SRCN: Joint Sub-bands Learning for Wavelet Domain Super Resolution

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Convolutional neural network (CNN) has recently achieved great success in single-image super-resolution (SISR). However, these methods tend to produce over-smoothed outputs and miss some textural details. To solve above problems, we propose the Super Resolution Clique Net (SRCN) to reconstruct high resolution (HR) images with better textural details in wavelet domain. The proposed SRCN firstly extracts a set of feature maps from the low resolution (LR) images by the clique blocks group. Then we send the set of feature maps to the clique up-sampling module to reconstruct HR images. The clique up-sampling module consists of four sub-nets which use to predict the high resolution wavelet coefficients of four sub-bands. Since we consider the edge feature properties of four sub-bands, four sub-nets have connections with the others so that they can learn the coefficients of four sub-bands jointly. We apply inverse discrete wavelet transform (IDWT) to the output of four sub-nets at the end of clique up-sampling module to increase the resolution and reconstruct HR images. Extensive quantitative and qualitative experiments on benchmark datasets show that our method achieves superior performance over the state-of-the-art methods.

1 Introduction

Single image super-resolution (SISR) is to reconstruct a high-resolution (HR) from a single low-resolution (LR) image, which is an ill-posed inverse problem. SISR has gained increasing research attention for decades. Recently, convolutional neural networks (CNNs) [6, 32, 25] significantly improve the peak signal-to noise ratio (PSNR) in SISR . These networks commonly use an extraction module to extract a series of feature maps from the LR image, cascading with up-sampling modules to increase resolution and construct the HR image.

The quality of extracting features will affect the performance of HD image reconstruction. The main part of extraction module used in modern SR networks can be primarily divided into three types: conventional convolution layers [23], residual blocks [9] and dense blocks [10].

Conventional convolution has been widely concerned by scholars since AlexNet [20] won the first prize of ILSVRC in 2012. The first model based on conventional convolution to solve SR problem is SRCNN [6]. After that, many improved networks such as FSRCNN [7], SCN [35], ESPCN [28] and DRCN [18] also use conventional convolution and achieve great results. Residual block [9] is a improved version of convolutional layer which exhibits excellent performance in computer vision problems. Since it can enhance the feature propagation in networks and alleviate the vanishing-gradient problem, many SR networks such as VDSR [17], LapSRN[22], EDSR [25] and SRResNet [24] import residual blocks and exhibit improved performances. To make use of the skip connections used in residual block further, Huang et al. proposed dense block [10]. Dense block builds more

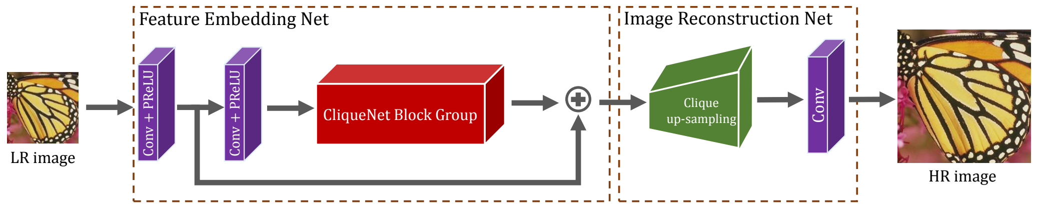

Figure 1: The architecture of the proposed network (SRCN).

connections among layers to enlarge the information flow. Tong et al. [34] proposed SRDenseNet using dense blocks, which boosting the performances further more.

Recently, Yang et al proposed a novel block called clique block [37] where the layers in a block are constructed as a clique and are updated alternately in a loop manner. Any layer is both the input and output of another one in the same block so that the information flow is maximized. The propagation of clique block contains two stages. The first stage does the same thing as dense block. The second stage distills the feature maps by using the skip connections among any layers including connections among subsequent layers.

Suitable up-sampling module can further improve image reconstruction performance. The up-sampling modules used in modern SR networks to increase the resolution can also be primarily divided into three types: interpolation up-sampling, deconvolution up-sampling and sub-pixel convolution up-sampling.

Interpolation up-sampling was first used in SRCNN [6]. At that time, there were no effective implementations of module that can make the output size larger than the input size. So SRCNN used pre-defined bicubic interpolation on input images to get desired size first. Following SRCNN using pre-interpolation, VDSR [17], IRCNN [41], DRRN [31] and MemNet [32] attempted different extraction modules. However, this pre-processing step increases computation complexity due to the size of feature maps is multiple. Deconvolution proposed in [39, 38] can be seen as multiplication of each input pixel by a filter, which could increase the input size if the stride $s > 1$. Many modern SR networks such as FSRCNN [7], LapCNN [22], DBPN [8] and IDN [14] got better results by using deconvolution as the up-sample module. However, the computation complexity of forward and back propagation of deconvolution is still a major concern. Sub-pixel convolution was proposed in [28] aiming at accelerating the up-sampling operation. Unlike previous up-sampling methods that change the height and width of the input feature maps, sub-pixel convolution implements up-sampling by increasing the number of channels. After that sub-pixel convolution uses a periodic shuffling operator to reshape the output feature map to desired height and width. Though ESPCN [28], EDSR [25] and SRMD [41] used sub-pixel convolution to achieve good performances on benchmark datasets, these networks tend to produce blurry and overly-smoothed SR images, lacking some texture details.

Wavelet transform (WT) has been shown to be an efficient and highly intuitive tool to represent and store images in a multi-resolution ways [30, 26]. WT can describe the contextual and textural information of an image at different scales. WT for super-resolution has been applied successfully to multi-frames SR problem [4, 16, 27].

Motivated by the remarkable properties of clique block and WT, we propose a novel network for SR called SRCN to address the above-mentioned challenges. We use clique blocks as the main part of the extraction module. We also design a novel up-sampling module called clique up-sampling. In our clique up-sampling module, we implement a modified version of clique block with four layers to learn the four sub-bands' coefficients of HR images jointly. For magnification factors greater than 2, we design a progressive SRCN upon image pyramids[1]. Our proposed network achieves superior performance over the state-of-the-art methods on benchmark datasets.

2 Our method

In this section, we first describe the overview of the proposed SRCN's architecture, then we introduce the feature embedding net (FEN) and the image reconstruction net (IRN), which are the key parts of our proposed method.

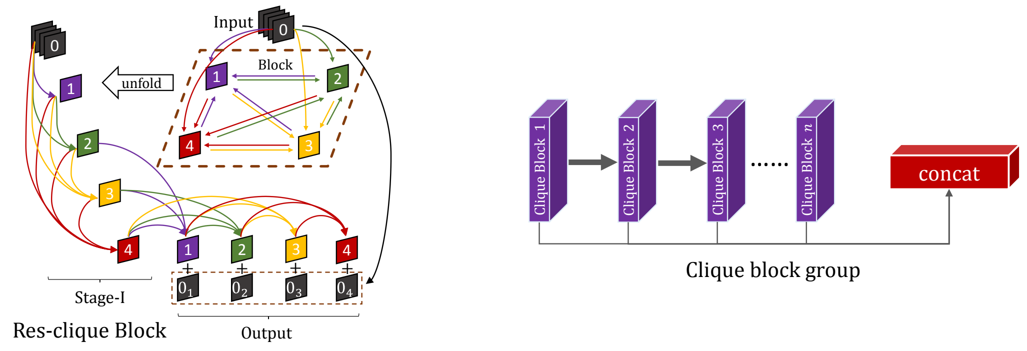

Figure 2: The illustrations of the clique block of residual vision (left) and clique block group (right).

2.1 Network architecture

As shown in Fig. 1, our SRCN mainly consists of two sub-networks: FEN and IRN. FEN represents the LR input image as a set of feature maps. Note that FEN does not change the size (h, w) of the input image, where h and w are the height and the width, respectively. IRN up-samples the feature map got by FEN and reconstructs the HR image. Here we denote $\mathcal{I}^{LR} \in \mathbb{R}^{3\times h\times w}$ as the input LR image and $\mathcal{I}^{HR} \in \mathbb{R}^{3\times rh\times rw}$ as the output HR image, where r is the magnification factor.

2.2 Feature embedding net

As shown in the left part of Fig. 1, FEN starts with two convolutional layers. The first convolutional layer tries to increase the number of channels of input, which can be added with the output of the clique block group via the skip connection. The clique block group will be introduced immediately. The skip connection after the first convolutional layer has been widely used in SR networks [24, 25, 14]. The output of the first convolutional layer is $\mathcal{F}_1 \in \mathbb{R}^{nlg\times h\times w}$, where n is the number of clique blocks that follow, l is the number of layers in each clique block and g is the growth rate of each clique block. The second convlutional layer tries to change the number of channels so that they can fit the input of clique block group. The output of the second convolutional layer is $\mathcal{F}_2 \in \mathbb{R}^{lg\times h\times w}$.

The illustrations of res-clique block and clique block group are shown in Fig. 2. We choose clique block as our main feature extractor for the following reasons. First, clique block's forward propagation contains two stages. The propagation of first stage does the same things as dense block, while the second stage distills the feature further. Second, clique block contains more skip connections compared with the dense block, so the information among layers can be easier propagated. We add a residual connection to the clique block, since the input feature contains plenty of useful information. We call such kind of clique block as res-clique block.

Suppose clique block has l layers and the input and output of the clique block are denoted by $\mathcal{X}_0 \in \mathbb{R}^{lg\times h\times w}$ and $\mathcal{Y} \in \mathbb{R}^{lg\times h\times w}$, respectively. The weight between layer i and layer j is represented by $\mathcal{W}_{ij}$. The feed-forward pass of the clique block can be mathematically described as the following equation. For stage one, $\mathcal{X}_i^{(1)} = \sigma(\sum_{k=1}^{i-1} \mathcal{W}_{ki} * \mathcal{X}_k^{(1)} + \mathcal{W}_{0i} * \mathcal{X}_0)$, where $*$ is the convolution operation. σ is the activation function. For stage two, $\mathcal{X}_i^{(2)} = \sigma(\sum_{k=1}^{i-1} \mathcal{W}_{ki} * \mathcal{X}_k^{(2)} + \sum_{k=i+1}^{l} \mathcal{W}_{ki} * \mathcal{X}_k^{(1)})$. For residual connection, $\mathcal{Y} = [\mathcal{X}_1^{(2)}, \mathcal{X}_2^{(2)}, \mathcal{X}_3^{(2)}, ..., \mathcal{X}_l^{(2)}] + \mathcal{X}_0$, where $[\cdot]$ represents the concatenation operation.

Then we combine n res-clique blocks into a clique block group. The output of clique block group makes use of feature from all preceding clique blocks and can be represented as $\mathcal{C}_{i+1} = f_i(\mathcal{C}_i), i = 1, 2, 3, ..., n-1, \mathcal{C}_i \in \mathbb{R}^{lg\times h\times w}$, where $\mathcal{C}_i$ is the input of i-th blocks and f_i is the underlying mapping of i-th block. $\mathcal{Y}_g = [\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_n] \in \mathbb{R}^{nlg\times h\times w}$ is the output of clique block group. Finally, the output of FEN is a summation of $\mathcal{Y}_g$ and $\mathcal{F}_1$, that is $\mathcal{Y}_F = \mathcal{Y}_g + \mathcal{F}_1$.

2.3 Image reconstruction net

Now we present details about IRN. As shown in the right part of Fig. 1. IRN consists of two parts: clique up-sampling module and a convolutional layer which used to reduce the number of feature maps to reconstruct SR image with 3 channels (RGB).

The clique up-sampling module showing in Fig. 3 is the most important part of IRN. The clique up-sampling module is motivated by discrete wavelet transformation (DWT) and clique block. It contains four sub-nets, representing four sub-bands denoted by LL, LH, HL and HH in the wavelet domain, respectively. Previous CNNs for wavelet domain SR [11, 21] ignore the relationship among the four sub-bands in the wavelet domain. The LL block represents low-pass filtering of the original image at half the resolution. The output feature maps of FEN encode the essential information in the original LR image. So we use the output feature $\mathcal{Y}_F$ to learn the LL block firstly. We represent the number of channels of input feature maps by c, then $\mathcal{Y}_F \in \mathbb{R}^{c\times h\times w}$, $c = nlg$. This process can be written as

$$\mathcal{Y}_{LL}^{(1)} = f_{LL}(\mathcal{Y}_F), \tag{1}$$

where f_{LL} denotes the learnable non-linear function for the LL block. The HL block shows horizontal edges, mostly. In contrast, the LH block mainly contains vertical edges. As illustrated in left part of Fig. 4, we take an image from Set5 [3] as an example. Both the HL and LH blocks can be learned from the LL block and the feature $\mathcal{Y}_F$, written as

$$\mathcal{Y}_{HL}^{(1)} = f_{HL}^{(1)}([\mathcal{Y}_F, \mathcal{Y}_{LL}^{(1)}]),\ \ \mathcal{Y}_{LH}^{(1)} = f_{LH}^{(1)}([\mathcal{Y}_F, \mathcal{Y}_{LL}^{(1)}]), \tag{2}$$

where f_{HL} and f_{LH} denote the learnable function to construct the HL and the LH blocks. The HH block finds edges of the original image in the diagonal direction. Also shown in left part of Fig. 4, the HH block looks similar to the LH and the HL blocks, so we suggest that using LH, HL, LL blocks and the output feature map of FEN could learn the HH block easier than using the feature map alone. We formulate it as

$$\mathcal{Y}_{HH}^{(1)} = f_{HH}([\mathcal{Y}_F, \mathcal{Y}_{LL}^{(1)}, \mathcal{Y}_{HL}^{(1)}, \mathcal{Y}_{LH}^{(1)}]). \tag{3}$$

We name the above-mentioned operations as the sub-band extraction stage. We also plot four histograms at the right part of Fig. 4 to prove that the sub-band extraction step is effective. We apply WDT to 800 images from DIV2K [25] which we use as our training dataset in our experiments and plot histograms of four sub-bands' DWT coefficients of these images. From Fig. 4, we find the distributions of LH, HL and HH blocks are similar to each other. So it is reasonable to use the LH and HL blocks to learn HH block.

The four sub-bands are followed by a few residual blocks after the sub-band extraction stage. Due to high frequency coefficients may be more difficult to learn than low frequency coefficients, we use different number of residual blocks for different sub-band. We denote the numbers of residual blocks of each sub-bands are n_{LL}, n_{HL}, n_{LH} and n_{HH}, respectively. we update each sub-band by the following equation

$$\mathcal{Y}_{LL}^{(1.5)} = f_{LL}^{(1.5)}(\mathcal{Y}_{LL}^{(1)}),\ \mathcal{Y}_{HL}^{(1.5)} = f_{HL}^{(1.5)}(\mathcal{Y}_{HL}^{(1)}),\ \mathcal{Y}_{LH}^{(1.5)} = f_{LH}^{(1.5)}(\mathcal{Y}_{LH}^{(1)}),\ \mathcal{Y}_{HH}^{(2)} = f_{HH}^{(2)}(\mathcal{Y}_{HH}^{(1)}) \tag{4}$$

where $f_{LL}^{(1.5)}, f_{HL}^{(1.5)}, f_{LH}^{(1.5)}, f_{HH}^{(2)}$ represent the residual learnable function of for four sub-bands.

After the operations of resdiual blocks, the IRN enters the sub-band refinement stage. At this stage, we use the high frequency blocks to refine the low frequency blocks, which is an inverse process of the extraction step. Concretely, we use the HH block to learn LH and HL blocks, represented as

$$\mathcal{Y}_{LH}^{(2)} = f_{LH}^{(2)}([\mathcal{Y}_{HH}^{(2)}, \mathcal{Y}_{LH}^{(1.5)}]),\ \ \mathcal{Y}_{HL}^{(2)} = f_{HL}^{(2)}([\mathcal{Y}_{HH}^{(2)}, \mathcal{Y}_{HL}^{(1.5)}]), \tag{5}$$

where $\mathcal{Y}_{HH}^{(2)}$ is the output of residual blocks of the HH sub-band. In a similar way, we update $\mathcal{Y}_{LL}$ by the following equation

$$\mathcal{Y}_{LL}^{(2)} = f_{LL}^{(2)}([\mathcal{Y}_{HH}^{(2)}, \mathcal{Y}_{LH}^{(2)}, \mathcal{Y}_{HL}^{(2)}, \mathcal{Y}_{LL}^{(1.5)}]). \tag{6}$$

Then we apply IDWT to these four blocks, we choose the simplest wavelet, Haar wavelet, for it can be computed by deconvolution operation easily. The dimensions of all blocks are same. They are all $p \times h \times w$, where p represents the number of feature maps produced by each sub-net. So the output of IDWT is $\mathcal{Y}_w = \mathrm{IDWT}([\mathcal{Y}_{LL}^{(2)}, \mathcal{Y}_{HL}^{(2)}, \mathcal{Y}_{LH}^{(2)}, \mathcal{Y}_{HH}^{(2)}]) \in \mathbb{R}^{p\times 2h\times 2w}$. At last, the output of IDWT is sent to a convolutional layer, which used to reduce the number of channels and get the HR image.

2.4 Comparison with clique block and clique up-sampling

Clique block and clique up-sampling have a lot in common. Concretely, clique up-sampling can be viewed as a modified clique block with fixed four layers determined by the formulation of IDWT. The

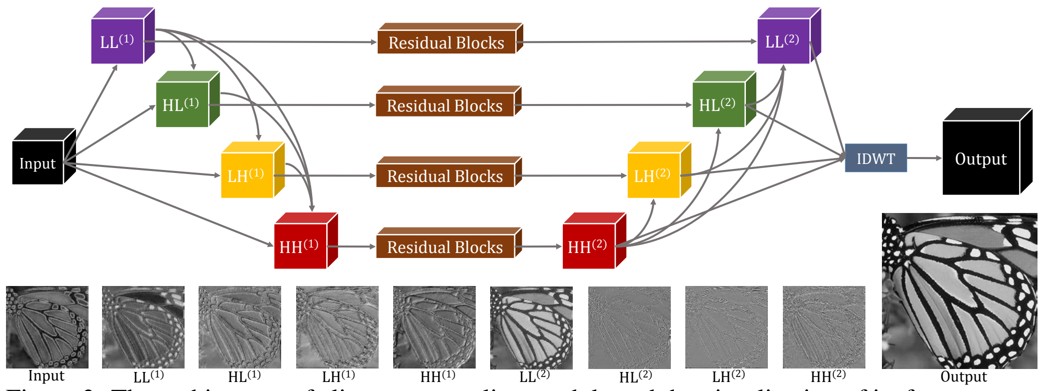

Figure 3: The architecture of clique up-sampling module and the visualization of its feature maps.

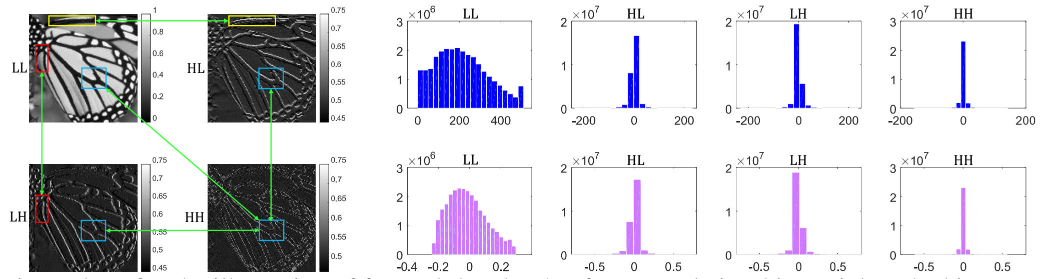

Figure 4: Left: The illustration of four sub-bands edge features relationships. Right: The histograms of four sub-bands' coefficients over 800 images from DIV2K [33]. Top: Do DWT on the original images. Bottom: Do DWT on images preprocessed with mode 4 described in Sec. 3.1.

forward propagations of both clique block and clique up-sampling consist of two stages. However, the clique up-sampling module has another residual refinement stage which represents as stage 1.5 above. Since we consider the edge feature properties of all sub-bands, the HL block shows horizontal edges, mostly, in contrast, the LH block mainly contains vertical edges. the outputs of these two blocks seem to be "orthogonal". So there is no connection between the second and the third layer in clique up-sampling module. At last, the outputs of these two modules are quite different. Concretely, the output of clique block is the concatenation form of the output of all layers, which makes it have more channels. The output of clique up-sampling is the output of all layers after IDWT, which increases the resolution.

2.5 Architecture for magnification factor $2^J\times$

Till now, we have introduced the network architecture for magnification factor $2\times$. In this subsection, we propose SRCN's architecture for magnification factor $2^J\times$, where J is the total level of the network. Image pyramid has been widely used in computer vision applications. LAPGAN [5] and LapSRN [22] used Laplacian pyramid for SR. Motivated by these works, we import image pyramid to our proposed network to deal with magnification factors at $2^J\times$. As shown in the left part of Fig. 5, our model generates multiple intermediate SR predictions in one feed-forward pass through progressive reconstruction. Due to our cascaded and progressive architecture, our final loss consists of J parts: $L = \sum_{j=1}^{J} L_j$. We use the bicubic down-sampling to resize the ground truth HR image $\mathcal{I}^{HR}$ to $\mathcal{I}^j$ at level j. Following [25, 14], we use MAE to measure the performance of reconstruction for each level: $L_j = \mathrm{mean}(|\mathcal{I}^j - \hat{\mathcal{I}}^j|)$, where $\hat{\mathcal{I}}^j$ is the predicted image at level j.

3 Experiments

3.1 Implementation and training details

Model Details. In our proposed SRCN, we set 3×3 as the size of most convolutional layers. We also pad zeros to each side of the input to keep size fixed. We also use a few 1×1 convolutional layers for feature pooling and dimension reduction. The details of our SRCN's setting are presented

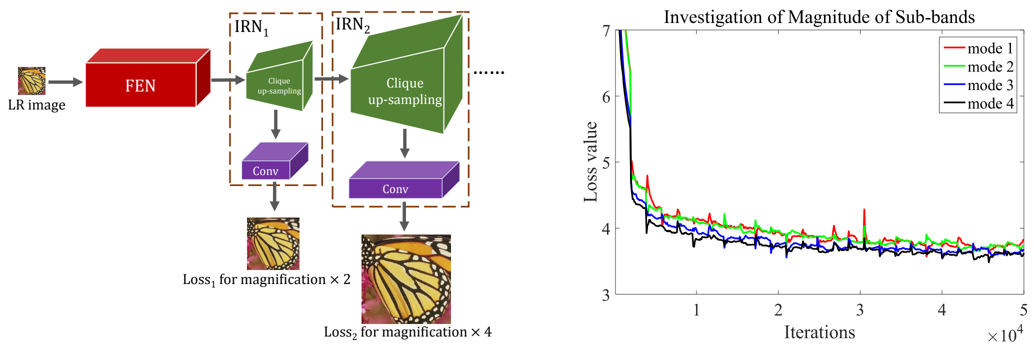

Figure 5: Left: SRCN architecture with magnification factor $4\times$. Right: the performances of input images transformed with 4 modes.

Table 1: Details of our proposed SRCN for magnification factor $2\times, 4\times$.

Models	Clique Block Group	Clique Up-sampling 1	Clique Up-sampling 2
SRCN ($2\times$)	$n = 15, l = 4, g = 32$	$c = 1920, n_{LL} = 2, n_{LH} = 3$ $p = 480, n_{LH} = 3, n_{HH} = 4$	-
SRCN ($4\times$)	$n = 15, l = 5, g = 32$	$c = 2400, n_{LL} = 2, n_{LH} = 3$ $p = 600, n_{LH} = 3, n_{HH} = 4$	$c = 600, n_{LL} = 2, n_{LH} = 3$ $p = 300, n_{LH} = 3, n_{HH} = 4$

in Tab. 1. In Tab. 1, n represents number of clique blocks. l and g represent the number of layer and the growth rate in each clique block, respectively. The numbers of input and output channels of clique up-sampling module are denoted as c and p, respectively. $n_{LL}, n_{LH}, n_{HL}, n_{HH}$ represent the number of residual blocks in the four sub-bands. Unlike most CNNs for computer vision problems, we avoid dropout [29], batch normalization[15] and instance normalization [13], which are not suitable for the SR problem, because they reduce the flexibility of features [25].

Datasets and training details. We trained all networks using images from DIV2K [33] and Flickr [25]. For testing, we used four standard banchmark datasets: Set5 [3], Set14 [40], BSDS100 [2] and Urban100 [12]. Follow settings of [25], we used a batch size of 16 with size 32×32 for LR images, while HR image size changes according to the magnification factor. We randomly augmented the patches by flipping horizontally or vertically and rotating $90°$. We chose parametric rectified linear units (PReLUs) as the activation function for our networks. The base learning rate was initialized to 10^{-5} for all layers and decreased by a factor of 2 for every 200 epochs. The total training epoch was 500. We used Adam [19] as our optimizer and conducted all experiments using PyTorch.

Magnitude of sub-bands. As mentioned before, our clique up-sampling module has four sub-nets and every sub-net has connection with the other sub-nets. Since the feature maps of one sub-band are learned from some other sub-bands', the magnitude of each sub-band block should be similar to others to get better use of each sub-net. As shown in Fig. 4, the histograms of DWT coefficients of original images are in the top right part. The coefficients' magnitude of LL sub-band is quite different from the other three, which may cause training process become difficult. So we want to transform original images to make the difference among magnitudes of four sub-bands decrease. We mainly propose 4 modes:1) Original pixel range from 0 to 255. 2) Each pixel divides 255. 3) Each pixel divides 255 and then subtracts the mean of the dataset by channel. 4) Each pixel divides 255 and then subtracts the mean of the dataset by channel, after the DWT, the coefficients of LL blocks divide a scalar which is around 4 to make the magnitude of LL sub-band more similar to other sub-bands'. The final histograms are showing in Fig. 4 bottom right part. Under the same experiment setting, we pre-processing the input images with 4 modes. The performance of 4 modes are shown in the right part of Fig. 5. From the figure, we can find that mode 4 get better performances in terms of loss value. So in other subsequent experiments, we pre-process our input following mode 4.

Table 2: Investigation of FEN (left) and IRN (right).

Metrics	RB + CU	DB + CU	CB + CU	CB + DC	CB + SC	CB + CU^-	CB + CU
PSNR	37.75	37.83	37.99	37.87	37.89	37.81	37.99
SSIM	0.960	0.960	0.962	0.960	0.961	0.960	0.962

3.2 Investigation of FEN and IRN

To verify the power of res-clique block and clique up-sampling module, we design two contrast experiments. In these two experiments, we used a small version network which contains 8 blocks, each block having 4 layers and each layer producing 32 feature maps. We fixed the clique up-sampling module in IRN and used different blocks i.e, residual block (RB), dense block (DB) and clique block (CB) in FEN in our first experiment. In the second experiment, we fixed the clique blocks in FEN and changed the up-sampling module, i.e, deconvolution (DC), sub-pixel convolution (SC), clique up-sampling without joint learning (CU^-), clique up-sampling (CU). We obeserved the best performance (PSNR/SSIM [36]) on Set5 with magnification factor $2\times$ in 200 epochs. The performances of all kind of settings are listed in Tab. 2.

Tab. 2 shows the power of clique block and clique up-sampling module. When we combine clique block with clique up-sampling module, we get the best performances comparing with other different settings.

We also visualize the feature maps of four sub-bands in two stages. Since the channel's number of the two stages is larger than 3. Specifically, in this paper, we will consider the mean of the feature maps in channel dimension for better visualization, which can be described by $\mathrm{mean}(\mathcal{F}) = \frac{1}{\mathrm{c}} \sum_{\mathrm{i}=1}^{\mathrm{c}} \mathcal{F}_{\mathrm{i},:,:}$. The channel-wise averaged feature maps are shown in Fig. 3. From Fig. 3, we can find that the feature maps of input and stage one do not present the properties of coefficients in wavelet domain. However the feature maps of stage two are close to the coefficients of WT and can reconstruct clear high resolution images after IDWT. The visualization results demonstrate that it is necessary to add sub-band refinement stage in clique upsampling module.

3.3 Comparison with the-state-of-the-arts

To validate the effectiveness of the proposed network, we performed several experiments and visualizations. We compare our proposed network with 8 state-of-the-art SR algorithms: DRCN [18], LapSRN [22], DRRN [31], MemNet [32], SRMDNF [41], IDN [14], D-DBPN [8] and EDSR [25]. We carried out extensive experiments using 4 benchmark datasets mentioned above. We evaluated the SR images with PSNR and SSIM. Tab. 3 shows quantitative comparisons on $2\times$ and $4\times$ SR. Our SRCN performs better than existing methods on most datasets.

In Fig. 6, we show visual comparisons on Set14, BSDS100 and Urban100 with a magnification factor of $4\times$. Our method accurately reconstructs more clear and textural details of English letters and more textural stripes on zebras. For structured architectural style images, our method tends to get more legible recovered images. The comparisons suggest that our method infers the high-frequency details directly in the wavelet domain and the results prove its effectiveness. Also our method got better quantitative results in terms of PSNR and SSIM comparing with other state-of-the-arts.

4 Conclusion

In this paper, we propose a novel CNN called SRCN for SSIR. We design a new up-sampling module called clique up-sampling which uses IDWT to increase the size of feature maps and jointly learn all sub-band coefficients depending on the edge feature property. We design a res-clique block to extract features for SR. We verify the necessity of both two modules on benchmark datasets, also we extend our SRCN with progressive up-sampling module to deal with larger magnification factors. Extensive evaluations on benchmark datasets demonstrate that the proposed network performs better comparing with the state-of-the-art SR algorithms in terms of quantitative metrics. For visual quality, our algorithm also reconstructs more clear and textual details than other state-of-the-arts.

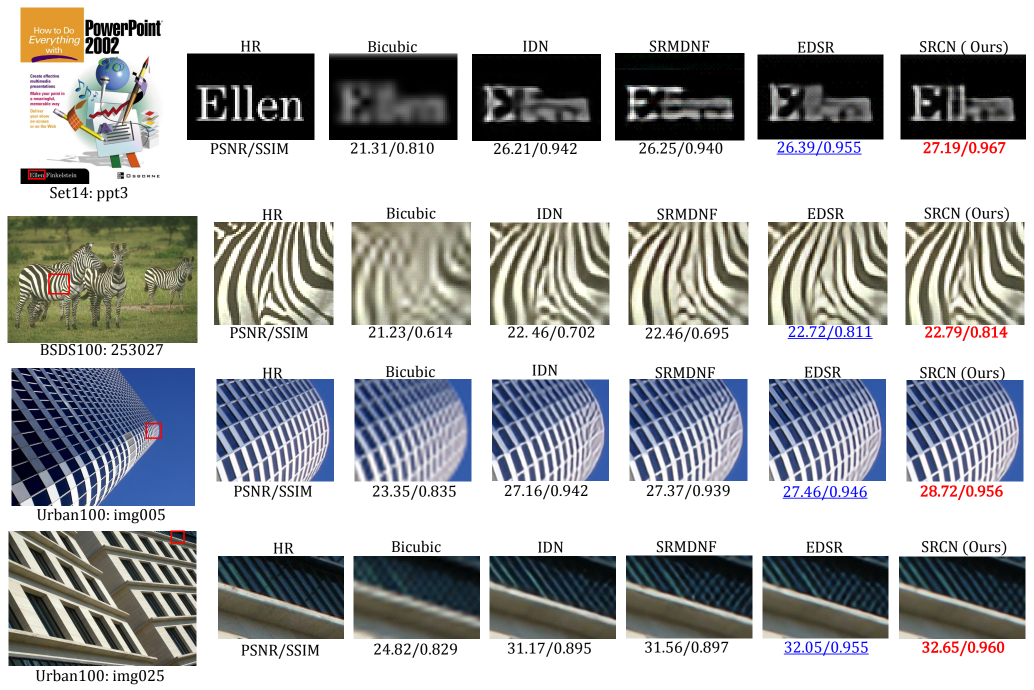

Figure 6: Vision comparisons on images sampling from Set14, BSDS100 and Urban100 with magnification factor $4\times$.

Table 3: Quantitative evaluation of state-of-the-art SR algorithms: average PSNR/SSIM for magnification factors $2\times, 4\times$. **Red** indicates the best and Blue indicates the second best performance. ('-' indicates that the method failed to reconstruct the whole images due to computation limitation mentioned in its paper.)

Models	Magni.	Set5		Set14		BSDS100		Urban100	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Bicubic	$2\times$	33.65	0.930	30.34	0.870	29.56	0.844	26.88	0.841
VDSR [17]	$2\times$	37.53	0.958	32.97	0.913	31.90	0.896	30.77	0.914
DRCN [18]	$2\times$	37.63	0.959	32.98	0.913	31.85	0.894	30.76	0.913
LapSRN [22]	$2\times$	37.52	0.959	33.08	0.913	31.80	0.895	30.41	0.910
DRRN [31]	$2\times$	37.74	0.959	33.23	0.913	32.05	0.897	31.23	0.919
MemNet [32]	$2\times$	37.78	0.960	33.28	0.914	32.08	0.898	31.31	0.920
SRMDNF [41]	$2\times$	37.79	0.960	33.32	0.915	32.05	0.898	31.31	0.920
IDN [14]	$2\times$	37.83	0.960	33.30	0.915	32.08	0.898	31.27	0.920
D-DBPN [8]	$2\times$	38.09	0.960	33.85	0.919	32.27	0.900	-	0.931
EDSR [25]	$2\times$	38.11	0.960	**33.92**	0.919	32.32	0.901	**32.84**	**0.935**
SRCN (Ours)	$2\times$	**38.20**	**0.963**	33.91	**0.923**	**32.34**	**0.905**	32.76	**0.935**
Bicubic	$4\times$	28.42	0.810	26.10	0.704	25.96	0.669	23.15	0.659
VDSR [17]	$4\times$	31.35	0.882	28.03	0.770	27.29	0.726	25.18	0.753
DRCN [18]	$4\times$	31.53	0.884	28.04	0.770	27.24	0.724	25.14	0.752
LapSRN [22]	$4\times$	31.54	0.885	28.19	0.772	27.32	0.728	25.21	0.756
DRRN [31]	$4\times$	31.68	0.888	28.21	0.772	27.38	0.728	25.44	0.764
MemNet [32]	$4\times$	31.74	0.890	28.26	0.772	27.40	0.728	25.50	0.763
SRMDNF [41]	$4\times$	31.96	0.893	28.35	0.777	27.49	0.734	25.68	0.773
IDN [14]	$4\times$	31.82	0.890	28.25	0.773	27.41	0.730	25.41	0.763
D-DBPN [8]	$4\times$	32.47	0.898	28.82	0.786	27.72	0.740	-	0.795
EDSR [25]	$4\times$	32.46	0.897	28.80	0.788	27.71	0.742	**26.64**	**0.803**
SRCN (Ours)	$4\times$	**32.57**	**0.902**	**28.84**	**0.793**	**27.74**	**0.750**	26.55	**0.803**

References

[1] E. H Adelson. Pyramid methods in image processing. *Rca Engineer*, 29, 1984.

[2] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE TPAMI*, (5):898–916, 2011.

[3] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie-Line Alberi Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In *BMVC*, 2012.

[4] Raymond H Chan, Tony F Chan, Lixin Shen, and Zuowei Shen. Wavelet algorithms for high-resolution image reconstruction. *SIAM Journal on Scientific Computing*, (4):1408–1432, 2003.

[5] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, pages 1486–1494, 2015.

[6] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *ECCV*, pages 184–199, 2014.

[7] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *ECCV*, pages 391–407, 2016.

[8] Muhammad Haris, Greg Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. In *CVPR*, 2018.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[10] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *CVPR*, number 2, page 3, 2017.

[11] Huaibo Huang, Ran He, Zhenan Sun, and Tieniu Tan. Wavelet-srnet: A wavelet-based cnn for multi-scale face super resolution. In *ICCV*, pages 1689–1697, 2017.

[12] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *CVPR*, pages 5197–5206, 2015.

[13] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *CVPR*, pages 1501–1510, 2017.

[14] Zheng Hui, Xiumei Wang, and Xinbo Gao. Fast and accurate single image super-resolution via information distillation network. In *CVPR*, 2018.

[15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015.

[16] Hui Ji and Cornelia Fermuller. Robust wavelet-based super-resolution reconstruction: theory and algorithm. *IEEE TPAMI*, (4):649–660, 2009.

[17] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *CVPR*, pages 1646–1654, 2016.

[18] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *CVPR*, pages 1637–1645, 2016.

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *Computer Science*, 2014.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012.

[21] Neeraj Kumar, Ruchika Verma, and Amit Sethi. Convolutional neural networks for wavelet domain super resolution. *Pattern Recognition Letters*, pages 65–71, 2017.

[22] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *CVPR*, pages 624–632, 2017.

[23] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, (11):2278–2324, 1998.

[24] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, pages 4681–4690, 2017.

[25] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *CVPR Workshops*, number 2, page 3, 2017.

[26] Stephane Mallat. Wavelets for a vision. *Proceedings of the IEEE*, (4):604–614, 1996.

[27] M Dirk Robinson, Cynthia A Toth, Joseph Y Lo, and Sina Farsiu. Efficient fourier-wavelet super-resolution. *IEEE TIP*, (10):2669–2681, 2010.

[28] Wenzhe Shi, Jose Caballero, Ferenc Huszar, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, pages 1874–1883, 2016.

[29] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, (1):1929–1958, 2014.

[30] Radomir S Stanković and Bogdan J Falkowski. The haar wavelet transform: its status and achievements. *Computers & Electrical Engineering*, (1):25–44, 2003.

[31] Ying Tai, Jian Yang, and Xiaoming Liu. Image super-resolution via deep recursive residual network. In *CVPR*, number 4, 2017.

[32] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *CVPR*, pages 4539–4547, 2017.

[33] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, Lei Zhang, Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, Kyoung Mu Lee, et al. Ntire 2017 challenge on single image super-resolution: Methods and results. In *CVPR Workshops*, pages 1110–1121. IEEE, 2017.

[34] Tong Tong, Gen Li, Xiejie Liu, and Qinquan Gao. Image super-resolution using dense skip connections. In *ICCV*, pages 4809–4817, 2017.

[35] Zhaowen Wang, Ding Liu, Jianchao Yang, Wei Han, and Thomas Huang. Deep networks for image super-resolution with sparse prior. In *ICCV*, pages 370–378, 2016.

[36] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE TIP*, (4):600–612, 2004.

[37] Yibo Yang, Zhisheng Zhong, Tiancheng Shen, and Zhouchen Lin. Convolutional neural networks with alternately updated clique. In *CVPR*, 2018.

[38] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833, 2014.

[39] Matthew D Zeiler, Graham W Taylor, and Rob Fergus. Adaptive deconvolutional networks for mid and high level feature learning. In *ICCV*, pages 2018–2025, 2011.

[40] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *International conference on curves and surfaces*, pages 711–730. Springer, 2010.

[41] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Learning a single convolutional super-resolution network for multiple degradations. In *CVPR*, 2018.