parameters and a very deep structure, is not adequately well-trained with "good" training samples. In this regard, *more exploration is needed to determine whether deeper networks are capable of further contributing to improving the performance of medical image SR and how to construct trainable networks with much deeper structures for medical images.*

A recent work [8] has alleviated the dilemma between the trainability and the performance of CNN models for MR image SR to some extent. The work presented an effective way to deepen the network but without significant enlargement in the number of model parameters, i.e., channel splitting. The model, however, is a kind of multistream structure, and multiple branches for information transmission are formed by channel splitting instead of the reuse of preceding features [20,29]. The multistream structure indicates that information flow in the network is ***locally parallel***. In this paper, we present a ***serial*** information fusion mechanism for channel splitting. The proposed model, which we term the channel splitting and serial fusion network (CSSFN), first splits the hierarchical features into a series of subfeatures and then fuses them together in a serial manner. *Despite channel splitting, our CSSFN is a single-branch network that is essentially different from the CSN [8]. Therefore, it can reach much deeper depths (up to 90 layers) without drastic parameter expansion* (e.g., CSSFN-B4 in Fig. 1(b)). On the other hand, channel splitting also allows our CSSFN model to deal with intermediate features discriminatorily. However, different from [8], the channel discrimination ability compared to the baseline is mainly derived from the subfeatures located at different network depths.

However, increasing the network depth will significantly increase the training instability when network parameters are roughly the same, resulting in training nonconvergence or failure. To alleviate the instability of model training caused by the single-branch structure and increase in network depth, we adopt a dense global feature fusion (DGFF) strategy [30] to improve the information flow. Although the DGFF increases model parameters slightly, our models still have moderate parameters compared with EDSR [18], RDN [19] and CSN [8]. Our CSSFN consists of a series of channel splitting and serial fusion blocks (CSSFB), each of which has one or more interblock connections to all subsequent blocks, thereby propagating its local features to all successors. The detailed structure of the proposed model is illustrated in Figs. 2 and 3. In summary, the main contributions of this work are as follows:

- We further improve the tradeoff between model performance and computing overhead for MR image SR by introducing a serial local feature fusion (SLFF) strategy for channel splitting;
- Through the combination of SLFF and DGFF, we alleviate the dilemma between the trainability and network scale caused by the degradation of medical training samples, and accidentally obtain performance gain;
- We argue that channel splitting helps to improve channel discrimination ability and reduce the risk of over/underfitting. However, once channel splitting is performed, increasing the number of subfeatures will exacerbate the fitting problem and result in performance degradation;
- We experimentally confirm that, through pseudo 3D execution, training samples of degraded MR images are indeed more likely to cause the fitting problem of large-scale networks, verifying the conjecture of [8].

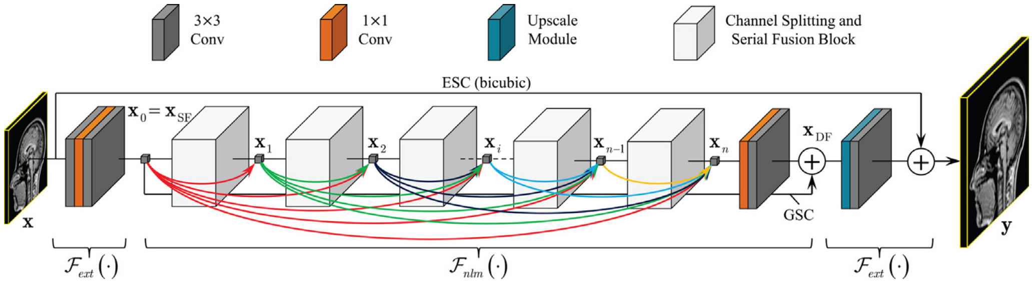


Fig. 2. Overall architecture of the proposed CSSFN model. The symbol "+" indicates elementwise summation between two tensors with the same shape. GSC and ESC denote the global skip connection and external skip connection, respectively. The hierarchical features are integrated together in a dense learning manner (DGFF).

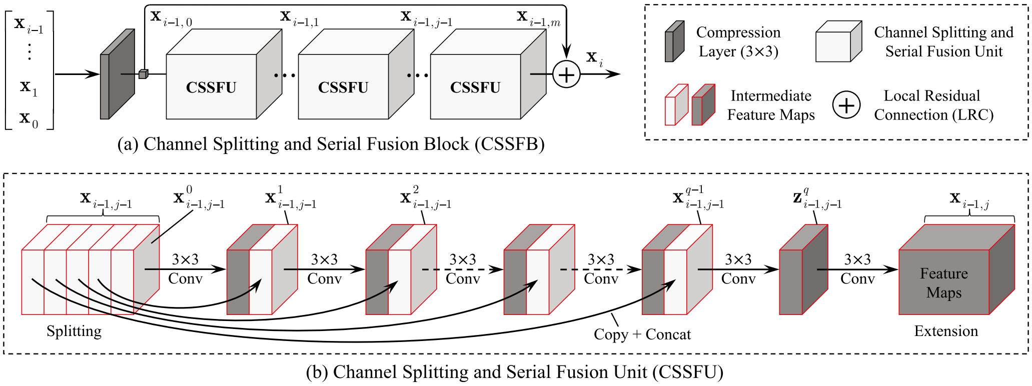


Fig. 3. Basic building block, CSSFB, consisting of m stacked channel splitting and serial fusion units (CSSFU). (a) Each CSSFB also has a short skip connection (SSC) to form local residual learning. (b) The input feature of each CSSFU is split into q subfeatures. Cuboids in light gray imply subfeatures from channel splitting of the input feature, and those in dark gray denote the subfeatures produced by the 3 × 3 conv layer.

The remainder of this paper is organized as follows. We first present some previous work related to the proposed model in Section 2. The proposed CSSFN is illustrated in detail in Section 3 and the experimental results are presented in Section 4. Finally, we discuss some related topics in Section 5 and conclude the work in Section 6.

2. Related work

2.1. MR image super-resolution

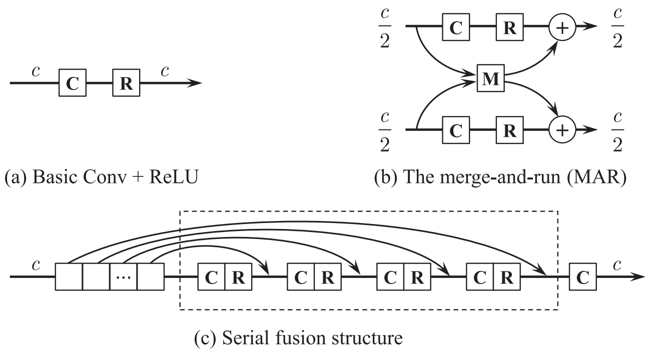
The purpose of MR image SR tasks is to overcome hardware limitations and meet the clinical needs of imaging procedures by reconstructing HR images from LR acquisitions using post-processing methods. SR methods could have strong impacts on structural MRI when focusing on cortical surface or fine-scale structure analysis [26]. The application of SR methods to MR images initially focuses on multiple image super-resolution (MISR), e.g., [5,31]. The MISR methods, however, usually need calibration and integration between multiple LR images, which is a very challenging problem in itself and is hard to achieve with satisfactory performance [8].

SISR methods can avoid the difficulty of calibration and integration of LR images, where only one LR image is required to reconstruct its HR counterpart. A major problem with SISR methods is that limited extra information is available for HR image reconstruction. Subsequently, some SR methods based on traditional machine learning, e.g., sparse representation [7], example learning [32] and compressive sensing [33] etc., have emerged. However, the limited representational capability of these SR methods makes them unable to execute accurate and highly nonlinear mapping between LR and HR images. Recently, advanced SISR methods based on deep learning [9] have been applied to MR image SR tasks [1,8,26–28,34], which have greatly promoted the performance of SR technologies for medical images. Two recent works, W^{2}AMSN [35] and SERAN [36], have improved the performance of MR SR through an attention mechanism and multiscale feature fusion. They further enrich the research on single MR image super-resolution based on CNNs.

2.2. Channel discrimination

The feature maps on different channels of deep CNN models have different types of information and different impacts on the performance of deep models [21]. It is reasonable to deal with these features discriminatively. One typical manner for channel discrimination is the self-attention mechanism, which is broadly viewed as a tool to bias the allocation of available processing resources toward the most informative components of the input signal [37]. In recent years, it has been introduced to deep neural networks (DNNs) to boost the performance of deep models, such as image generation, image captioning, image classification and image restoration. All these methods have further improved the best state of related fields. For instance, the residual channel attention network (RCAN) [21] pushed the state-of-the-art performance of SR tasks forward on natural images, with a channel attention mechanism and an extremely deep architecture (over 400 layers).

However, few works have investigated the effect of channel discrimination for low-level computer vision tasks in the medical image processing community (e.g., MR image SR). In this respect, a representative work for single MR image SR is the CSN network [8], where channel discrimination is achieved by channel splitting and merge-and-run mapping between different branches [29]. This model adopted a *parallel* two-way channel splitting strategy to handle the hierarchical features on different channels, which limited the network depth to some extent. Motivated by the channel discrimination and increase in network



(a) Basic Conv + ReLU

(b) The merge-and-run (MAR)

(c) Serial fusion structure

Fig. 4. Stage mappings for comparing branch information fusion (BIF). C, R and + denote Conv, ReLU, and skip connection respectively. (a) Basic Conv + ReLU. (b) Merge-and-run [29] with channel splitting [8]. The number of branches is set to 2 for display purposes, but in the experiments, it is set to 4 for fair comparison. (c) The proposed serial fusion.

depth, we integrate subfeatures into a single branch in a ***serial*** manner (Fig. 3(b)). In this way, the network becomes deeper and thinner (if we keep the channel of subfeatures fixed), which is analogous to stretching a rubber band. Despite the single-branch structure, serial fusion retains the channel discrimination ability of the network, because these subfeatures have different network depths in feed-forward or feed-back propagation.

2.3. Hierarchical feature fusion

The notorious problem of gradient vanishing and weakened information flow becomes more obvious as the network depth increases [21], which seriously hinders the training of deep models. Unfortunately, the degradation of training samples will further aggravate the difficulty of training deep models for medical images [8]. To promote information transmission and improve the trainability of deep networks, many recent works have been devoted to resolving these problems. A popular method is to fuse the hierarchical features through skip connections, e.g., DenseNet [30] helps to explore new features, and ResNet [38] contributes to the reuse of preceding features. The basic idea of fusing hierarchical features by residual and dense learning is also broadly applied to many CNN-based methods [14,17–19,21,34,39] to build very wide and deep models for performance improvement.

Due to modular design of the recent CNN-based models, hierarchical feature fusion can be divided into local feature fusion (LFF) and global feature fusion (GFF), which integrate intrablock and interblock features, respectively. LFF is conducive to learning more effective hierarchical features and stabilizing model training [19], while GFF enables short paths to be built from high-level features to low-level features directly and ease the vanishing gradient problem for training very deep networks [30]. In the proposed CSSFN, local features are fused together in a serial manner, as shown in Fig. 3(b) and Fig. 4(c). It can be viewed as a manner of partially dense learning where subfeatures are "densely" connected to subsequent layers. A short skip connection (SSC) [19,21] (shortcut connection [38]) is then used to conduct local residual learning. For GFF, we present a dense global feature fusion (DGFF) for effective feature exploitation and important information preservation (Fig. 2). In addition, it helps to alleviate the instability of model training caused by the increase in network depth and the decrease in network width.

3. Proposed method

3.1. Network architecture

In this work, we focus on the task of single 2D MR image SR. Given an LR image $\mathbf{x} \in \mathbb{R}^{h\times w}$, the target is to recover an HR image $\mathbf{y} \in \mathbb{R}^{(r\cdot h)\times(r\cdot w)}$ that corresponds to LR input $\mathbf{x}$, where r is the scaling factor. The overall structure of the proposed CSSFN is outlined in Figs. 2 and 3, which consists of three typical parts, i.e., shallow feature extraction, nonlinear mapping from shallow features to deep features and HR image recovery. As investigated in [8], we extract the shallow features by two 3 × 3 conv layers with a 1 × 1 conv layer in the middle. We denote $\mathcal{F}_{ext}(\cdot)$ as the corresponding mapping function of the entire shallow feature extraction stage, then the extracted shallow features $\mathbf{x}_{\mathrm{SF}}$ can be represented as:

$$\mathbf{x}_{\mathrm{SF}} = \mathcal{F}_{ext}(\mathbf{x}), \tag{1}$$

where $\mathbf{x}$ stands for the original LR input. Next, $\mathbf{x}_{\mathrm{SF}}$ is fed into the nonlinear mapping, which contains a series of stacked CSSFB blocks. The entire nonlinear mapping process can be expressed as follows:

$$\mathbf{x}_{\mathrm{DF}} = \mathcal{F}_{nlm}(\mathbf{x}_0), \tag{2}$$

where $\mathbf{x}_0 = \mathbf{x}_{\mathrm{SF}}$ is the extracted shallow features and $\mathcal{F}_{nlm}(\cdot)$ is the function corresponding to the entire nonlinear mapping. To make more full use of features and further stabilize model training, we also use GFF [8,19] to integrate these intermediate features. However, unlike [8,19], we integrate hierarchical features in a dense learning manner [30], instead of concatenating all interblock features together and then fusing them through a 1 × 1 conv layer. Therefore, the input to the ith building block is the concatenation of the output feature maps of all preceding blocks, i.e., $[\mathbf{x}_{i-1}, \ldots, \mathbf{x}_1, \mathbf{x}_0]$, where $[\ldots]$ denotes the concatenation operation along the channel direction. Assuming that the mapping function of the ith building block is $\mathcal{F}_b^i(\cdot)$, we then have:

$$\mathbf{x}_i = \mathcal{F}_b^i([\mathbf{x}_{i-1}, \ldots, \mathbf{x}_1, \mathbf{x}_0]), \quad i = 1, 2, \ldots, n, \tag{3}$$

where n is the number of building blocks in the network. Each block is connected to all preceding and subsequent blocks, and therefore facilitates the information propagation of the entire network. Iteratively, we can obtain the final output of all these stacked blocks:

$$\mathbf{x}_n = \mathcal{F}_b^n([\mathbf{x}_{n-1}, \ldots, \mathbf{x}_1, \mathbf{x}_0]). \tag{4}$$

$[\mathbf{x}_n, \ldots, \mathbf{x}_0]$ is further fused as the deep feature $\mathbf{x}_{\mathrm{DF}}$ through a 1 × 1 and a 3 × 3 conv layers, followed by a global skip connection (GSC) [8,14,18,19]:

$$\mathbf{x}_{\mathrm{DF}} = \mathbf{x}_0 + \mathcal{F}_c([\mathbf{x}_n, \ldots, \mathbf{x}_0]), \tag{5}$$

where $\mathcal{F}_c(\cdot)$ corresponds to the mapping function of two conv layers, as shown in Fig. 2. Subsequently, deep feature $\mathbf{x}_{\mathrm{DF}}$ is used to recover HR image $\mathbf{y}$ by the reconstruction subnetwork:

$$\mathbf{y} = \mathcal{F}_{rec}(\mathbf{x}_{\mathrm{DF}}) = \mathcal{F}_{up}(\mathbf{x}_{\mathrm{DF}}) + \hat{\mathbf{x}}, \tag{6}$$

where $\mathcal{F}_{up}(\cdot)$ represents the mapping function of the upscale module followed by a 3 × 3 conv layer, and $\hat{\mathbf{x}}$ is the (bicubic) interpolated version of $\mathbf{x}$. This is termed as an external skip connection (ESC) in [8], which approximates the residual between the original input and the final output of the network by interpolation [14], and further contributes to stabilizing model training.

As suggested in [18], we use the L_1 loss as the training objective. Given a training dataset $\mathcal{D} = \{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^{|\mathcal{D}|}$, where $|\mathcal{D}|$

Table 1
Statistics of network depth and model parameters ($c = 256$, $n = m = 4$).

	Simple 2D ($c_{in} = 1$)						Pseudo 3D ($c_{in} = 96$)					
B	$q = 2$			$q = 4$			$q = 2$			$q = 4$		
r	×2	×3	×4	×2	×3	×4	×2	×3	×4	×2	×3	×4
D	59	59	60	91	91	92	59	59	60	91	91	92
P	16.4	19.4	18.8	11.1	14.0	13.5	16.8	19.8	19.2	11.5	14.5	13.9

c_{in}: input channel; *B*: network branches; *D*: network depth; *P*: model parameters.

denotes the number of training examples in $\mathcal{D}$, the loss function is expressed as:

$$L(\boldsymbol{\theta}) = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \|\mathbf{y}^{(i)} - \mathcal{F}_{net}(\mathbf{x}^{(i)}; \boldsymbol{\theta})\|_1. \tag{7}$$

$\mathcal{F}_{net}(\cdot)$ denotes the function corresponding to the entire CSSFN network, and $\boldsymbol{\theta}$ is the set of model parameters. In terms of the overall model structure, our CSSFN is very similar to other typical SR models, mainly to make a relatively fair comparison and to highlight the role of our serial SLFF.

3.2. Serial stack over serial local feature fusion

The building block of our model, i.e., CSSFB, is structured in Fig. 3(a). At the beginning of each CSSFB, there is a 3 × 3 *channel compression* layer that is adopted to reduce the feature channel to a predefined value c, thereby improving the computational efficiency. According to (3), we have:

$$\mathbf{x}_{i-1,0} = \mathcal{H}_1^c([\mathbf{x}_{i-1}, \ldots, \mathbf{x}_1, \mathbf{x}_0]), \quad i = 1, 2, \ldots, n, \tag{8}$$

where $\mathcal{H}_1^c(\cdot)$ represents a convolutional layer with 1 × 1 kernel size and c output channels. This indicates that $\mathbf{x}_{i-1,0}$ keeps c channels. Subsequently, a series of stacked channel splitting and serial fusion units (CSSFUs) form the main part of our CSSFB, as shown in Fig. 3(a). Denoting the function of the jth CSSFU as $\mathcal{F}_u^j(\cdot)$, which we will describe in 3.3 in detail, we have the following formulation for the CSSFU:

$$\mathbf{x}_{i-1,j} = \mathcal{F}_u^j(\mathbf{x}_{i-1,j-1}), \quad j = 1, 2, \ldots, m, \tag{9}$$

where m denotes the number of CSSFUs in each CSSFB. We can also iteratively obtain the output of the last CSSFU $\mathbf{x}_{i-1,m}$:

$$\begin{aligned} \mathbf{x}_{i-1,m} &= \mathcal{F}_u^m(\mathbf{x}_{i-1,m-1}) \\ &= \mathcal{F}_u^m(\mathcal{F}_u^{m-1}(\cdots \mathcal{F}_u^1(\mathbf{x}_{i-1,0}) \cdots)). \end{aligned} \tag{10}$$

Local residual learning (LRL) [8,18,19,21] is another manner to stabilize model training. We also introduce LRL into our CSSFB modules, so the final output of the ith CSSFB can be expressed as:

$$\mathbf{x}_i = \mathbf{x}_{i-1,0} + \mathbf{x}_{i-1,m}. \tag{11}$$

3.3. Serial Local Feature Fusion (SLFF)

In the CSN model [8], features transmitted to a channel splitting block are first split into two branches with different local structures, which are then *parallelly* fused together with the merge-and-run (MAR) mapping [20,29]. The proposed CSSFN also splits features into several subfeatures. However, we do not transmit local information in a multibranch way. Instead, subfeatures are reintegrated into a single branch through convolutional and concatenation operations, which can be viewed as *partially* dense learning with channel splitting [30].

The input map of the jth CSSFU in the ith CSSFB is first split into q subfeatures equally, i.e., $\{\mathbf{x}^0_{i-1,j-1}, \ldots, \mathbf{x}^{q-1}_{i-1,j-1}\}$. Denote $\mathbf{z}^k_{i-1,j-1}$ as the output of the kth 3 × 3 conv layer in Fig. 3(b) (cubes in dark gray), which is followed by a ReLU layer. Then we have:

$$\mathbf{z}^k_{i-1,j-1} = \max\left\{0, \mathcal{H}_3^{c/q}\left([\mathbf{x}^{k-1}_{i-1,j-1}, \mathbf{z}^{k-1}_{i-1,j-1}]\right)\right\}, \tag{12}$$

where $k = 1, 2, \ldots, q$ and $\mathbf{z}^0_{i-1,j-1} = 0$. Therefore, all these subfeatures are reintegrated together and the network is in a single branch. Finally, we extend the channel of the last output feature, $\mathbf{z}^q_{i-1,j-1}$, by a 3 × 3 channel extension layer at the end of the CSSFU:

$$\mathbf{x}_{i-1,j} = \mathcal{H}_3^c\left(\mathbf{z}^q_{i-1,j-1}\right), \tag{13}$$

where $\mathbf{x}_{i-1,j}$ is the output of the jth CSSFU in the ith CSSFB. It is worth noting that the purpose of channel splitting in this work is not to form a multibranch structure but to be a preprocessing step for serial fusion. The single-branch structure makes the network deeper and narrower, also causing model training to be more unstable. This is part of the reason why we adopt DGFF to mitigate the training difficulty. Therefore, channel splitting and serial fusion can be regarded as "stretching" a shallower but wider network into a deeper but narrower one.

3.4. Network depth and parameters

Network depth is usually defined as the length of the longest path from the input to the output [8,20]. According to the entire structure of the proposed model, the depth of our CSSFN is given by:

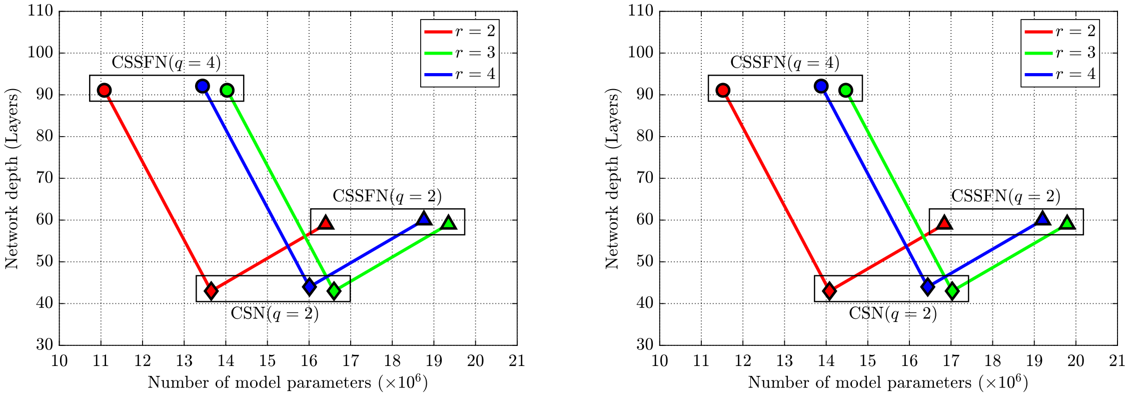
$$D = n[1 + m \times (q + 1)] + s + 6, \tag{14}$$

where s denotes the depth of the upscale modula and depends on the specific value of upsampling factor r. Specifically, $s = 1$ for $r = 2$ or $r = 3$, and $s = 2$ for $r = 4$. The first "1" in Eq. (14) corresponds to the compression layer at the beginning of each CSSFB, and the second one denotes the extension layer at the end of each CSSFU.

Table 1 exhibits the network depth (D) and parameters (P) of our CSSFN under several configurations, where pseudo 3D execution implies that the model regards 96 slices of a 3D MR volume as 96 channels of a 2D image. All models take approximately 10M∼20M parameters. The most similar model to our CSSFN is CSN [8], so we display the comparison of network configuration between CSN [8] and CSSFN in Fig. 5. It can be observed that CSSFN increases in network depth for both $q = 4$ and $q = 2$. However, it has fewer parameters when $q = 4$ and more parameters when $q = 2$ than CSN [8].

4. Experimental results

In this section, we first introduce the datasets used in the experiments and implementation details. Subsequently, we investigate and analyze the influence of network components on model performance, including GFF, LFF and the number of branches etc. Finally, the proposed method is quantitatively and qualitatively compared with other advanced SISR models, where the performance on 2D images, pseudo 3D images, and in-vivo images is



(a) 2D execution ($c_{in} = 1$)

(b) Pseudo 3D execution ($c_{in} = 96$)

Fig. 5. Comparison of model depth and parameters between CSN [8] and CSSFN. For all settings, we set $m = n = 4$ and $c = 256$. The symbols $\triangle$ and $\circ$ represent CSSFN with $q = 2$ and $q = 4$ respectively, and $\diamond$ denotes CSN [8] with 2 branches.

studied. The frequently used peak-signal-to-noise ratio (PSNR) and structural similarity index metric (SSIM) [40] are chosen as the metrics of quantitative evaluation.[1]

4.1. Datasets and implementation details

We employ the same MR datasets as in [8] to perform the experiments. They are derived from the IXI[2] dataset and contain the following three types of 3D brain MR volumes: T1-weighted, T2-weighted and PD-weighted images. Each type of MR volume contains 500, 70 and 6 samples for model training, testing and quick validation respectively. Two degradations, i.e., bicubic downsampling (**BD**) and k-space truncation (**TD**), are implemented to simulate LR images. For convenience, we follow the convention of [8] to indicate a particular subset of data, i.e., the subdataset with a specific type of MR images and degradation is denoted as the *dataset type* (*MR type*, *degradation type*). The size of each 3D volume is clipped to $240 \times 240 \times 96$, where 96 is the number of slices of a 3D volume. If the model takes a single slice of a 3D volume as input, we call it **common 2D execution**; if the model regards 96 slices as 96 channels of a 2D input, we term it as **pseudo 3D execution**. Note that the model parameters vary slightly with the number of input channels, as shown in Table 1.

The overall configuration of the proposed network is shown Figs. 2 and 3 with $c = 256$ and $m = n = q = 4$. The size of the minibatch is set to 16. The kernel size of each conv layer is marked in Figs. 2 and 3. For each convolution layer in CSSFU, we keep the channel size of the output feature the same as that of the subfeatures, i.e., c/q, except that the last channel extension layer has c output channels. For fair comparison, we also train our model with LR image patches of size 24×24 with their corresponding HR patches. The training patches are further augmented by random horizontal and vertical flips, and 90° rotations, as in [8,18,19,21]. All models are implemented in TensorFlow 1.11.0 and trained on a NVIDIA GeForce GTX 1080 Ti GPU for one million iterations. We apply Xavier's method [41] to initialize model parameters. The Adam [42] optimizer is used to minimize the L_1 loss with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. The learning rate is initialized as 10^{-4} for all layers and halved every 200k iterations.

4.2. Feature fusion

In this section, we study the effects of GFF and BIF. To this end, we designed several structures for ablation investigation. For GFF, we compare DGFF (Fig. 2) and concat global feature fusion (CGFF) [8,19]. For BIF, we compare our serial fusion (SF) strategy and MAR mapping [29] (Fig. 4(b)). Note that the latter is a local parallel fusion for multibranch structures. In addition, we also constructed a benchmark structure without either GFF or BIF, where the GFF is removed from the entire network and the part in the dotted box in Fig. 4(c) is replaced with the basic Conv + ReLU in Fig. 4(a). Table 2 shows the results of the comparison evaluated on $\mathcal{T}$(T2, BD), for SR×2. As seen, the benchmark without channel splitting (3rd column) achieves a PSNR of 39.90 dB, a relatively good result. This is probably because the stage mapping in the benchmark (refer to Fig. 4(a) and (c)) evolves into the residual block of EDSR.[3][18].

For convenience, we employ "0" and "1" to identify different GFF and BIF strategies: 0 for the benchmark, GFF1 for CGFF, GFF2 for DGFF, BIF1 for MAR (Fig. 4(b)), and BIF2 for SF (Fig. 4(c)). According to the 6th, 8th and 10th columns, we can observe that MAR mapping degrades model performance seriously when $q = 4$, which implies that one cannot improve the performance by simply increasing the branches of CSN [8]. In contrast, the proposed SF can boost model performance (7th, 9th and 11th columns), compared with the benchmark (3rd column). Another interesting finding is that CGFF performs significantly better than DGFF without channel splitting (4th column vs. 5th column). However, the situation is reversed with channel splitting (8th column vs. 10th column). This phenomenon shows that the combination of DGFF and SF can better promote the information flow of the network, thus improving the SR performance.

We also visualize the convergence process of these models in Fig. 6. It can be observed that these curves are consistent with the results in Table 2. It is worth noting that *although the curves for GFF0_BIF2, GFF1_BIF2, and GFF2_BIF2 seem to be similar, collecting the data points is very different*. As SF greatly increases the network scale, model training often collapses for SF and CGFF + SF, and we need to detect training interruptions and restart training via extra code. However, it rarely undergoes training crashes for DGFF + SF. Therefore, we believe that the increase in model scale (depth

[1] The source code will be released soon.

[2] http://brain-development.org/ixi-dataset/.

[3] Nevertheless, the model is still trainable because $m = n = 4$ indicates that the network parameters and depth are much smaller than those of the original EDSR [18].

Table 2
Ablation study on GFF and BIF. All models ($c = 256, m = n = q = 4$) are trained on $\mathcal{D}$(T2, BD) for one million iterations and tested on $\mathcal{T}$(T2, BD). The best and second best results are marked in **red** and **blue**, respectively.

GFF	CGFF	×	√	×	×	×	√	√	×	×
	DGFF	×	×	√	×	×	×	×	√	√
BIF	MAR	×	×	×	√	×	√	×	√	×
	SF	×	×	×	×	√	×	√	×	√
$r = 2$	PSNR	39.90	39.95	39.88	39.66	40.01	39.66	**40.03**	39.68	**40.05**
	SSIM	98.67	98.68	98.66	98.62	**98.69**	98.61	**98.69**	98.63	**98.70**

Remark: GFF1 — CGFF; GFF2 — DGFF; BIF1 — MAR; BIF2 — SF; PSNR: (dB); SSIM: (%).

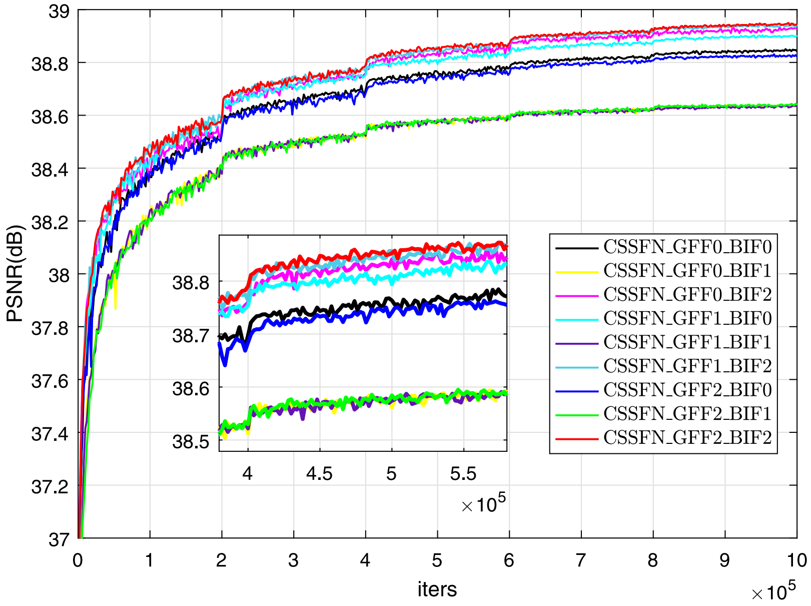


Fig. 6. Comparison of validation performance between different combinations of GFF and BIF. The PSNR curves are evaluated on $\mathcal{V}$(T2, BD) with $r = 2$ and correspond to the testing results in Table 2.

Table 3
The impact of the output width of serial fusion on model performance. All models are trained on $\mathcal{D}$(PD, BD) for one million iterations. The basic configuration is $c = 256$ and $m = n = 4$ (PSNR|SSIM|P|D).

c_0	r	$q = 2$	$q = 4$	$q = 8$	$q = 16$
c/q	2	41.45|98.98|16.40|59	41.30|98.96|11.09|91	41.20|98.93|7.990|155	40.99|98.89|6.341|283
	3	36.15|97.11|19.35|59	36.00|97.03|14.04|91	35.83|96.90|10.95|155	35.56|96.73|9.294|283
	4	33.71|95.20|18.76|60	33.59|95.09|13.45|92	33.38|94.84|10.35|156	33.07|94.47|8.703|284
64	2	41.32|98.95|9.911|59	41.30|98.96|11.09|91	41.35|98.96|13.46|155	41.28|98.95|18.18|283
	3	36.06|97.06|12.86|59	36.00|97.03|14.04|91	36.02|97.04|16.41|155	35.99|97.01|21.13|283
	4	33.57|95.06|12.27|60	33.59|95.09|13.45|92	33.59|95.06|15.82|156	33.59|95.08|20.54|284

Remark: r – scaling factor; PSNR: (dB); SSIM: (%); P: (M); D: layers.

and parameters) aggravates instable training and that absorbing DGFF helps to ease such training difficulty. Both the quantitative results in Table 2 and the PSNR curves in Fig. 6 demonstrate the effectiveness and benefits of our SF and its combination with DGFF.

4.3. Channel splitting

In previous settings, the output width of serial fusion, c/q, will be changed according to the number of subfeatures q. At the same time, model parameter P decreases as q increases. However, if we set the output channel of serial fusion to a fixed value, then P increases with the increase of q. Denoting the output channel of serial fusion as c_o, we study the effects of q and c_o on model performance in this section. To this end, we train our CSSFN with different configurations with $\mathcal{D}$(PD, BD) and collect the results in Table 3.

4.3.1. Unfixed output width

Unfixed output width implies $c_o = c/q$, where c is fixed and represents the channel number of input features to serial fusion. It is a general consensus that increasing network depth D and parameters P helps to improve the performance of SR models. According to Table 3, we can see that D increases and P decreases with the increase in q. On the other hand, model performance worsens as q increases. It seems that, in serial fusion, the impact of model parameter P is more pronounced than that of network depth D. However, the inference contradicts the channel splitting that deceases P of the baseline. Hence, we further conduct the following experiment with fixed q.

4.3.2. Fixed output width

We set $c_o = 64$ for comparison in this case. As shown in Table 3, we cannot obtain a significant performance gain as q increases. This strange phenomenon is puzzling because both P and

Table 4
Quantitative comparison between different methods. The best values are marked in red and the second ones are marked in blue (PSNR/SSIM).

Method	r	Bicubic downsampling $\mathcal{T}(:, \text{BD})$			k-space truncation $\mathcal{T}(:, \text{TD})$		
		PD	T1	T2	PD	T1	T2
Bicubic	×2	35.04/0.9664	33.80/0.9525	33.44/0.9589	34.65/0.9625	33.38/0.9460	33.06/0.9541
NLM [43]	×2	37.26/0.9773	35.80/0.9685	35.58/0.9722	36.18/0.9707	34.71/0.9581	34.56/0.9641
SRCNN [11]	×2	38.96/0.9836	37.12/0.9761	37.32/0.9796	38.23/0.9802	36.52/0.9705	37.04/0.9773
ESPCNN [16]	×2	38.27/0.9814	36.91/0.9747	36.92/0.9773	37.88/0.9792	36.35/0.9693	36.79/0.9754
VDSR [14]	×2	39.97/0.9861	37.67/0.9783	38.65/0.9836	39.89/0.9850	37.58/0.9760	38.74/0.9823
IDN [44]	×2	40.27/0.9869	37.79/0.9787	39.09/0.9846	40.43/0.9862	37.79/0.9765	39.48/0.9842
RDN [19]	×2	40.31/0.9870	37.95/0.9795	38.75/0.9838	40.39/0.9862	38.08/0.9784	40.02/0.9826
RecNet [39]	×2	40.43/0.9873	37.86/0.9792	39.13/0.9848	40.10/0.9857	37.54/0.9764	39.03/0.9832
FSCWRN [34]	×2	40.72/0.9880	37.98/0.9797	39.44/0.9855	40.91/0.9876	38.04/0.9786	39.82/0.9851
CSN [8]	×2	41.28/0.9895	38.27/0.9810	39.71/0.9863	41.77/0.9897	38.62/0.9813	40.47/0.9868
CSSFN-B4 [Ours]	×2	41.30/0.9896	38.33/0.9812	40.05/0.9870	41.91/0.9900	38.67/0.9815	40.64/0.9872
CSSFN-B2 [Ours]	×2	41.45/0.9898	38.36/0.9813	40.10/0.9871	41.97/0.9902	38.76/0.9818	40.73/0.9874
Bicubic	×3	31.20/0.9230	30.15/0.8900	29.80/0.9093	30.88/0.9167	29.79/0.8793	29.50/0.9016
NLM [43]	×3	32.81/0.9436	31.74/0.9216	31.28/0.9330	32.02/0.9324	30.83/0.9027	30.57/0.9197
SRCNN [11]	×3	33.60/0.9516	32.17/0.9276	32.20/0.9440	32.90/0.9432	31.72/0.9187	31.80/0.9381
ESPCNN [16]	×3	33.52/0.9505	32.10/0.9242	32.13/0.9421	32.54/0.9417	31.52/0.9140	31.64/0.9353
VDSR [14]	×3	34.66/0.9599	32.91/0.9378	33.47/0.9559	34.27/0.9555	32.57/0.9304	33.23/0.9515
IDN [44]	×3	34.96/0.9619	33.06/0.9394	33.92/0.9591	34.88/0.9598	32.86/0.9348	33.95/0.9569
RDN [19]	×3	35.08/0.9628	33.31/0.9430	33.91/0.9591	35.00/0.9609	33.33/0.9416	33.99/0.9576
RecNet [39]	×3	34.96/0.9623	33.05/0.9399	33.85/0.9588	34.67/0.9590	32.80/0.9347	33.69/0.9554
FSCWRN [34]	×3	35.37/0.9653	33.24/0.9423	34.27/0.9618	35.30/0.9636	33.09/0.9390	34.34/0.9603
CSN [8]	×3	35.87/0.9693	33.53/0.9464	34.64/0.9647	36.09/0.9697	33.68/0.9464	34.95/0.9653
CSSFN-B4 [Ours]	×3	35.99/0.9702	33.56/0.9468	34.84/0.9661	36.23/0.9706	33.73/0.9469	35.12/0.9663
CSSFN-B2 [Ours]	×3	36.15/0.9711	33.59/0.9471	34.96/0.9668	36.32/0.9713	33.75/0.9472	35.23/0.9671
Bicubic	×4	29.13/0.8799	28.28/0.8312	27.86/0.8611	28.82/0.8713	27.96/0.8182	27.60/0.8511
NLM [43]	×4	30.27/0.9044	29.31/0.8655	28.85/0.8875	29.27/0.8906	28.68/0.8439	28.37/0.8718
SRCNN [11]	×4	31.10/0.9181	29.90/0.8796	29.69/0.9052	30.52/0.9078	29.31/0.8616	29.32/0.8960
ESPCNN [16]	×4	31.02/0.9169	29.77/0.8781	29.32/0.9022	30.22/0.9034	29.29/0.8618	29.28/0.8954
VDSR [14]	×4	32.09/0.9311	30.57/0.8932	30.79/0.9240	31.69/0.9244	30.14/0.8818	30.51/0.9162
IDN [44]	×4	32.47/0.9354	30.74/0.8966	31.37/0.9312	32.33/0.9318	30.40/0.8889	31.31/0.9270
RDN [19]	×4	32.73/0.9387	31.05/0.9042	31.45/0.9324	32.64/0.9362	31.00/0.9018	31.49/0.9301
RecNet [39]	×4	32.58/0.9378	30.86/0.9005	31.30/0.9310	32.16/0.9310	30.46/0.8900	31.03/0.9243
FSCWRN [34]	×4	32.91/0.9415	30.96/0.9022	31.71/0.9359	32.78/0.9387	30.79/0.8973	31.71/0.9334
CSN [8]	×4	33.40/0.9486	31.23/0.9093	32.05/0.9413	33.51/0.9489	31.27/0.9092	32.28/0.9421
CSSFN-B4 [Ours]	×4	33.60/0.9509	31.34/0.9102	32.27/0.9441	33.64/0.9501	31.35/0.9095	32.46/0.9440
CSSFN-B2 [Ours]	×4	33.71/0.9520	31.37/0.9104	32.38/0.9453	33.75/0.9514	31.39/0.9098	32.57/0.9453

D become larger with the increase of q, but model performance is not improved or even slightly decreased (e.g., SR×3). Therefore, simply enlarging the number of subfeatures q in terms of channel splitting cannot boost the performance, and we speculate that *the network is over/underfitted with the degradation of training samples*. In this case, channel splitting itself is helpful in mitigating model over/underfitting. However, once channel splitting is performed, a larger q will exacerbate the over/underfitting effect, resulting in performance degradation. When c_0 is fixed, the adverse effect caused by an increase in q counteracts the benefit of an increase in model parameters P. This is why we cannot observe performance gain in this case. Additionally, increasing q actually makes it more difficult to integrate subfeatures effectively. Therefore, channel splitting should *not* aggressively enlarge the number of subfeatures unless there exists a more effective information fusion mechanism.

4.4. Comparison with other methods

To illustrate the effectiveness and superiority of our CSSFN, we compare it with several typical SISR methods in this section, including the following:

- a classic method for MR image upsampling: NLM [43];
- two lightweight CNN models: SRCNN [11] and ESPCNN [16];
- two moderate-scale CNN models: VDSR [14] and IDN [44];
- two large-scale CNN models: RDN [19] and EDSR [18];
- two CNN models specifically for MR images: FSCWRN [34], CSN [8];
- a UNet-based model specifically for MR image recovery: RecNet [39];

Some results are directly cited from [8] because of the same implementation details and datasets, while others are obtained by retraining the corresponding models with the datasets described in 4.1. Note that we do not compare two recent works, W^{2}AMSN [35] and SERAN [36], since they use 32 × 32 LR patches to train the models, while all the compared methods in this paper adopt 24 × 24 LR image patches.

4.4.1. Bicubic Degradation (BD)

As one of the most common degradations for simulating LR images, bicubic degradation simply utilizes a bicubic interpolation kernel to shrink the image size in the spatial domain. Table 4 shows the quantitative results over the testing datasets of PD, T1 and T2 images for all scales, i.e., $\mathcal{T}(:, \text{BD})$. It can be seen that our CSSFN surpasses CSN [8] and achieves the best performance on all MR image types and scaling factors. In particular, CSSFN-B4 with relatively few parameters also gives excellent performance on all image types and scales.

Fig. 7 exhibits a visual comparison of these methods on PD, T1 and T2 images with SR×4. We can see that most CNN-based methods (e.g., VDSR [14] and RDN [19]) can surpass traditional SR methods (bicubic and NLM [43]) by a large margin, achieving good approximations to the ground truth. Although it is not easy to observe the difference between our model and other CNN-based methods on PD and T1 images, the proposed CSSFN is the most accurate in quantitative evaluation. Besides, we can see that the results of our CSSFN show sharper edges from the position

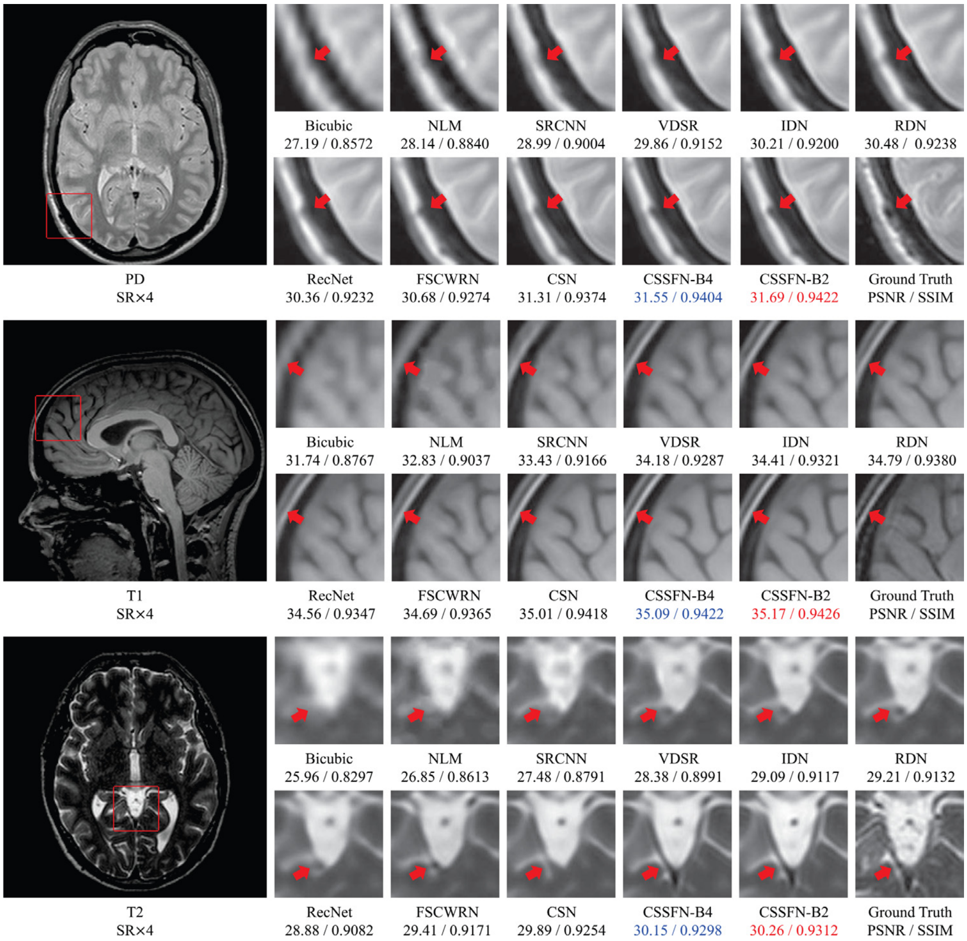


Fig. 7. Visual comparison between several advanced SISR methods on a PD (top), T1 (middle) and T2 (bottom) image with $r = 4$. The image degradation is **bicubic degradation**. The maximal PSNR (dB) and SSIM for each group of comparisons are in red and the second ones are in blue.

indicated by the red arrows marked on the T1 image, and the dark trench between the two bright curves is also clearer than other methods. A more significant visual difference can be observed on the T2 image. The gap between the white and gray areas is more obvious in the results of our model, but it can hardly be observed in the results of other methods. In addition, the black spot in the white area can be more clearly observed in the results of the CSSFN.

4.4.2. Truncation Degradation (TD)

k-space truncation of HR images is a process to simulate the real MR image acquisition where a LR image is scanned by reducing acquisition lines in phase and encoding directions [8]. When the scaling factor remains the same, *k*-space truncation often degrades the quality of HR images more radically than bicubic downsampling due to the "steep" loss of *k*-space data. This can be verified by the fact that bicubic interpolation performs better than *k*-space truncation in bicubic downsampling (3rd and 6th columns of Table 4). Table 4 also presents the quantitative results of the compared methods in terms of *k*-space truncation degradation. Again, our CSSFN obtains better performance on all image types and scaling factors. Interestingly, the performance of CSSFN (or CSN [8]) is better than that of bicubic downsampling, which may imply that the proposed CSSFN is more suitable for MR image SR tasks.

Visually, we can observe the advantages of the proposed method over other methods more easily. Fig. 8 shows the visual effect of the compared methods on PD, T2 and T1 images with $r = 4$, $r = 3$ and $r = 4$, respectively. The proposed model recovers images with clearer and sharper edges, thus making them more faithful to the ground truth HR images. In this case (TD), the visual advantage of our CSSFN is more observable. For instance, it displays clearer contours and details at the locations indicated by the red arrows in the PD and T2 images. In particular, CSSFN-B2 has more obvious advantages and is closer to the ground truth. In the T1 image, the superiority of our CSSFN is more significant. The dark trench that can be clearly identified in the results of our model, can hardly be marked in the results of other methods.

4.4.3. Pseudo 3D execution

One of the major problems in training large-scale models with MR images is the degradation of training samples. This problem can be alleviated by pseudo 3D execution at the cost of appropriate drop in accuracy [8]. We also conduct experiments with pseudo 3D case to make the comparison more comprehensive and thorough. Note that we do not include some models because the reduction of training samples makes the training of most other models fail. Table 5 shows the quantitative results of EDSR [18], CSN [8], CSSFN-B4 and CSSFN-B2 in this case, with both degradations. The performance gain of the proposed CSSFN is obvious. However, the advantage of CSSFN-B2 over CSSFN-B4 seems to be weakened when compared with common 2D execution, such as $\mathcal{T}(\text{PD, BD})$ with $r = 4$ and $\mathcal{T}(\text{T1, TD})$ with $r = 3$. This is mainly due to the over/underfitting caused by the reduction in training samples, as shown in Fig. 9. It can be seen that CSSFN-B2 essentially performs better than CSSFN-B4 in that it converges faster and has higher PSNR maxima. Besides, CSN [8] alone does not show obvious over/underfitting in all cases. To

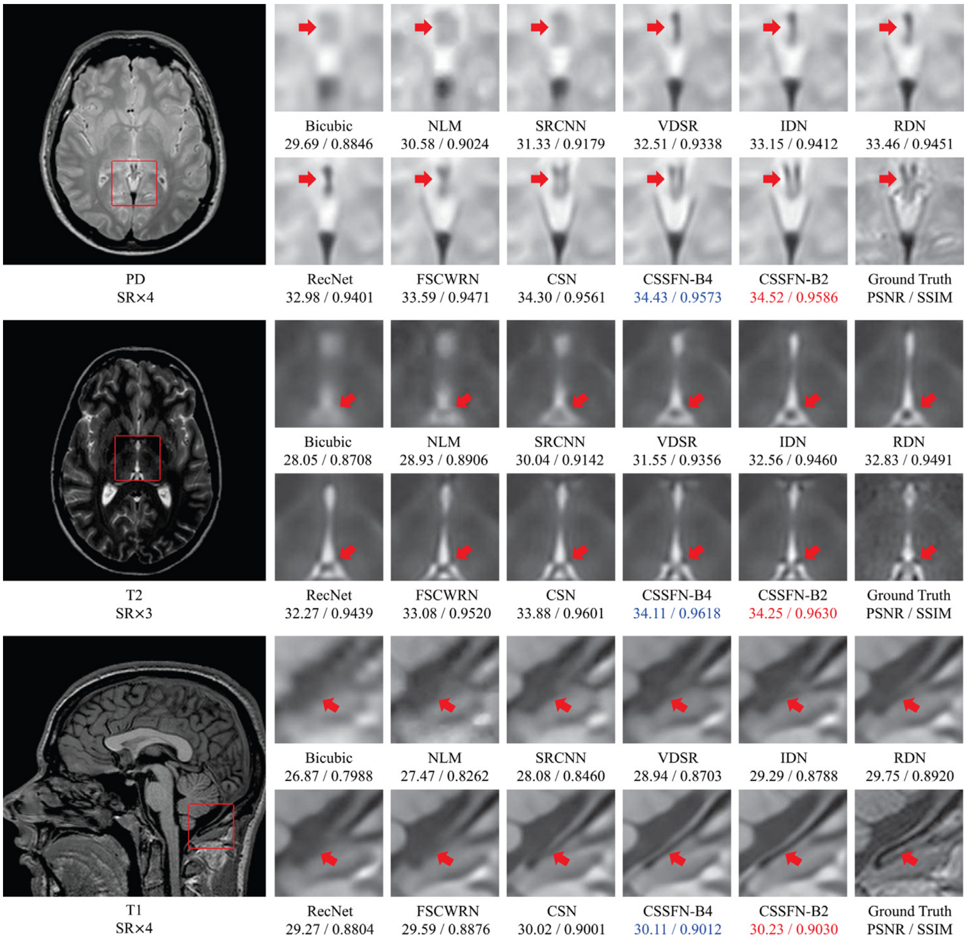


Fig. 8. Visual comparison between several state-of-the-art SISR methods on a PD (top), T2 (middle) and T1 (bottom) image with $r = 4$, $r = 3$ and $r = 4$, respectively. The image degradation is **truncation degradation**. The maximal PSNR (dB) and SSIM for each group of comparisons are in red, and the second ones are in blue.

Table 5
Quantitative comparison in case of pseudo 3D execution. The maximal PSNR (dB) and SSIM of each comparative group are marked in red, and the second ones are in blue.

	r	Bicubic downsampling $\mathcal{T}(:, \text{BD})$				k-space truncation $\mathcal{T}(:, \text{TD})$)			
		EDSR [18]	CSN [8]	CSSFN ($q = 4$)	CSSFN ($q = 2$)	EDSR [18]	CSN [8]	CSSFN ($q = 4$)	CSSFN ($q = 2$)
PD	×2	39.87/0.9857	40.15/0.9865	40.28/0.9869	40.34/0.9871	39.47/0.9837	39.50/0.9839	39.80/0.9849	39.91/0.9853
	×3	34.39/0.9578	34.68/0.9598	34.78/0.9609	34.76/0.9611	33.97/0.9531	34.12/0.9540	34.24/0.9554	34.15/0.9550
	×4	31.80/0.9284	32.19/0.9325	32.21/0.9332	32.11/0.9329	31.44/0.9219	31.72/0.9246	31.78/0.9252	31.68/0.9257
T1	×2	37.56/0.9774	37.60/0.9778	37.74/0.9786	37.81/0.9789	37.09/0.9741	36.99/0.9737	37.18/0.9748	37.25/0.9754
	×3	32.76/0.9347	32.83/0.9360	32.86/0.9362	32.85/0.9366	32.27/0.9274	32.25/0.9266	32.34/0.9276	32.32/0.9275
	×4	30.46/0.8902	30.53/0.8915	30.58/0.8919	30.61/0.8923	30.04/0.8803	30.07/0.8794	30.09/0.8795	30.14/0.8812
T2	×2	38.28/0.9824	38.53/0.9831	38.79/0.9836	38.92/0.9842	38.11/0.9803	38.20/0.9807	38.54/0.9817	38.92/0.9842
	×3	33.15/0.9528	33.36/0.9547	33.46/0.9556	33.50/0.9559	32.89/0.9482	33.00/0.9490	33.21/0.9512	33.26/0.9518
	×4	30.52/0.9198	30.81/0.9231	30.93/0.9242	30.89/0.9241	30.31/0.9137	30.54/0.9163	30.62/0.9182	30.58/0.9178

this end, the increase in model scale tends to cause the fitting problem when trained with degraded samples, which is not easy to observe through CSN [8]. Fig. 10 presents the visual comparison between these methods in terms of the pseudo 3D case, from which we can also see that our CSSFN provides a clearer indication of underlying structures, compared with other methods. Despite the over/underfitting effects, our CSSFNs still outperform EDSR [18] and CSN [8], indicating their excellent structure design and powerful representational capacity, especially in the presence of degraded training samples.

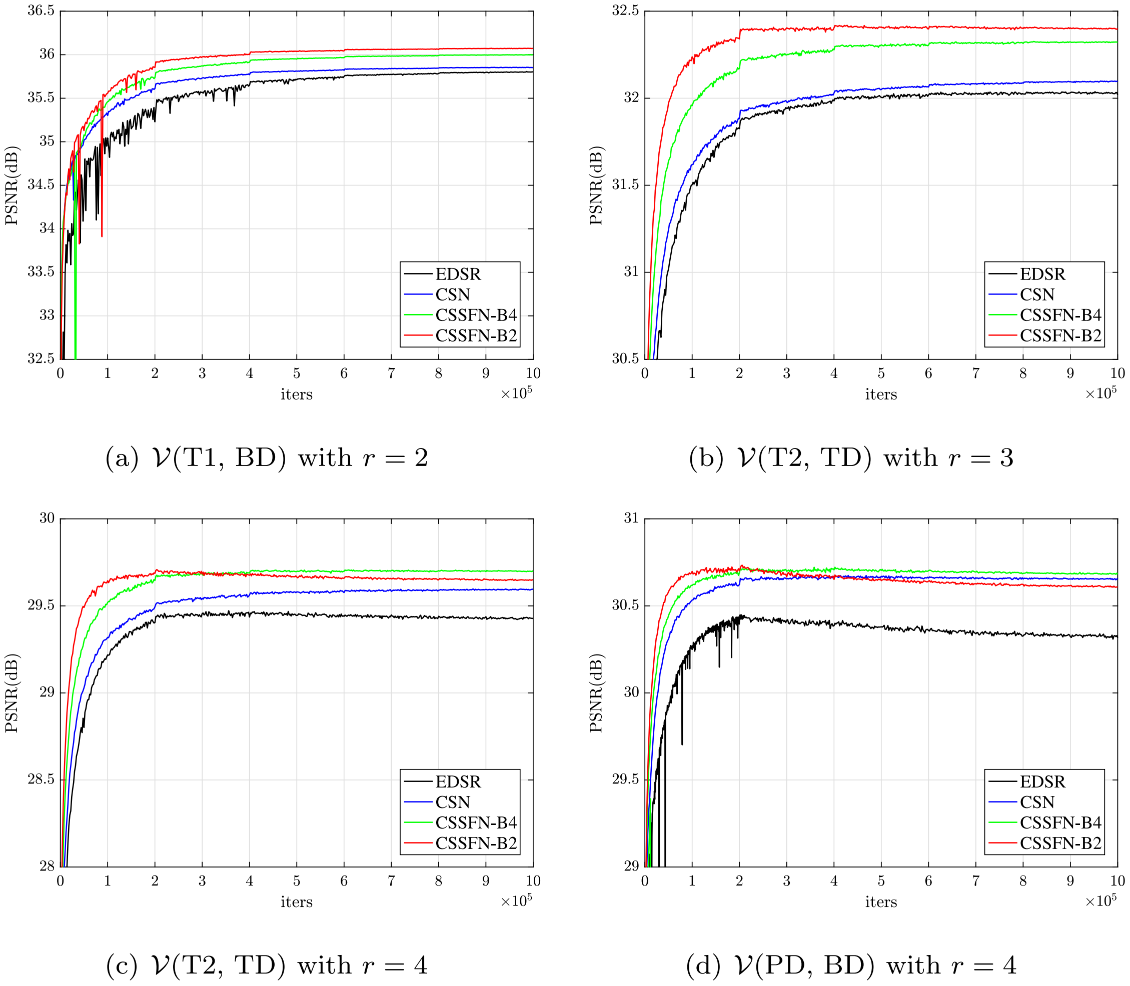
4.4.4. Performance on in-vivo images

We also conducted SR experiments on two in-vivo T1 images collected from Alltech Medical Systems Co., LTD. In this case, the ground truth is not available and image degradation is not known either. We compare our CSSFN-B4 with NLM [43], SRCNN [11], VDSR [14], RDN [19], FSCWRN [34], and CSN [8] etc. As shown in Fig. 11, our CSSFN-B4 recovers sharper edges and finer details than other state-of-the-art methods, as indicated by the arrows.

5. Discussion

5.1. Channel discrimination capacity

In CSN [8], channel discrimination capacity is achieved by different branch structures. The hierarchical features are divided into 2 parts by channel splitting and fused together by merge-and-run mapping [29]. The propagation paths of subfeatures have



(a) $\mathcal{V}$(T1, BD) with $r = 2$

(b) $\mathcal{V}$(T2, TD) with $r = 3$

(c) $\mathcal{V}$(T2, TD) with $r = 4$

(d) $\mathcal{V}$(PD, BD) with $r = 4$

Fig. 9. Validation performance comparison of the compared methods on several randomly selected subdatasets in the case of pseudo 3D execution. It can be observed that the severity of model performance degradation due to over/underfitting: (a) < (b) < (c) < (d).

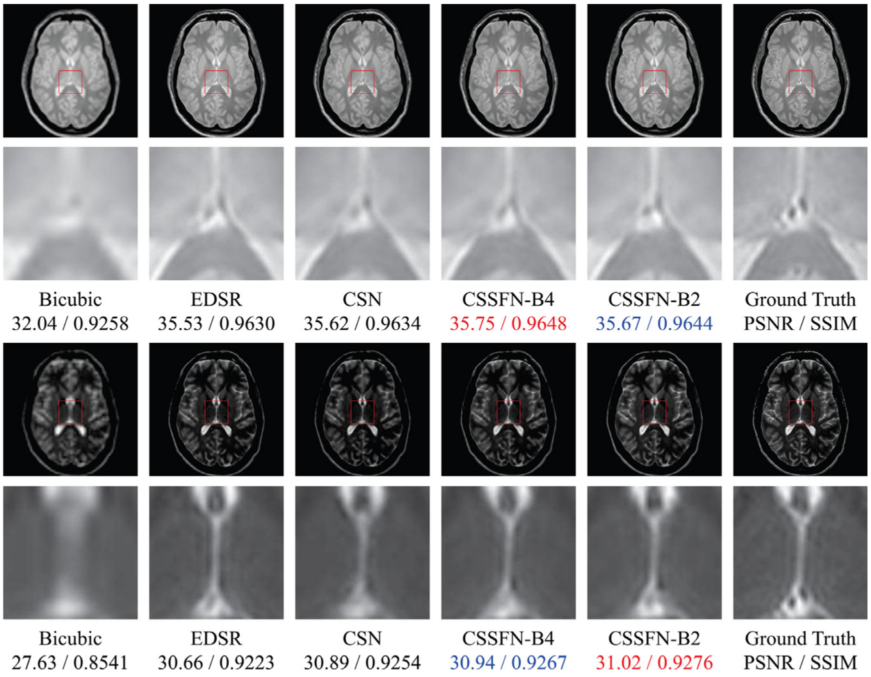


Fig. 10. Visual comparison of pseudo 3D execution on a PD (top) and a T2 (bottom) image with $r = 3$ and $r = 4$, respectively. Image degradation is **TD**.

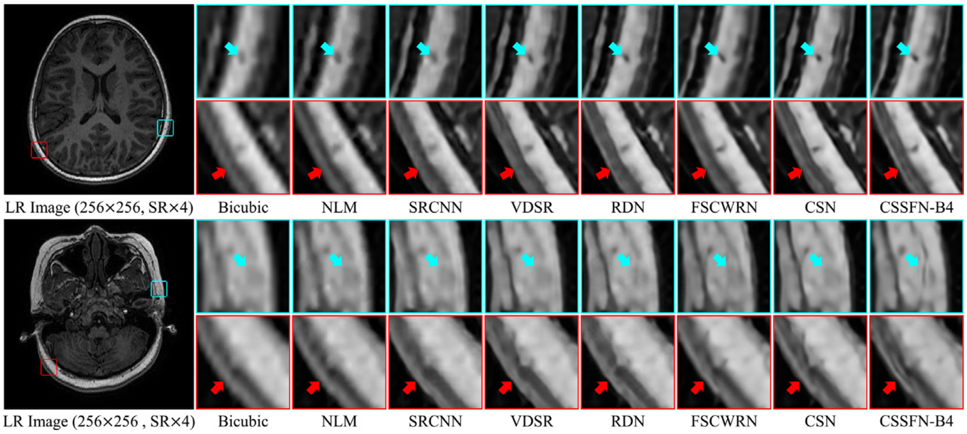


Fig. 11. Visual comparison between several state-of-the-art SISR methods on in-vivo T1 MR images with $r = 4$. In this case, the ground truth HR images are not available.

different structures. In the proposed CSSFN, the subfeatures are also processed discriminatorily in that they are located at different depths of the network; although, they are placed in a single branch. This can be regarded as a *fine-grained hierarchy* of intermediate features, thus realizing a partial continuous memory mechanism [19], which is believed to be beneficial to feedback propagation [38]. In this regard, it makes subfeature fusion of channel splitting more effective. On the other hand, network depth is crucially important for the representational capacity of deep models [45,46]. The SLFF in our CSSFN significantly increases the network depth of CSN [8], which is one of the main reasons that it can achieve better performance.

From Table 3, we can see that model performance degrades with the increase of subfeatures (q) when $c_o = c/q$. However, the increase in model parameter P leads to no performance gain when c_o remains fixed at 64. As discussed earlier, in addition to the over/underfitting, the difficulty of information fusion may also impede the capacity of channel discrimination. If more effective mechanisms for information fusion are explored, it is possible to achieve a better trade-off between network performance and scale.

5.2. Feature integration

Network structure design for deep models is very important for the high-level representation of data and the final performance of the task. In the context of channel splitting, it is crucial to find an effective subfeature integration strategy to promote the trade-off between the performance and efficiency of MR image SR. Due to the degradation of training samples, the basic principle is to improve the representational ability of the network while keeping it trainable. Possible options include (1) an attention mechanism, (2) diverse multiscale features, and (3) further simplification of operations between subfeatures, such as element-wise summation or 1 × 1 conv. However, these possible strategies mainly rely on the professional knowledge and experience of researchers, and it is difficult for people to alter their original thinking paradigm and present an optimal choice. Another promising idea may be the methods based on neural architecture search (NAS) [47,48], which can be used to automatically search the optimal design.

5.3. Image degradation model

We also investigate two degradations as in [8], i.e., bicubic downsampling and k-space truncation. Truncation degradation is considered more aggressive because the information outside the sampling range is "steeply" cut off without any cushion, as shown in Fig. 14. As mentioned earlier, it can be verified by the fact that bicubic interpolation performs better in bicubic downsampling than in k-space truncation. However, in Table 4, the performance gain of some models (e.g., CSN [8] and CSSFN) is contrary to that of bicubic interpolation. On the other hand, truncation degradation simulates real MRI acquisition, which operates in k-space and truncates the frequency spectrum of imaging objects or scenes. This indicates that these models may be more suitable for the scenarios of MR image SR due to stronger representational capacity.

5.4. Limitations

The proposed CSSFN may fail to recover the underlying structure correctly when the MR image to be super-resolved contains complex textures. As shown in Fig. 12, a T2 image is SR × 3 magnified under **TD**. Although our CSSFNs present the best quantitative and qualitative results, it, like the other compared methods, fails to recover the correct structure indicated by the red circles. The possible reasons for the failure case are threefold. First, the most important reason may be that the degradation of MR samples limits the representational capacity of deep models. Second, we use 24 × 24 LR image patches to train the models, which is too small to capture adequate global features. Finally, complex textures may need more support from global features, whereas our model focuses on local features without nonlocal attention. Consequently, unpleasant results are shown in these challenging situations.

5.5. Efficiency compromise

To demonstrate the efficiency superiority of our CSSFNs to other compared models, especially CSN [8], we introduce MultiAdds [49] as another quantitative evaluation index. For conv layers, MultiAdds [49] is computed as follows:

$$\text{MultiAdds} = k \times k \times C_{in} \times C_{out} \times H \times W \tag{15}$$

where k is the size of the conv kernel, C_{in} and C_{out} are the input and output channels of this layer, and H and W denote the spatial size of output features. For fair comparison, we follow the convention of [49] and assume the size of HR images to be 720p (1280 × 720). Fig. 13 shows the compromise study of PSNR vs. MultiAdds (Giga). It can be observed that CSSFN-B2 consumes the most MultiAdds to achieve the best performance for all scales, but CSSFN-B4 always provides better trade-off between PSNR and MultiAdds than the CSN [8]. More specifically, our CSSFN-B4 consistently achieves higher PSNR values than the CSN [8] with less computational overhead.

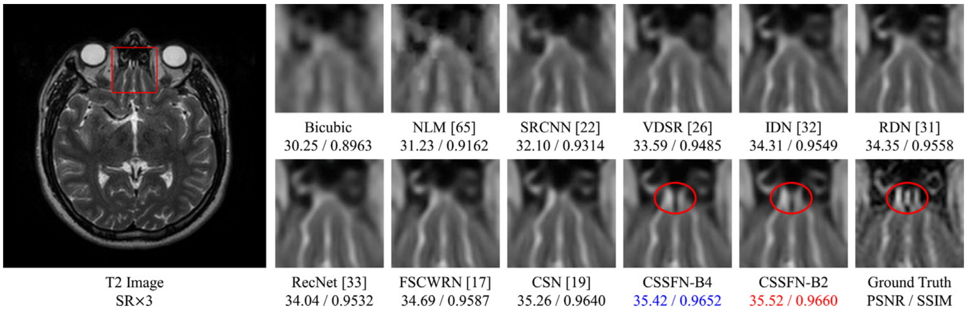
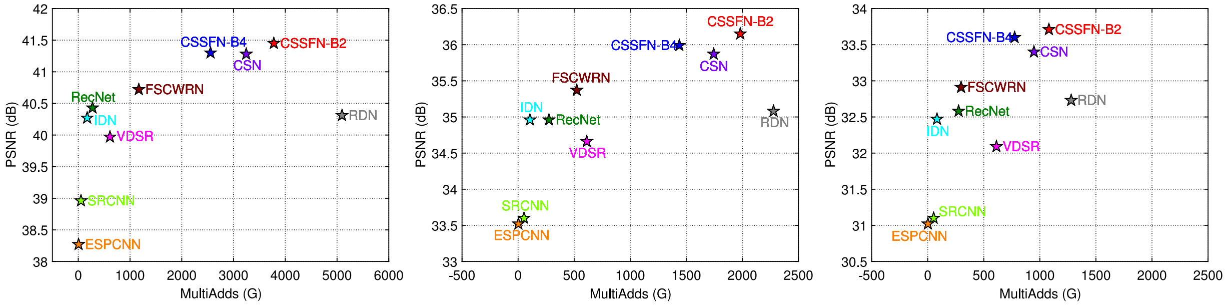
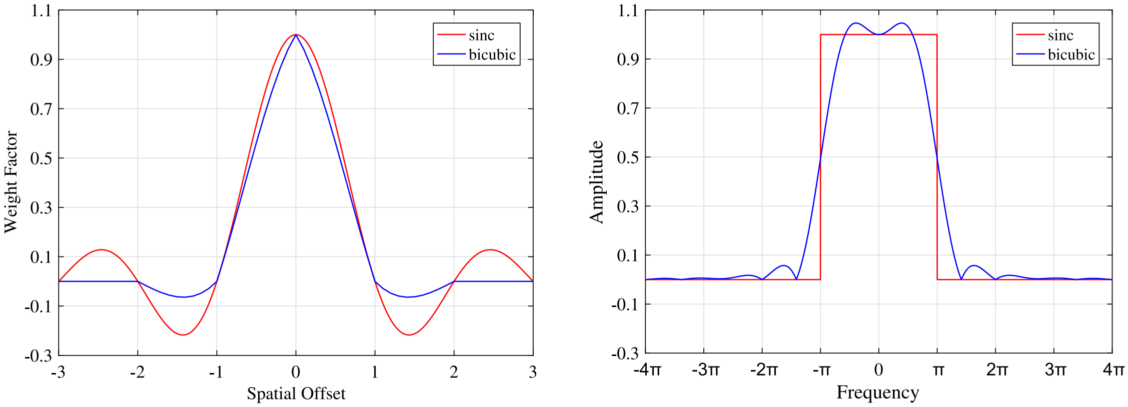


Fig. 12. Failure case for super-resolution (SR×3) on a T2 slice. The image degradation is **truncation degradation**. The maximal PSNR (dB) and SSIM for each group of comparisons are in red and the second ones are in blue.



(a) SR×2 (b) SR×3 (c) SR×4

Fig. 13. Efficiency analysis on computational complexity of compared models. We follow the convention of [49] and assume the size of HR images to be 720p (1280 × 720) to calculate MultiAdds. The data are collected on $\mathcal{T}(\mathbf{PD}, \mathbf{BD})$.



(a) Spatial domain (b) Frequency domain

Fig. 14. Comparison between bicubic downsampling and k-space truncation in the spatial domain (a) and frequency domain (b). The truncation in the frequency domain corresponds to a sinc function in the spatial domain (1D).

6. Conclusion

Channel discrimination is an effective manner to improve the performance of deep SR models in the context of degradation of training samples, and channel splitting is a simple and direct implementation for dealing with feature channels discriminatively. However, further study is needed on the integration of the subfeatures produced by channel splitting and ease model fitting problems. In this work, we demonstrate a serial fusion strategy for channel splitting. Hierarchical features are first divided into multiple subfeatures along the channel direction and then integrated into a single branch in a serial way. These subfeatures also assign the model channel discrimination capacity in that each subfeature is located at different network depths. To improve the information flow through the network and avoid training instability, we combine a serial fusion strategy with DGFF to fuse intermediate features. Extensive experiments demonstrate the superiority of our CSSFN to other advanced SISR methods. Additionally, serial fusion might shed some light on other feature fusion and channel discrimination methods. In the future, we would like to develop more effective strategies for deep feature fusion and lightweight SR models, improving the compromise between the performance and resource consumption.

CRediT authorship contribution statement

Xiaole Zhao: Conceptualization, Methodology, Software, Validation, Roles/Writing – original draft. **Yulun Zhang:** Roles/Writing – original draft, Formal analysis, Investigation. **Yun Qin:** Writing – review & editing, Data curation. **Qian Wang:** Data curation, Visualization. **Tao Zhang:** Methodology, Supervision, Resources. **Tianrui Li:** Funding acquisition, Resources, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] O. Oktay, W. Bai, M. Lee, et al., Multi-input cardiac image super-resolution using convolutional neural networks, in: International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI), 2016, pp. 246–254.

[2] E. Plenge, D.H.J. Poot, M. Bernsen, et al., Super-resolution methods in MRI: can they improve the trade-off between resolution, signal-to-noise ratio, and acquisition time? Magn. Reson. Med. 68 (6) (2012) 1983–1993.

[3] R. Keys, Cubic convolution interpolation for digital image processing, IEEE Trans. Acoust. Speech Signal Process. 29 (6) (1981) 1153–1160.

[4] J. Sun, Z. Xu, H.-Y. Shum, Image super-resolution using gradient profile prior, in: International Conference on Computer Vision and Pattern Recognition (CVPR), 2008, pp. 1–8.

[5] S. Peled, Y. Yeshurun, Super-resolution in MRI: application to human white matter fiber tract visualization by diffusion tensor imaging, Magn. Reson. Med. 45 (1) (2015) 29–35.

[6] W.T. Freeman, T.R. Jones, E.C. Pasztor, Example-based super-resolution, IEEE Comput. Graph. Appl. 22 (2) (2002) 56–65.

[7] J. Yang, J. Wright, T.S. Huang, Y. Ma, Image super-resolution as sparse representation of raw image patches, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2008, pp. 1–8.

[8] X. Zhao, Y. Zhang, T. Zhang, X. Zou, Channel splitting network for single MR image super-resolution, IEEE Trans. Image Process. 28 (11) (2019) 5649–5662.

[9] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.

[10] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.

[11] C. Dong, C.C. Loy, K. He, X. Tang, Image super-resolution using deep convolutional networks, IEEE Trans. Pattern Anal. Mach. Intell. 38 (2) (2016) 295–307.

[12] J. Kim, J.K. Lee, K.M. Lee, Deeply-recursive convolutional network for image super-resolution, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1637–1645.

[13] Y. Tian, Y. Zhang, Y. Fu, C. Xu, TDAN: Temporally-deformable alignment network for video super-resolution, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 3360–3369.

[14] J. Kim, J.K. Lee, K.M. Lee, Accurate image super-resolution using very deep convolutional networks, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1646–1654.

[15] Y. Tai, J. Yang, X. Liu, C. Xu, MemNet: A persistent memory network for image restoration, in: IEEE International Conference on Computer Vision (ICCV), 2017, pp. 4549–4557.

[16] W. Shi, J. Caballero, F. Huszar, et al., Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1874–1883.

[17] C. Ledig, L. Theis, F. Huszar, et al., Photo-realistic single image super-resolution using a generative adversarial network, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 105–114.

[18] B. Lim, S. Son, H. Kim, S. Nah, K.M. Lee, Enhanced deep residual networks for single image super-resolution, in: IEEE International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2017, pp. 1132–1140.

[19] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, Y. Fu, Residual dense network for image super-resolution, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 2472–2481.

[20] Y. Hu, X. Gao, J. Li, Y. Huang, H. Wang, Single image super-resolution via cascaded multi-scale cross network, 2018, arXiv preprint arXiv:1802.08808.

[21] Y. Zhang, K. Li, K. Li, et al., Image super-resolution using very deep residual channel attention networks, in: Europeon Conference on Computer Vision (ECCV), 2018, pp. 294–310.

[22] Y. Qiu, R. Wang, D. Tao, J. Cheng, Embedded block residual network: A recursive restoration model for single-image super-resolution, in: IEEE International Conference on Computer Vision (ICCV), 2019, pp. 4180–4189.

[23] T. Dai, J. Cai, Y. Zhang, S. Xia, L. Zhang, Second-order attention network for single image super-resolution, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 11065–11074.

[24] Y. Mei, Y. Fan, Y. Zhou, L. Huang, T.S. Huang, H. Shi, Image super-resolution with cross-scale non-local attention and exhaustive self-exemplars mining, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 5690–5699.

[25] Y. Mei, Y. Fan, Y. Zhou, Image super-resolution with non-local sparse attention, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 3517–3526.

[26] C.H. Pham, A. Ducournau, R. Fablet, F. Rousseau, Brain MRI super-resolution using deep 3D convolutional networks, in: 14th IEEE International Symposium on Biomedical Imaging (ISBI), 2017, pp. 197–200.

[27] Y. Chen, Y. Xie, Z. Zhou, et al., Brain MRI super resolution using 3D deep densely connected neural networks, in: 15th IEEE International Symposium on Biomedical Imaging (ISBI), 2018, pp. 739–742.

[28] C. Zhao, A. Carass, B.E. Dewey, J.L. Prince, Self super-resolution for magnetic resonance images using deep networks, in: 15th IEEE International Symposium on Biomedical Imaging (ISBI), 2018, pp. 365–368.

[29] L. Zhao, M. Li, D. Meng, et al., Deep convolutional neural networks with merge-and-run mappings, in: International Joint Conference on Artificial Intelligence (IJCAI), 2018, pp. 3170–3176.

[30] G. Huang, Z. Liu, L. Van Der Maaten, et al., Densely connected convolutional networks, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4700–4708.

[31] R.Z. Shilling, T.Q. Robbie, et al., A super-resolution framework for 3D high-resolution and high-contrast imaging using 2D multislice MRI, IEEE Trans. Med. Imaging 28 (5) (2009) 633–644.

[32] J. Hu, X. Wu, J. Zhou, Single image super resolution of 3D MRI using local regression and intermodality priors, in: International Conference on Digital Image Processing (ICDIP), 10033, 2016, 100334C.

[33] S. Roohi, J. Zamani, M. Noorhosseini, M. Rahmati, Super-resolution MRI images using compressive sensing, in: Iranian Conference on Electrical Engineering (ICEE), 2012, pp. 1618–1622.

[34] J. Shi, Z. Li, S. Ying, et al., MR image super-resolution via wide residual networks with fixed skip connection, IEEE J. Biomed. Health Inf. 23 (3) (2018) 1129–1140.

[35] H. Wang, X. Hu, X. Zhao, Y. Zhang, Wide weighted attention multi-scale network for accurate MR image super-resolution, IEEE Trans. Circuits Syst. Video Technol. (2021).

[36] Y. Zhang, K. Li, K. Li, Y. Fu, MR image super-resolution with squeeze and excitation reasoning attention network, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 13425–13434.

[37] J. Hu, L. Shen, S. Albanie, G. Sun, E. Wu, Squeeze-and-excitation networks, 2017, arXiv preprint arxiv:1709.01507.

[38] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.

[39] C.M. Hyun, H.P. Kim, S.M. Lee, S. Lee, J.K. Seo, Deep learning for undersampled MRI reconstruction, Phys. Med. Biol. 63 (13) (2018) 135007.

[40] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Trans. Image Process. 13 (4) (2004) 600–612.

[41] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, J. Mach. Learn. Res. 9 (2010) 249–256.

[42] D.P. Kingma, J. Ba, ADAM: A method for stochastic optimization, 2014, arXiv preprint arXiv:1412.6980.

[43] J.V. Manjón, P. Coupé, A. Buades, et al., Non-local MRI upsampling, Med. Image Anal. 14 (6) (2010) 784–792.

[44] Z. Hui, X. Wang, Fast and accurate single image super-resolution via information distillation network, 2018, arXiv preprint arxiv:1803.09454.

[45] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: ICLR, 2015.

[46] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.

[47] X. Chu, B. Zhang, H. Ma, R. Xu, Q. Li, Fast, accurate and lightweight super-resolution with neural architecture search, in: The 25th International Conference on Pattern Recognition (ICPR), 2021, pp. 59–64.

[48] P. Ren, Y. Xiao, X. Chang, P.-y. Huang, Z. Li, X. Chen, X. Wang, A comprehensive survey of neural architecture search: Challenges and solutions, ACM Comput. Surv. 54 (4) (2021) 76:1–76:34.

[49] N. Ahn, B. Kang, et al., Fast, accurate, and lightweight super-resolution with cascading residual network, in: Europeon Conference on Computer Vision (ECCV), 2018, pp. 256–272.