D2C-SR: A Divergence to Convergence Approach for Real-World Image Super-Resolution

Youwei Li [1] Haibin Huang [2] Lanpeng Jia [3] Haoqiang Fan [1]
Shuaicheng Liu [4,1,†]

[1] Megvii Technology, Beijing, China [2] Kuaishou Technology, Beijing, China
[3] Great Wall Motor Company Limited, Baoding, China
[4] University of Electronic Science and Technology of China, Chengdu, China
{liyouwei, fhq}@megvii.com jialanpeng@gwm.cn jackiehuanghaibin@gmail.com
liushuaicheng@uestc.edu.cn
[†]Corresponding Author

Abstract. In this paper, we present D2C-SR, a novel framework for the task of real-world image super-resolution. As an ill-posed problem, the key challenge in super-resolution related tasks is there can be multiple predictions for a given low-resolution input. Most classical deep learning based approaches ignored the fundamental fact and lack explicit modeling of the underlying high-frequency distribution which leads to blurred results. Recently, some methods of GAN-based or learning super-resolution space can generate simulated textures but do not promise the accuracy of the textures which have low quantitative performance. Rethinking both, we learn the distribution of underlying high-frequency details in a discrete form and propose a two-stage pipeline: divergence stage to convergence stage. At divergence stage, we propose a tree-based structure deep network as our divergence backbone. Divergence loss is proposed to encourage the generated results from the tree-based network to diverge into possible high-frequency representations, which is our way of discretely modeling the underlying high-frequency distribution. At convergence stage, we assign spatial weights to fuse these divergent predictions to obtain the final output with more accurate details. Our approach provides a convenient end-to-end manner to inference. We conduct evaluations on several real-world benchmarks, including a new proposed D2CRealSR dataset with x8 scaling factor. Our experiments demonstrate that D2C-SR achieves better accuracy and visual improvements against state-of-the-art methods, with a significantly less parameters number and our D2C structure can also be applied as a generalized structure to some other methods to obtain improvement. Our codes and dataset are available at https://github.com/megvii-research/D2C-SR.

1 Introduction

Super-resolution (SR) is one of the fundamental problems in computer vision with its applications for several important image processing tasks. Despite decades of development in SR technologies, it remains challenging to recover high-quality and accurate details from low-resolution (LR) inputs due to its ill-posed nature.

In fact, given a LR image, there exist infinitely possible high-resolution (HR) predictions by adding different high-frequency information. This is the key challenge for the design of learning-based SR methods. Recently, deep neural network (DNN) based methods [13, 15, 19, 20, 32] have achieved tremendous success and outperformed most classical methods based on sparse coding [6, 30] or local liner regression [23, 29]. However, these early deep learning methods only rely on L_1 or L_2 reconstruction loss, which have a relatively high quantitative performance but can not reproduce rich texture details due to ill-posed nature. Later on, conditional GAN-based methods [7] are adapted into SR tasks, which provide methods for learning the distribution in a data-driven way and led to generate richer simulated textures. However, these methods still suffered from mode collapse and tend to generate implausible or inaccurate texture details. Thus, the quantitative performance of GAN-based methods are not satisfactory. After realizing the importance of modeling the underlying high-frequency distribution explicitly, some methods of learning SR space appear. SRFlow [14] introduced a normalizing flow-based method which tried to address the mode collapse problem and make a continuous Gaussian distribution to model all the high-frequency. Therefore, although SRFlow can sample to get a certain prediction, it does not promise the most accurate prediction. Furthermore, when adapting more complicated distribution, like large amount of real-world scenes, it would significantly increase the hardness of convergence during training. Some works [11, 16, 24, 28] introduced ensemble-based SR approaches that trained a same network from different initialization or different down-sampling methods and merged the outputs into the SR results. Due to the lack of explicit modeling of the SR space, the different results produced by these methods are very dependent on the randomness of the initialization or artificially rules, and are therefore unstable and has a risk of degradation. In other words, they lack an explicit and stable way to make the results divergent. However, these works still give us meaningful inspiration. In this paper, some discussions are made based on this.

In this paper, we present D2C-SR, a novel divergence to convergence framework for real-world SR, by rethinking the ill-posed problem and the SR space approaches. D2C-SR follows the idea of explicitly learning the underlying distribution of high-frequency details. But unlike conditional GAN-based methods or SRFlow [14], our D2C-SR model the high-frequency details distribution using a discrete manner. Our key insight is that: most classical SR methods use a single output to fit all high-frequency details directly and 'regression to mean', therefore only obtains average and blurred outputs. While SRFlow [14] uses an continuous Gaussian distribution to fit all high-frequency details, but the fit is very difficult and does not guarantee stable and accurate high-frequency details. The GAN-based approaches have the same problem. Therefore, we adopt a trade-off approach by using a finite discrete distribution to fit the high-frequency details, thus ensuring a stable, controllable and accurate output relatively.

Specifically, at divergence stage, we first propose a tree-based structure network, where end branches are designed to learn random possible high-frequency details separately. To stabilize the divergence process, we also propose the diver-

gence loss which can explicitly allows for divergence of the results produced by different branches. Meanwhile, we use the construction loss to ensure consistency with HR during the entire process. Due to there are multiple local minima in the network optimization process, we expect different branches to have access to different local minima through the appropriate strength of these restraints. Theoretically, if there are multiple branches and fit to the full range of high-frequency possibilities, the best HR image should be the fusion of the high-frequency details exactly. Hence, at convergence stage, we assign spatial weights to combine the divergent predictions to produce a more accurate result.

To fully evaluate the efficiency and generality of D2C-SR for real-world SISR, we conduct experiments on several benchmarks, including RealSR [4], DRealSR [27] and our new benchmark, D2CRealSR, with x8 upscaling factor. Experimental results show that our D2C-SR can achieve state-of-the-art performance and visual improvements with less parameters. In sum, the main contributions are as follows:

- We present D2C-SR, a novel framework with divergence and convergence stage for real-world SR. D2C-SR explicitly model the underlying distribution of high-frequency details in a discrete manner and provide a convenient end-to-end inference. For stable divergence, we also propose the divergence loss to generated multiple results with divergent high-frequency representations.
- A new real-world SR benchmark (D2CRealSR), which has a larger scaling factor (x8) compared to existing real-world SR benchmarks.
- D2C-SR sets up new state-of-the-art performance on many popular real-world SR benchmarks, including our new proposed D2CRealSR benchmark and, it can provide compatible performance with significantly less parameter number.
- Our D2C structure can also be applied as a generalized structure to some other methods to obtain improvement.

2 Related Work

2.1 DNN-based SISR

Single Image Super-Resolution (SISR) is a long standing research topic due to its importance and ill-posed nature. Traditional learning-based methods adopts sparse coding [6, 21, 30] or local linear regression [22, 23, 29]. Deep learning (DL)-based methods have achieved dramatic advantages against conventional methods for SISR [1, 26]. It is first proposed by SRCNN [19] that employs a relatively shallow network and adopts the bicubic degradation for HR and LR pairs. Following which, various SISR approaches have been proposed, such as VDSR that adopts very deep network [19]; EDSR that modifies the ResNet for enhancement [13]; ESPCN that uses efficient sub-pixel CNN [18]; CDC that divides images into multiple regions [27], and VGG loss [19], GAN loss [7] that improve the perceptual visual quality [12, 17, 25]. Ensemble-based methods [11, 16, 24, 28] train a same network from different initialization or different training pair generation but it lacks explicit modeling of SR space. In this work, we propose a tree-based network structure, for the purpose of multi-mode learning.

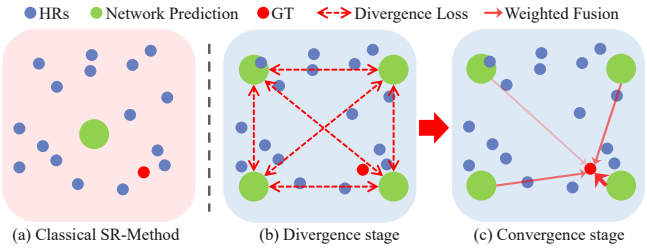


Fig. 1: Difference between our method and the classical methods in handling ill-posed problems. Blue and red dots in the box indicate possible HR texture details and the ideal details for a settled LR region. Classical methods only find the single average prediction. Our method can produce multiple divergent predictions at divergence stage and fuse to obtain more accurate predictions at convergence stage.

2.2 Learning Super-Resolution Space

Single image super-resolution is ill-posed, where infinitely many high-resolution images can be downsampled to the same low-resolution images. Therefore learning a deterministic mapping may not be optimal. The problem can be converted to stochastic mapping, where multiple plausible high-resolution images are predicted given a single low-resolution input [2]. DeepSEE incorporates semantic maps for explorative facial super-resolution [3]. SRFlow adopts normalizing flow to learn the high-frequency distribution of facial datasets by using an continuous Gaussian distribution [14]. Unlike this, our method model the high-frequency distribution using a discrete manner. Thus the prediction of our method has better texture consistency with HR and the details are more accurate.

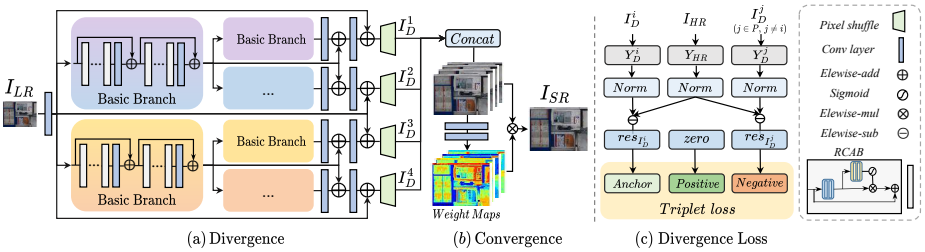
2.3 Real-World Super-Resolution and Datasets

Bicubic downsampling datasets are widely used but the gap between the simulation and the real degradation limits the performance in the real-world application [8]. Therefore, some works explore degradation of real scenarios by collecting real-world datasets, e.g., City100 [5], RealSR [4], SR-RAW [31] and DRealSR [27]. We propose the D2CRealSR dataset with larger scaling factor (x8) compared to the above datasets. Real-world degradation loses more details compared to simulated degradation [5], and the reduction of the prior makes recovery more difficult. Our method can handle these problems better than others.

3 Method

3.1 Overview

As shown in Fig. 2, D2C-SR consists of two sub-networks: a divergence network to learn the possible predictions and a convergence network to fuse the predictions to



(a) Divergence (b) Convergence (c) Divergence Loss

Fig. 2: Two stages in D2C architecture: (a) Divergence stage, (b) Convergence stage. Divergence network with tree-based structure outputs multi-predictions I_D^i with different high-frequency recovery. Convergence network obtains more accurate result by weighted combining divergence results. (c) Divergence loss.

final result. In this section, we first describe divergence network in Sec. 3.2. Then, we introduce convergence network in Sec. 3.3. The network training strategy are described in Sec. 3.4. To facilitate the explanation, we also present in Fig. 1.

3.2 Divergence Network

Divergence network produce multiple divergent predictions to address the ill-posed nature. We built a tree-based structure network to get the desired predictions. There are four main modules in the divergence network: shallow feature extraction, basic branch module, multi-branch deep residual structure and upscale module. We define the depth of the tree network is L, and each branch has C sub-branches. Each branch and its sub-branches are composed of a basic branch module and non-weight shared. Further, the basic branch module contains G residual groups, and each residual group contains B residual channel attention blocks, which refer to RCAN [32]. The divergence network first extracts the shallow features from the input LR image by the shallow feature extraction module, which is a simple Conv layer. Then, the input features are each fed into their own branch and sub-branches modules. The feature outputs of the end branches go through the multi-branch deep residual structure, then upscaled by the upscale module. So far, the divergence network generate P divergent predictions. These predictions are expressed as $I_D = \mathcal{F}(I_{LR}; \Theta_D)$, where Θ_D is the divergence network parameter and I_{LR} is the LR image. I_D^i denotes i-th prediction of divergence network.

Multi-branch Deep Residual Structure. We construct relatively deeper residuals and apply residuals structure to all branches of our tree-based divergence network. In addition to the residuals within the residual group and the residuals from the input features to the end branches, we also add the residuals from intermediate branches to their own sub-branches as shown in Fig. 2. Residual learning address the problem of long-term memory in deep SR network. Therefore, inspired by previous works [10, 12, 32], we utilize multi-branch deep residual

structure to achieve more performance gain. We validate the effect of deep residual structure in Sec. 4.5.

Divergence loss. Divergence loss used in divergence network, which composed of L_2 loss and modified triplet loss. The L_2 loss can be defined as

$$L_2^D = \sum_{i=1}^{P} \left\| I_D^i - I_{HR} \right\|_2 . \tag{1}$$

It is worth noting that we used L_2 loss on all branches separately instead of just averaging them out with an overall L_2 loss. Because it allows for a more independent divergence of each branch, which is beneficial for the divergence stage. In order to make the divergence network produce divergence results more stable, we adopt triplet loss between all pairs of different predictions from divergence network. Our goal is to make the distance between I_D^i and HR close and the distance between different members within I_D farther. However, using triplet loss directly on RGB images causes the network to focus more on other differentiated directions (e.g., RGB space and luminance) than texture features. Therefore, we perform a series of processes on the triplet inputs shown in Fig. 2. Firstly, we process I_D^i by $G(\cdot)$:

$$G(I_D^i) = \frac{Y_D^i - \mu_{Y_D^i}}{\sigma_{Y_D^i}}, \tag{2}$$

where Y_D^i is Y channel of I_D^i in YCbCr space, $\mu_{Y_D^i}$ and $\sigma_{Y_D^i}$ are mean and standard deviation of Y_D^i respectively. $G(\cdot)$ operation enables network to focus more on the differentiation of texture space rather than color and luminance space. Secondly, the triplet inputs are converted to the residual domain where texture differences can be better represented. We express the residual of I_D^i as

$$res_{I_D^i} = \left| G(I_D^i) - G(I_{HR}) \right|, \tag{3}$$

where $|\cdot|$ is absolute value function. The pixel value of the residual images can oscillate between positive and negative, and absolute value function can reduce the checkerboard phenomenon caused by triplet loss as shown in Fig 6. Overall, the formula for triplet loss is

$$trip(a, p, n) = Max[d(a, p) - d(a, n) + margin, 0], \tag{4}$$

where a, p and n are anchor, positive and negative, respectively. Therefore, we can represent our final triplet loss as

$$T_D = \frac{\sum_{i=1}^{P} \sum_{j=1, j\neq i}^{P} \beta_{ij} * trip(res_{I_D^i}, zero, res_{I_D^j})}{(P(P-1))}, \tag{5}$$

where $zero$ is zero map and it is the positive in residual domain. β is a attenuation coefficient which can be described as

$$\beta_{ij} = \theta^{l-1}, l \in [1, L], \tag{6}$$

where θ is parameter and $\theta \in (0, 1]$. The l is index of the tree depths where the common parent branch of I_D^i and I_D^j is located. We use β because that in a tree structure, differentiation should be progressive, i.e., branches with closer relatives should have relatively smaller degrees of divergence. Finally, our divergence loss is described as

$$L_D = L_2^D + \alpha * T_D, \tag{7}$$

where α is weight hyperparameter of T_D. Some detailed parameter settings will be introduced in Sec. 4.1.

3.3 Convergence Network

Combining the divergence results generated by the divergence network can produce more accurate results. We think that different areas on the predictions have different contribution weights for the final result. So we construct convergence network to merge divergent predictions weighted pixel-by-pixel, which can generate the result closer to the real HR image. Convergence network concatenates all the P predictions of the divergence network and outputs weight map for each prediction I_D^i. Weight maps can be expressed as $W = F(Concat(I_D); \Theta_C)$, where Θ_C is the convergence network model parameter. We denote W_i as the weight map of I_D^i. Every I_D^i are element-wise multiplied by respective weight map W_i, and then all of the results are summed to produce the final SR result. Accordingly, SR result can be defined as

$$I_{SR} = \sum_{i=1}^{P} \left(I_D^i \cdot W_i \right), \tag{8}$$

where I_{SR} represents final SR result.

Convergence loss. The goal of convergence network is to merge I_D^i from divergence network. Therefore, loss function of convergence network is called convergence loss, which only consists of L_2 loss. We denote convergence loss as L_2^C, which can be expressed as

$$L_2^C = \left\| I_{SR} - I_{HR} \right\|_2. \tag{9}$$

How to generate I_{SR} has been introduced in Eq. 8. The goal of convergence loss is to make the generated SR image get closer to the ground-truth HR image.

3.4 Training Strategy

The two networks in our framework are trained separately. We firstly train the divergence network into a stable status where it can generate super-resolution divergence predictions. We then freeze the parameters of divergence network and train the convergence network by enabling the whole pipeline. More details will be discussed in Sec. 4.1.

Table 1: Performance comparison on RealSR [4], DRealSR [27] and our proposed D2CRealSR datasets. The best results are **highlighted**. '-' indicates either the available model is not supported for such a test, or not open-sourced.

Method	RealSR						DRealSR (train on RealSR)						D2CRealSR	
	x2		x3		x4		x2		x3		x4		x8	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Bicubic	31.67	0.887	28.61	0.810	27.24	0.764	32.67	0.877	31.50	0.835	30.56	0.820	27.74	0.822
DRCN [11]	33.42	0.912	30.36	0.848	28.56	0.798	32.46	0.873	31.58	0.838	30.14	0.816	29.99	0.833
SRResNet [12]	33.17	0.918	30.65	0.862	28.99	0.825	32.85	0.890	31.25	0.841	29.98	0.822	30.01	0.864
EDSR [13]	33.88	0.920	30.86	0.867	29.09	0.827	32.86	0.891	31.20	0.843	30.21	0.817	30.23	0.868
RCAN [32]	33.83	0.923	30.90	0.864	29.21	0.824	32.93	0.889	31.76	0.847	30.37	0.825	30.26	0.868
ESRGAN [25]	33.80	0.922	30.72	0.866	29.15	0.826	32.70	0.889	31.25	0.842	30.18	0.821	30.06	0.865
SR-Flow [14]	-	-	-	-	24.20	0.710	-	-	-	-	24.97	0.730	23.11	0.600
LP-KPN [4]	33.49	0.917	30.60	0.865	29.05	**0.834**	32.77	-	31.79	-	30.75	-	-	-
CDC [27]	33.96	0.925	30.99	0.869	29.24	0.827	32.80	0.888	31.65	0.847	30.41	**0.827**	30.02	0.841
D2C-SR(Ours)	**34.40**	**0.926**	**31.33**	**0.871**	**29.72**	0.831	**33.42**	**0.892**	**31.80**	**0.847**	**30.80**	0.825	**30.55**	**0.871**

4 Experiment

4.1 Dataset and Implementation Details

D2CRealSR. Existing RealSR datasets generally include x2, x3 and x4 scaling factor only, but lack of larger scaling factor. We collect a new dataset on x8 scaling factor, which called D2CRealSR. We construct the LR and HR pairs by zooming the lens of DSLR cameras. D2CRealSR consists of 115 image pairs and 15 image pairs are selected randomly for testing set; the rest pairs construct the training set. We use SIFT method to register the image pairs iterative: we first register the image pairs roughly and crop the main central area, then we align the brightness of the central area of image pairs and register them once again. After, we crop off the edges of aligned image pairs. The image size of each pair is 3,456×2,304 after the alignment process.

Existing datasets. We also conduct experiments on existing real-world SR datasets: RealSR and DRealSR. RealSR has 559 scenes captured from DSLR cameras and align the image pairs strictly. 459 scenes for training and 100 scenes for testing. The image sizes of RealSR image pairs are in the range of 700˜3000 and 600˜3500. DRealSR has 83, 84 and 93 image pairs in testing set, 884, 783 and 840 image pairs in training set for x2, x3, x4 scaling factors, respectively. Note that we found some of the image pairs in DRealSR are misaligned, which is caused by the depth of field. So we only validate our results on testing set of DRealSR to show the performance of the cross-dataset of our method.

Implementation details. In the experiment, we set the number of branch layers as $L = 2$, the number of child branches $C = 2$. The basic branch module include $G = 2$ residual groups and $B = 4$ residual blocks in each group. We use Adam optimizer and set exponential decay rates as 0.9 and 0.999. The initial learning

rate is set to 10^{-4} and then reduced to half every $2k$ epochs. For each training batch, we randomly extract 4 LR image patches with the size of 96×96. We implement D2C-SR method with the Pytorch framework and train the network using NVIDIA 2080Ti GPU.

4.2 Comparisons with Existing Methods

To evaluate our method, we train and test our model on our D2CRealSR with scaling factor x8 and an existing real-world SR dataset, RealSR with scaling factor x4, x3, x2. In addition, we validate on DRealSR tesing set for performance of cross-dataset. We compare our model with other state-of-the-art SISR methods, including DRCN [11], SRResNet [12], EDSR [13], RCAN [32], ESRGAN [25], LP-KPN [4] and CDC [27]. The SISR results were evaluated on the Y channel in the YCbCr space using PSNR and SSIM. Among these SR methods, EDSR and RCAN are the classic SISR methods and DRCN is the open-source ensemble-based method. In addition, LP-KPN and CDC are designed to solve the real-world SR problem and they outperform on real-world SR benchmarks.

Quantitative comparison. The evaluation results of the SR methods, including our model and the other 8 methods are demonstrated in Table 1. The best results are highlighted and our method are $0.48dB$ higher than the second ranked CDC method on x4 RealSR. Our model outperforms by a large margin on all the benchmarks and achieve state-of-the-art performance.

Qualitative comparison. Visualization results of SR methods and ours on RealSR and D2CRealSR datasets are shown in Fig. 3 and Fig. 4. It is observed that existing SR methods (e.g., RCAN, LP-KPN, CDC) tend to restore the details to be thicker, blurry, or unnatural. Meanwhile, in Canon-041, Nikon-050 and Sony-101, our method recovered sharper and more accurate edge textures. In Nikon-024, we recover more information on the letters. From the visual comparisons, our method has the ability to recover richer and more accurate details.

Visualization of D2C processes. To further represent the progress of the D2C-SR, we visualize and compare the LR image, HR image, divergence intermediate results, and final results as well as the results of other comparison methods in Fig. 5. Classical SR methods directly use a single prediction to fit the distribution of all high-frequency details, thus obtain blurred outputs due to the ill-posed nature. Our two-stage approach explicitly learns the distribution of high-frequency details using a discrete manner, thus we can get rich and accurate texture details.

Comparison in perceptual metric. In recent years, the perceptual metric becomes another dimension to evaluate the image quality, and we also conduct a comparison on the perceptual metric in Table 2. Based on Sec. 3, we add vgg loss to train our model [12] to obtain a balance of multiple metrics. Experiments

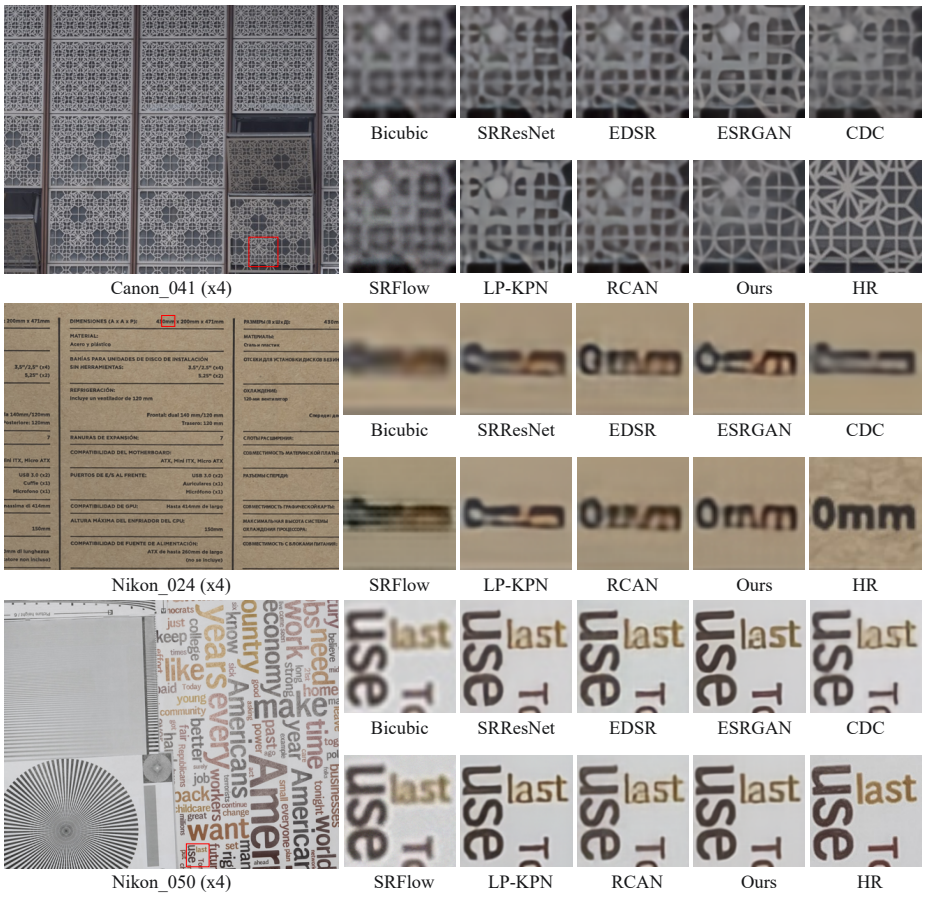


Fig. 3: Comparison for x4 SR on RealSR [4] dataset.

demonstrate that our method can achieve better performance on several evaluation measures, including the perceptual metric.

4.3 Applying the D2C Structure to Other Methods

In Table 3, we apply the D2C structure to other methods and compare the performance with the original method. To make a fair comparison and reduce the influences caused by different sizes of parameters, we reduce the number of feature channels or the basic modules such that the number of parameters of the D2C counterpart model is smaller than the original model. Our results show that multiple methods can achieve better performance with fewer number of parameters by applying the D2C structure.

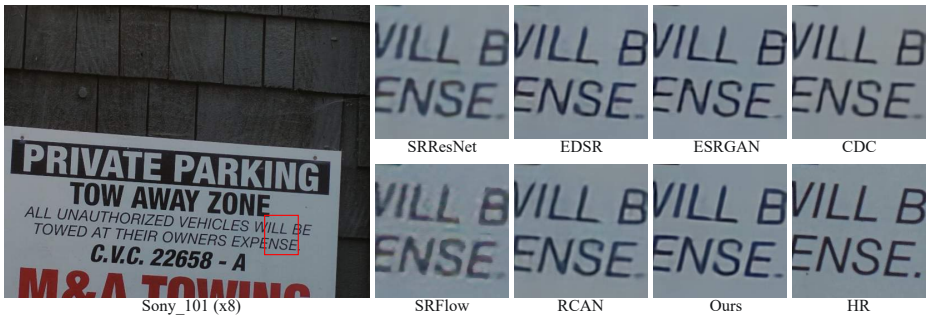


Fig. 4: Comparison for x8 SR on our captured D2CRealSR dataset.

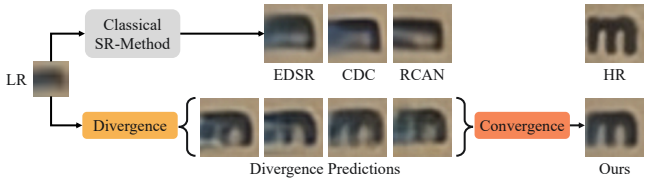


Fig. 5: Visualization of D2C processes and other methods (x4).

4.4 Model Size Analyses

We show comparisons about model size and performance in Fig. 7. We list six sizes of models: 5.88M, 4.53M, 3.63M, 0.91M, 0.23M and 0.19M. These models are constructed by changing the number of residual groups G and the number of residual block B. Our 0.23M model can achieve a better effect than other methods. At this PSNR level, CDC uses 39.92M, and RCAN also uses 15M parameters. Our baseline model chieve higher performance using 5.88M parameters only. Our method have a better trade-off between model size and performance.

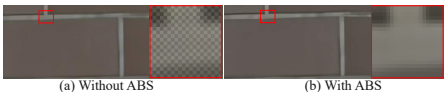


Fig. 6: Absolute value function (ABS) reduces the checkerboard phenomenon.

Table 4: Effect study on the width and depth of the tree-based network (x4).

	1	2	3	4
Width (C)	29.41	29.54	29.56	**29.58**
Depth (L)	29.30	29.54	29.63	**29.64**

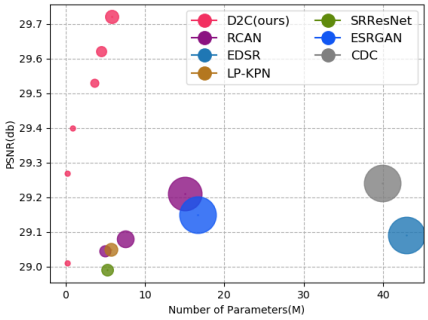


Fig. 7: Performance vs. parameters.

Table 2: Performance comparison in LPIPS on RealSR.

Method	x2			x3			x4		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Bicubic	31.67	0.887	0.223	28.61	0.810	0.389	27.24	0.764	0.476
SRResNet [12]	33.17	0.918	0.158	30.65	0.862	0.228	28.99	0.825	0.281
EDSR [13]	33.88	0.920	0.145	30.86	0.867	0.219	29.09	0.827	0.278
RCAN [32]	33.83	0.923	0.147	30.90	0.864	0.225	29.21	0.824	0.287
CDC [27]	33.96	0.925	0.142	30.99	0.869	0.215	29.24	0.827	0.278
D2C-SR(Ours)	**34.39**	**0.926**	**0.136**	**31.31**	**0.870**	**0.214**	**29.67**	**0.830**	**0.270**

Table 3: Performance of applying our D2C structure to other methods on RealSR.

Method	x2		x3		x4		Parameters
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	
SRResNet [12]	33.17	0.918	30.65	0.862	28.99	**0.825**	1.52M
D2C-SRResNet	**33.82**	**0.920**	**30.69**	**0.862**	**29.19**	0.822	**1.37M**
VDSR [10]	31.39	0.876	30.03	0.845	27.07	0.751	0.67M
D2C-VDSR	**34.08**	**0.920**	**30.29**	**0.858**	**28.21**	**0.793**	**0.67M**
RRDBNet [25]	33.44	0.919	30.29	0.858	28.34	0.813	16.7M
D2C-RRDBNet	**33.49**	**0.920**	**30.33**	**0.859**	**28.47**	**0.814**	**9.16M**
RDN [33]	33.97	0.922	30.90	0.864	29.23	0.824	6.02M
D2C-RDN	**34.03**	**0.922**	**30.93**	**0.865**	**29.32**	**0.825**	**5.59M**
EDSR [13]	33.88	0.920	30.86	0.867	29.09	0.827	43.1M
D2C-EDSR	**34.17**	**0.924**	**31.08**	**0.868**	**29.41**	**0.829**	**7.91M**
IMDN [9]	33.59	0.916	30.74	0.859	29.17	0.819	0.874M
D2C-IMDN	**33.95**	**0.920**	**30.98**	**0.862**	**29.45**	**0.824**	**0.738M**

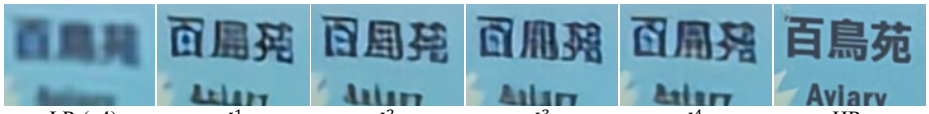
4.5 Ablation Studies

The width and depth of tree-based network. Our divergence network is a tree-based architecture. We set different L and C to provide experimental evaluations on the width and depth of the tree. Because increasing the depth and width increases memory usage. So we do these experiments on a small baseline model, which has $G = 2$ residual groups and each group has $B = 1$ block. As show in Table 4, we increase the width and depth from 1 to 4 respectively on x4 scaling factor and show the changes in PSNR. By increasing the width or depth of the tree-based network, the performance of our D2C-SR improves.

Divergence loss. As mentioned in Sec. 3.2, the divergence loss is used to enforce the outputs from different branches divergent. In Fig. 8, the visualization divergent results show that different branches produce different texture predictions. We verify the effectiveness of our divergence loss by removing the term of Eq. 5. As shown in Table 5 "w/o. divergence loss", the approach appears degraded. Because of the random nature of the model initialization, it is possible to lead to some degree of divergence in the results without using triplet loss and thus still

Table 5: Effect of our divergence loss and multi-branch deep residual structure.

	x2		x3		x4	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
w/o. Divergence Loss	33.98	0.920	31.12	0.863	29.43	0.824
w/o. Multi Deep Res	34.28	0.925	31.28	0.870	29.68	0.830
D2C-SR(Ours)	**34.40**	**0.926**	**31.33**	**0.871**	**29.72**	**0.831**



LR (x4) I_D^1 I_D^2 I_D^3 I_D^4 HR

Fig. 8: Visualization of the divergent predictions.

gain some performance, it is still possible to gain some benefit from the effect of divergence. However, as mentioned in Sec. 1, it is not stable without the explicit divergence constraint, and thus may still bring degradation of performance. Further, we change the weight of the divergence loss and observe the changes that occur in different branches. As shown in Fig. 9, the results are relatively similar when divergence loss is not used. As the coefficients α increase, the differentiation of the different branches becomes more obvious. In this example, the results containing richer and more details appear. Theoretically, if α or the margin is too large it may lead to performance degradation, so we determine the α or the margin based on the model convergence.

Multi-branch deep residual structure. As mentioned in Sec. 3.2, we also disable the structure to verify its effectiveness. As shown in Table 5 "w/o. Multi Deep Res", the PSNR decreased without the multi-branch deep residual structure.

4.6 Visualization of Weight Maps

We assign spatial weights to the divergence results and use this weight for the fusion of the divergence results at the convergence stage. In Fig. 10, we visualize the weight maps using the heatmap. The visualization results show that predictions have some regional and texture correlation. We suggest that the weight generation depends mainly on the sharpness, edge style and texture preferences of the divergence results. We further measure the gap between the learned weight and the approximate ideal weight by finding the pixel that best matches the HR in the divergence results. The PSNR between the ideal fusion results and HR is 31.06dB on RealSR x4, thus the gap is 1.34dB (29.72dB ours).

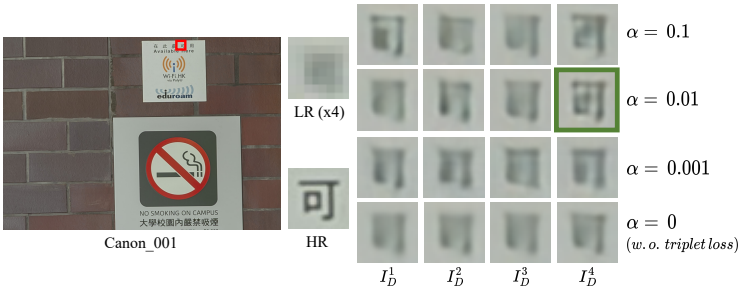


Fig. 9: Comparisons on different weighting coefficient α in our divergence loss.

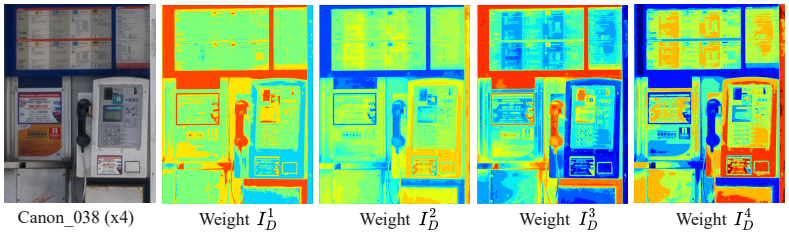


Fig. 10: Weight maps in convergence stage. Red indicates higher values.

4.7 Simulated SISR and Real-World SISR

More works have focused on real-world SR because the great gap between simulated and real-world degradation hinders practical SR applications [4, 27]. On the other hand, real-world degradation loses more information than bicubic compared with the HR, which also have been discussed in [5], and ill-posed problem becomes more apparent in real-world datasets. Therefore, the real-world datasets can better reflect the effectiveness of our method.

5 Conclusion

In this study, we revisit the problem of image super-resolution and provide a new two-stage approach: divergence stage for multiple predictions learning as well as convergence for predictions fusion. Considering the limitations of traditional SR methods and SR space methods, we adopt a trade-off approach by using a finite discrete distribution to fit the high-frequency details. This allows the network to be more efficient and achieves state-of-the-art performance with much less computational cost. Futher, our D2C framework is a promising direction for image processing tasks like image inpainting as well as image denoising, and it is worth further exploration.

Acknowledgment This work was supported by the National Natural Science Foundation of China (NSFC) under grants No. 61872067 and No. 61720106004.

References

1. Anwar, S., Khan, S., Barnes, N.: A deep journey into super-resolution: A survey. ACM Computing Surveys (CSUR) **53**(3), 1–34 (2020) 3
2. Bahat, Y., Michaeli, T.: Explorable super resolution. In: CVPR. pp. 2716–2725 (2020) 4
3. Buhler, M.C., Romero, A., Timofte, R.: Deepsee: deep disentangled semantic explorative extreme super-resolution. In: ACCV (2020) 4
4. Cai, J., Zeng, H., Yong, H., Cao, Z., Zhang, L.: Toward real-world single image super-resolution: A new benchmark and a new model. In: CVPR. pp. 3086–3095 (2019) 3, 4, 8, 9, 10, 14
5. Chen, C., Xiong, Z., Tian, X., Zha, Z.J., Wu, F.: Camera lens super-resolution. In: CVPR. pp. 1652–1660 (2019) 4, 14
6. Dai, D., Timofte, R., Van Gool, L.: Jointly optimized regressors for image super-resolution. In: Computer Graphics Forum. vol. 34, pp. 95–104 (2015) 2, 3
7. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. arXiv preprint arXiv:1406.2661 (2014) 2, 3
8. Gu, J., Lu, H., Zuo, W., Dong, C.: Blind super-resolution with iterative kernel correction. In: CVPR. pp. 1604–1613 (2019) 4
9. Hui, Z., Gao, X., Yang, Y., Wang, X.: Lightweight image super-resolution with information multi-distillation network. In: Proceedings of the 27th acm international conference on multimedia. pp. 2024–2032 (2019) 12
10. Kim, J., Kwon Lee, J., Mu Lee, K.: Accurate image super-resolution using very deep convolutional networks. In: CVPR. pp. 1646–1654 (2016) 5, 12
11. Kim, J., Lee, J.K., Lee, K.M.: Deeply-recursive convolutional network for image super-resolution. In: CVPR. pp. 1637–1645 (2016) 2, 3, 8, 9
12. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In: CVPR. pp. 4681–4690 (2017) 3, 5, 8, 9, 12
13. Lim, B., Son, S., Kim, H., Nah, S., Mu Lee, K.: Enhanced deep residual networks for single image super-resolution. In: CVPRW. pp. 136–144 (2017) 2, 3, 8, 9, 12
14. Lugmayr, A., Danelljan, M., Van Gool, L., Timofte, R.: Srflow: Learning the super-resolution space with normalizing flow. In: ECCV. pp. 715–732 (2020) 2, 4, 8
15. Michelini, P.N., Liu, H., Zhu, D.: Multigrid backprojection super–resolution and deep filter visualization. In: AAAI. vol. 33, pp. 4642–4650 (2019) 2
16. Pan, Z., Li, B., Xi, T., Fan, Y., Zhang, G., Liu, J., Han, J., Ding, E.: Real image super resolution via heterogeneous model ensemble using gp-nas. In: ECCV. pp. 423–436. Springer (2020) 2, 3
17. Sajjadi, M.S., Scholkopf, B., Hirsch, M.: Enhancenet: Single image super-resolution through automated texture synthesis. In: ICCV. pp. 4491–4500 (2017) 3
18. Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: CVPR. pp. 1874–1883 (2016) 3
19. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014) 2, 3
20. Song, D., Xu, C., Jia, X., Chen, Y., Xu, C., Wang, Y.: Efficient residual dense block search for image super-resolution. In: AAAI. vol. 34, pp. 12007–12014 (2020) 2

21. Sun, L., Hays, J.: Super-resolution from internet-scale scene matching. In: Proc. ICCP. pp. 1–12 (2012) 3
22. Timofte, R., De Smet, V., Van Gool, L.: Anchored neighborhood regression for fast example-based super-resolution. In: ICCV. pp. 1920–1927 (2013) 3
23. Timofte, R., De Smet, V., Van Gool, L.: A+: Adjusted anchored neighborhood regression for fast super-resolution. In: ACCV. pp. 111–126 (2014) 2, 3
24. Wang, L., Huang, Z., Gong, Y., Pan, C.: Ensemble based deep networks for image super-resolution. Pattern recognition **68**, 191–198 (2017) 2, 3
25. Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., Qiao, Y., Change Loy, C.: Esrgan: Enhanced super-resolution generative adversarial networks. In: ECCV. pp. 0–0 (2018) 3, 8, 9, 12
26. Wang, Z., Chen, J., Hoi, S.C.: Deep learning for image super-resolution: A survey. IEEE Trans. on Pattern Analysis and Machine Intelligence (2020) 3
27. Wei, P., Xie, Z., Lu, H., Zhan, Z., Ye, Q., Zuo, W., Lin, L.: Component divide-and-conquer for real-world image super-resolution. In: ECCV. pp. 101–117 (2020) 3, 4, 8, 9, 12, 14
28. Xiong, D., Gui, Q., Hou, W., Ding, M.: Gradient boosting for single image super-resolution. Information Sciences **454**, 328–343 (2018) 2, 3
29. Yang, C.Y., Yang, M.H.: Fast direct super-resolution by simple functions. In: ICCV. pp. 561–568 (2013) 2, 3
30. Yang, J., Wright, J., Huang, T., Ma, Y.: Image super-resolution as sparse representation of raw image patches. In: CVPR. pp. 1–8 (2008) 2, 3
31. Zhang, X., Chen, Q., Ng, R., Koltun, V.: Zoom to learn, learn to zoom. In: CVPR. pp. 3762–3770 (2019) 4
32. Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: ECCV. pp. 286–301 (2018) 2, 5, 8, 9, 12
33. Zhang, Y., Tian, Y., Kong, Y., Zhong, B., Fu, Y.: Residual dense network for image super-resolution. In: CVPR. pp. 2472–2481 (2018) 12