# Multi-Dimensional Dynamic Model Compression for Efficient Image Super-Resolution

Zejiang Hou
Princeton University
zejiangh@princeton.edu

Sun-Yuan Kung
Princeton University
kung@princeton.edu

## Abstract

*Modern single image super-resolution (SR) system based on convolutional neural networks achieves substantial progress. However, most SR deep networks are computationally expensive and require excessively large activation memory footprints, impeding their effective deployment to resource-limited devices. Based on the observation that the activation patterns in SR networks exhibit high input-dependency, we propose Multi-Dimensional Dynamic Model Compression method that can reduce both spatial and channel wise redundancy in an SR deep network for different input images. To reduce the spatial-wise redundancy, we propose to perform convolution on scaled-down feature-maps where the down-scaling factor is made adaptive to different input images. To reduce the channel-wise redundancy, we introduce a low-cost channel saliency predictor for each convolution to dynamically skip the computation of unimportant channels based on the Gumbel-Softmax. To better capture the feature-maps information and facilitate input-adaptive decision, we employ classic image processing metrics, e.g., Spatial Information, to guide the saliency predictors. The proposed method can be readily applied to a variety of SR deep networks and trained end-to-end with standard super-resolution loss, in combination with a sparsity criterion. Experiments on several benchmarks demonstrate that our method can effectively reduce the FLOPs of both lightweight and non-compact SR models with negligible PSNR loss. Moreover, our compressed models achieve competitive PSNR-FLOPs Pareto frontier compared with SOTA NAS-based SR methods.*

## 1. Introduction

Single image super-resolution (SISR) is a classic low-level computer vision task, which entails recovering a high-resolution (HR) image from a single low-resolution (LR) image, often assumed to be a bicubic downsampled version of the HR couterpart. Since these exists multiple HR images that can be downsampled to the same LR input, the SISR problem is ill-posed. Recently, deep convolutional neural network (CNN) based methods have demonstrated remarkable improvements on SISR. The pioneer work SRCNN [8] designed a three-layer convolutional network, and outperformed previous non deep learning methods significantly. Since then, more complicated and deeper architectures [30, 50, 52] are proposed to expand the CNN representation ability and continuously enhance the SISR performance. Nevertheless, the state-of-the-art SISR networks are computationally more expensive, and require excessively larger memory footprints, compared to image classification CNNs. For example, the FLOPs for processing a $224 \times 224$ color image using the state-of-the-art RDN [52] (x4 SISR) and ResNet-50 (classification) are 1140G and 4.1G, respectively. Moreover, the memory footprint of the SR network can be two order of magnitude larger than the classification network. Therefore, model compression and inference acceleration on SISR CNNs become a necessity.

Channel pruning [27, 34, 35, 16] has been broadly acknowledged as an effective compression approach, due to its advantage in significant speedup and memory reduction of both model storage and intermediate feature-maps. Prior arts evaluate the channel importance over entire training set and discard the least important channels to minimize the accuracy loss. One of the limitations is that the representation ability of the network is permanently reduced after channel pruning. It neglects the diverse demands for network parameters and capacity from different images. thus the pruned model may not regain its accuracy. In addition, conventional methods do not take into consideration that the channel activation patterns exhibit high variation with respect to different input images, which is a more significant phenomena in SISR CNNs, as illustrated in Fig.1. We plot the variation of the maximum activation response for all channels of the first convolution layer in the EDSR-baseline [30] network, given 100 DIV2K validation images. We observe that some input images elicit very high activation values, whereas the other images elicit little response from the same set of channels. This phenomena suggests

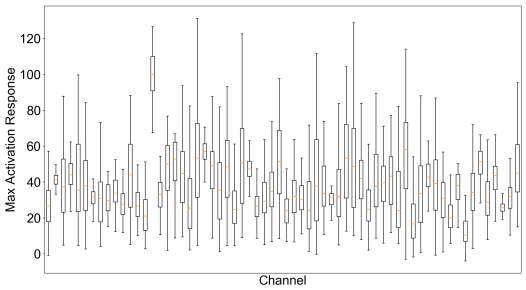
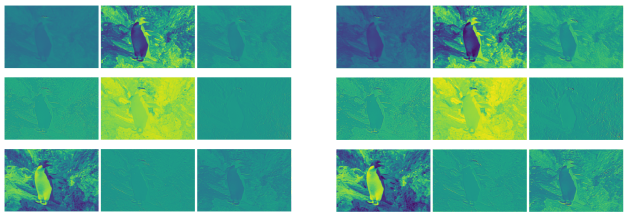



Figure 1: Variation of the maximum activation response for all 64 channels of first convolution in EDSR-baseline, given 100 DIV2K validation images for x4 SISR.

that the relative importance of each channel can vary greatly w.r.t. different input images (i.e., high input-dependency). Last but not least, the spatial redundancy in the intermediate feature-maps is not considered by most of the channel pruning methods. To illustrate the spatial redundancy in SR networks, we conduct the following experiment. In Fig.2(a), we visualize the output feature-maps generated by the first-layer convolution in EDSR-baseline for one DIV2K image. For comparison in Fig.2(b), we downsample the input feature-maps by 2x, then apply the optimized first-layer convolution, and finally upscale the output to original spatial size. As shown, although the convolution is performed on spatially reduced feature-maps in Fig.2(b), the generated output feature-maps have little visual discrepancy compared to Fig.2(a). This result indicates that the spatial redundancy can be reduced with minimal information loss.

To rectify the aforementioned limitations and maximally excavate the redundancy, we propose *Multi-Dimensional Dynamic Model Compression*, that allows the SR network to learn to jointly reduce the spatial and channel wise redundancy for different input images, as depicted in Fig.3. To achieve spatial-wise dynamic compression, we perform convolution on scaled-down feature-maps, where input feature-maps corresponding to different input images are processed at different spatial resolutions based on a learnable router module. To achieve channel-wise dynamic compression, we learn the channel importance w.r.t. different inputs by a channel saliency predictor. Instead of permanently pruning unimportant channels, we accelerate the convolution by selectively computing a subset of channels predicted to be important, while skipping the computation of unimportant ones. The binary gating decisions on the channel-wise computation are made trainable by employing Gumbel-Softmax. Both the routers and saliency predictors take conditioning information computed from the feature-maps as input. To facilitate input-adaptive decisions, we employ classic image processing metrics, e.g., Spatial Information (SI) [20], as the feature-maps statistics to guide the routers and saliency predictors. Compared with naive global average/maximum pooling, SI can capture high-frequency spatial details in the feature-maps better, which



(a) Conv($\mathbf{X}_{\text{in}}$) (b) $f_{\uparrow 2}(\text{Conv}(f_{\downarrow 2}(\mathbf{X}_{\text{in}})))$

Figure 2: Illustrations of spatial redundancy in SR networks. We take the first convolution layer in EDSR-baseline (x4 SR) as an example. (a): output feature-maps generated by the standard convolution. (b): output feature-maps generated by (i) down-scale the spatial resolution of input feature-maps, (ii) perform convolution, (iii) up-scale the output to the original spatial size .

makes it more suitable to SISR. The proposed method can be trained end-to-end with gradient-based optimization using standard super-resolution loss, in conjunction with a sparsity criterion which drives the network to satisfy a computation budget. Experiments on four benchmarks demonstrate that our method can achieve 2x FLOPs reduction with negligible quantitative/qualitative loss for both lightweight (e.g., CARN [2]) and non-compact (e.g., RDN [52]) networks. In addition, our extremely compressed models establish a new series of efficient SISR with superior PSNR-FLOPs Pareto frontier compared with SOTA methods.

## 2. Related Works

**Channel pruning.** Channel pruning methods [27, 35, 34, 16, 54, 15, 32, 48, 7, 31, 28] aim to remove uninformative channels and associated filte parameters to reduce the computation cost and memory requirement of deep CNNs. However, existing channel pruning methods ignore the spatial redundancy, thus limiting the achievable FLOPs reductions. Recently, MDP [12] proposes to jointly prune channels and compress the spatial-temporal dimension. Our method differs that (1) MDP is a static pruning method, thus suffers from the limitations discussed in Section 1: the model capacity is permanently restricted after pruning, and it ignores that the channel importance exhibit high input-dependency. We propose dynamic model compression that can preserve the model capacity and learn input-dependent compression policies by considering diverse demands from inputs. (2) MDP imposes L1 regularization on the scaling factors of channels/branches, and prune channels/branches with small scaling factors, i.e., MDP cannot explicitly control the computation cost of the pruned network. In contrast, we utilize Gumbel-Softmax together with a sparsity criterion so that our compressed models satisfy explicit computation budgets, which is more attractive in real applications.

**Dynamic inference.** Dynamic inference methods provide input-dependent compression strategies and vary the computation cost for each individual input. The pioneer work

ACT [11] dynamically halts the execution of layers when the generated features are good enough for the final classifier. Later methods observe that residual networks are robust to block dropping, and propose to dynamically execute the residual blocks for different inputs. For example, SkipNet [46] and BlockDrop [47] leverages reinforcement learning (RL) to train extra gating modules or policy nets to selectively execute residual blocks. Another line of works prune channels dynamically. For example, FBS [10] rank the channel importance dynamically and only execute the top-$k$ channels. Apart from layer or channel skipping methods, [44] proposes to train pixel-wise gating masks so that the convolution is applied to important spatial regions only. Our proposed dynamic model compression is a differentiable method, due to the usage of Gumbel-Softmax. Thus, our method can be efficiently trained via gradient optimization, in contrast to the inefficient RL-based methods. Moreover, we learn input-dependent compression policies in multi-dimensions, jointly reducing spatial and channel wise redundancy. Thus, our method holds more flexibility than prior arts that singly prune channels or layers.

**Efficient image super-resolution.** Although numerous deep CNN based methods [8, 22, 43, 24, 25, 51, 52, 14, 33, 29, 6, 40] have been proposed for SISR, these models are difficult to deploy on resource-limited devices due to their excessively large computation cost and memory footprint. Therefore, designing efficient SR networks becomes a popular research topic. For example, FSRCNN [9] accelerates SISR by a compact hourglass CNN architecture. DRCN [23] and DRRN [42] adopt recursive layers to build efficient SR networks with fewer parameters. CARN [2], GhostSR [39] reduce the computation cost by combining lightweight residual blocks with variants of group convolutions. SESR [5] proposes a new training protocol for efficient SR by replacing standard convolutions with collapsible linear blocks. In comparison, our method holds better generality such that it can be applied to a variety of architectures. Moreover, Our method can achieve further FLOPs reduction on top of efficient SR models (e.g., CARN) with negligible PSNR loss. Another line of works [36, 53, 38] exploit the attention mechanism to improve discriminate learning and representation power of SR networks, so that a compact network achieves good performance. More recent works apply neural architecture search to SISR by directly searching efficient architectures [26]. While the focus of this paper is not on NAS, we show that our compressed models achieve competitive PSNR-FLOPs trade-off compared with SOTA NAS-based SR models.

# 3. Methodology

The proposed multi-dimensional dynamic model compression is illustrated in Fig.3. When an input image comes in, at each residual block, the router chooses a spatial resolution to perform convolution on. Then, the input feature-maps are down-sampled to the chosen resolution, and processed by convolution layers in the residual block. The output is up-sampled back to the original spatial size and added to the input feature-maps to form a residual learning. For each convolution layer, the Spatial Information (SI) of the input feature-maps are computed and fed into fully-connected (FC) layers to predict the channel importance scores. Then, Gumbel-Softmax (GS) is used to selectively execute the important channels to produce the output.

## 3.1. Dynamic spatial-wise compression

Residual block (RB) has become the building mainstay in modern SR networks. A standard RB is expressed as: $\mathbf{X}_{b+1} = \mathcal{F}(\mathbf{X}_b) + \mathbf{X}_b$, where $\mathbf{X}_b$ is the input feature-maps to $b^{\text{th}}$ RB, and $\mathcal{F}(\cdot)$ is the residual function composed of two convolution layers with a non-linear function in-between. Our method leverages the spatial redundancy of the feature-maps, and makes $\mathcal{F}$ conditional on the spatial resolution.

Specifically, a low-cost router receives the feature-maps $\mathbf{X}_b$ and decides the spatial resolution that the following convolution layers would perform on. The router consists of Spatial Information (SI) operator and one fully-connected (FC) layer with softmax activation. The SI operator [20] measures the amount of spatial details in the feature-maps by calculating the standard deviation over the pixels after Sobel-filtering: $\text{SI}(\mathbf{X}_b) = \text{Std}[\text{Sobel}(\mathbf{X}_b)]$. After the SI operator, we obtain a channel descriptor vector, which is converted into the routing probability distribution by:

$$\mathbf{h}_b = \mathbf{W}_b^{\text{router}} \cdot \text{SI}(\mathbf{X_b}) \tag{1}$$

$$p_i(\mathbf{X}_b) = \frac{\exp([\mathbf{h}_b]_i)}{\sum_{j=1}^{N} \exp([\mathbf{h}_b]_j)} \tag{2}$$

where $\mathbf{W}_b^{\text{router}}$ denotes the parameters of the FC layer in the router, $\mathbf{h}_b$ is the pre-softmax logits, $p_i(\mathbf{X}_b)$ is the probability to choose spatial resolution $i$, and $N$ is the number of available resolution levels.

For a specific spatial resolution $i \in [N]$, we first down-sample the input feature-maps to this RB along the spatial dimension with a factor $i$. For example, $i = 2$ indicates that the spatial size becomes 1/4 of the original size. The down-sampled feature-maps is fed into the residual function $\mathcal{F}$. As mentioned before, our method also perform dynamic channel selection within $\mathcal{F}$ for each convolution layer, which will be detailed out in Sec.3.2. After all the convolution operations in $\mathcal{F}$, we up-sample the output to the original spatial size so that the output feature-maps can be added to the original input feature-maps to form the residual learning. In practice, we adopt average pooling for down-sampling and nearest-neighbour interpolation for up-sampling, due to their efficiencies.

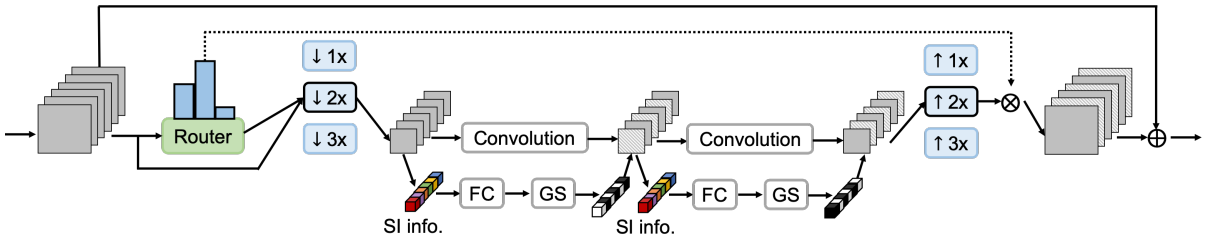



Figure 3: Overview of Multi-Dimensional Dynamic Model Compression for one residual block in SR nets. Sec.3 explains detailed flow.

Based on the softmax distribution in Eq.(2), the spatial resolution corresponding to the maximum probability is selected to perform the convolutions in $\mathcal{F}$. Thus, the RB operated on the selected spatial level $i$ is described as:

$$\mathbf{X}_{b+1} = p_i(\mathbf{X}_b) f_{\uparrow i}(\mathcal{F}(f_{\downarrow i}(\mathbf{X}_b))) + \mathbf{X}_b \tag{3}$$

where $i = \text{argmax}\ \{p_j(\mathbf{X}_b), j = 1, ..., N\}$, $f_{\uparrow i}, f_{\downarrow i}$ represent the up-sampling and down-sampling operation, respectively. We multiply the probability value to the residual output in order to make the router trainable end-to-end.

### 3.2. Dynamic channel-wise compression

For each convolution, our method dynamically determines the most important subset of channels by Gumbel-Softmax, according to the importance scores produced by a channel saliency predictor. The convolution is conducted on the selected channels only to generate the output feature-maps. To make the predictor low-cost, we follow a similar design of the squeeze-and-excitation [18]. We use the SI operator to extract the global information from the feature-maps. Then, two fully-connected (FC) layers (with ReLU in-between) and Sigmoid activation are applied to generate the importance scores, where the first FC has a reduction ratio of $r$ (we set $r = 16$). For the $l^{\text{th}}$ convolution layer, the score vector $\mathbf{g}^l \in \mathbb{R}^{C^l}$ is calculated as:

$$\mathbf{g}^l = \delta(\mathbf{W}^l_{\text{fc1}} \cdot \sigma(\mathbf{W}^l_{\text{fc2}} \cdot \text{SI}(\mathbf{X}^{l-1}))) \tag{4}$$

where $\mathbf{X}^{l-1} \in \mathbb{R}^{C^{l-1} \times H^{l-1} \times W^{l-1}}$ denotes the input feature-maps, $\mathbf{W}^l_{\text{fc1}} \in \mathbb{R}^{C^{l-1} \times C^{l-1}/r}$, $\mathbf{W}^l_{\text{fc2}} \in \mathbb{R}^{C^{l-1}/r \times C^l}$ are the parameters of two FC layers, $\delta, \sigma$ represent Sigmoid and ReLU activation. ($C, H, W$ represent the channel number, spatial height and width, respectively.) The computation cost of this predictor module is cheap, which requires $C^{l-1}H^{l-1}W^{l-1} + {C^{l-1}}^2/r + C^l C^{l-1}/r$ Multi-Adds. This is relatively negligible compared with the computation cost of convolution, which requires $C^{l-1}C^l H^l W^l k^2$ Multi-Adds, where $k$ is the kernel size.

To determine which channels get computed by convolution, we view $\mathbf{g}^l_i$ as the probability to estimate how likely each channel is selected, since Sigmoid squeezes $\mathbf{g}^l_i$ to $[0, 1]$. Directly sampling channels from a Bernoulli distribution using $\mathbf{g}^l_i$ as the probability is a reasonable option. However, this sampling procedure is not differentiable. To address this limitation, we employ the Gumbel-Softmax (GS) trick [21], which performs differentiable sampling to approximate the categorical distribution. GS converts the soft probabilities into hard decisions while enabling back-propagation to update the channel saliency predictor. Formally, GS defines a continuous and differentiable approximation as:

$$z_i = \frac{\exp((\log(\pi_i) + \xi_i)/\tau)}{\sum_{j=1}^{n} \exp((\log(\pi_j) + \xi_j)/\tau)} \tag{5}$$

where $\xi_i$ are noise samples drawn from Gumbel(0,1) distribution, and $\tau$ is a temperature constant. The output of GS becomes identical to a Bernoulli sample as $\tau$ approaches to 0. Since channel selection is binary ($n = 2$ in Eq.(5)), we can further simplify the GS formulation. Take $i^{\text{th}}$ channel in $l^{\text{th}}$ layer as an example, $\pi_1 = \mathbf{g}^l_i$ indicates the probability that such channel is selected for convolution, then $\pi_2 = 1 - \mathbf{g}^l_i$ is the probability to not select the channel. To ease the following presentation, denote the pre-Sigmoid vector in Eq.(4) as $\hat{\mathbf{g}}^l$. Substituting $\pi_1, \pi_2$ into Eq.(5) can reduce the GS formulation into:

$$z_1 = \sigma((\hat{\mathbf{g}}^l_i + \xi_1 - \xi_2)/\tau) \tag{6}$$

Throughout our experiment, we set the GS temperature $\tau = 1$. We adopt the straight-through estimator [3] to generate the hard decisions during forward pass, and gradients are computed from soft samples during backward pass:

$$\mathbf{m}^l_i = \begin{cases} \mathbb{1}_{z_1 > 0.5} = \mathbb{1}_{(\hat{\mathbf{g}}^l_i + \xi_1 - \xi_2)/\tau > 0} & \text{forward} \\ z_1 & \text{backward} \end{cases} \tag{7}$$

where $\mathbf{m}^l_i$ is the decision for $i^{th}$ channel. During inference time, we do not add Gumbel noise (i.e. $\xi_1, \xi_2$=0) for generating the hard decisions. After $\mathbf{m}^l \in \mathbb{R}^{C^l}$ is obtained, the convolution will be conducted on selected channels only:

$$\mathbf{X}^l = f(\hat{\mathbf{W}}^l, \mathbf{X}^{l-1}),\ \hat{\mathbf{W}}^l = \mathbf{W}^l[\boldsymbol{\nu}^l, :, :, :],\ \boldsymbol{\nu}^l = \{i | \mathbf{m}^l_i = 1\} \tag{8}$$

where $f(\cdot, \cdot)$ stands for convolution operation, $\mathbf{W}^l \in \mathbb{R}^{C^l \times C^{l-1} \times H^l \times W^l}$ denotes the convolution tensor. Since we only execute the selected channels, the output feature-maps $\mathbf{X}^l$ has a dimension of $(1 - \eta^l)C^l \times H^l \times W^l$, where

$\eta^l$ is the channel sparsity computed by $1 - |\boldsymbol{\nu}^l|/C^l$. This implies that the subsequent $(l+1)^{\text{th}}$ convolution layer can also make use of the input-side sparsity $\eta^l$. Thus, every convolution can exploit both input and output channel sparsity.

### 3.3. Sparsity loss and training strategy

Without additional constraints, our method becomes accuracy driven only, and the optimal compression policy is to execute each input image at the original resolution with all channels. Therefore, apart from the super-resolution loss, we incorporate a sparsity loss into the training objective. We define a computation budget $\kappa \in (0, 1)$ to represent the relative amount of desired computation. For example, $\kappa = 0.5$ indicates that on average 50% FLOPs of the full model would be executed. Suppose there are $B$ residual blocks (RB) in the SR network, each of which has FLOPs $\mathbb{F}_b$ when there is no compression. With dynamic compression, the averaged FLOPs of $b^{\text{th}}$ RB depends on the spatial resolution $i_b$ in Eq.(3) and the channel sparsity $\eta_b^1, \eta_b^2$ (two convolutions per block) in Eq.(8). Specifically, the post-compression FLOPs of $b^{\text{th}}$ RB can be calculated by: $\mathbb{F}_{\text{b,sp}} = \frac{\eta_b^1 + \eta_b^1 \eta_b^2}{2(i_b)^2} \mathbb{F}_b$. To satisfy the computation budget, we design a sparsity loss (denoted by $\mathcal{L}_{\text{sp}}$) as follows:

$$\mathcal{L}_{\text{sp}} = \log(|\frac{\sum_{b=1}^{B} \mathbb{F}_{\text{b,sp}}}{\sum_{b=1}^{B} \mathbb{F}_{\text{b}}} - \kappa| + 1) \tag{9}$$

During training, this sparsity loss is averaged over images in each mini-batch, and the network learns how to distribute FLOPs among different RBs and images.

To make the reconstructed SR images ($\mathbf{I}_{\text{SR}}$) possess similar visual quality as the ground-truth images ($\mathbf{I}_{\text{GT}}$), we use L1 distance as the super-resolution loss (denotes as $\mathcal{L}_{\text{sr}}$):

$$\mathcal{L}_{\text{sr}} = \|\mathbf{I}_{\text{SR}} - \mathbf{I}_{\text{GT}}\|_1 \tag{10}$$

The final optimization objective involves two kinds of losses, $\mathcal{L}_{\text{sr}}$ for reconstructing high quality images and $\mathcal{L}_{\text{sr}}$ for constraining the computation cost of the SR network:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{sr}} + \alpha \mathcal{L}_{\text{sp}} \tag{11}$$

Optimizing Eq.(11) from scratch produces models with relatively lower PSNR at target computation budget. We conjecture that the reduced ability to learn is due to the interaction between compression policy learning and image reconstruction learning. Therefore, we propose a two-stage training strategy to improve our models' performance:

1. **Pre-training stage**: Train the plain SR network from scratch without incorporating routers (Eq.(1)) and channel salience predictors (Eq.(4)). The training loss involves $\mathcal{L}_{\text{sr}}$ only. This supervised pre-training is able to leverage labeled data to initialize the backbone network parameters, paving way for the searching stage.
2. **Searching stage**: Add routers and channel salience predictors into the SR network. Train the backbone network together with routers and predictors using joint loss Eq.(11). The model learns the optimal input-dependent compression policies across different RBs to satisfy the target FLOPs budget.

## 4. Experiments

We evaluate our method on both lightweight and non-compact SR networks. Both quantitative and qualitative comparisons are provided to verify the efficacy. The superiority to competing methods is also evidenced by ablation studies. All experiments are conducted on NVIDIA Tesla P100 GPUs. Our method is implemented in PyTorch.

### 4.1. Setup

**Datasets and evaluation metrics.** Following [39], we use the 800 RGB images from DIV2K [1] to train our models. In order to compare with previous SOTA methods, we evaluate on four standard benchmark datasets: Set5 [4], Set14 [49], BSD100 [37], and Urban100 [19]. LR images are generated by downsampling the corresponding HR images using the bicubic kernel. Quantitatively, we calculate PSNR and SSIM between the reconstructed SR images and the HR ground-truth on Y-channel of the YCbCr color space.

**Implementation details.** Our method is applied to three SOTA SR architectures: EDSR-baseline [30], CARN [2], and RDN [52]. The FLOPs of these networks range from 389G to 5911G (on DIV2K for x4 SISR), covering both lightweight and non-compact regimes. To improve the performance of our models, we do not compress the first and last convolutoin layers, since they count for very tiny portion of FLOPs. Throughout all experiments, we set $\alpha = 1$ in Eq.(11) for training. We train the SR models with batch-size 16 and RGB input patches of size $48 \times 48$ cropped from the LR input. Standard data augmentations are performed, including random rotation by 90° and random horizontal flipping. In addition, all input images are substracted by the mean values computed over DIV2K images. We adopt ADAM optimizer with $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$. The initial learning rate is set to be $10^{-4}$ and gets halved every 200 epochs for a total of 1000 training epochs.

### 4.2. Comparison with state-of-the-art

**Quantitative results.** In Table 1, we compare the PSNR/SSIM of our compressed models against baseline models and other model compression methods for x4 SISR. The baseline results are obtained by our re-training, which match the results reported in the original papers. Among various competing methods, C-SGD [7] and DI [17] are SOTA static channel pruning methods; Ghost-DW [13] and GhostSR [39] replace the standard convolutions by

Table 1: Quantitative results of our compressed models compared with baseline models and other model compression methods for x4 SISR. Our method achieves better PSNR/SSIM while reducing more FLOPs compared with other methods.

| Scale | Method | FLOPs reduction | Set5 PSNR/SSIM | Set14 PSNR/SSIM | B100 PSNR/SSIM | Urban100 PSNR/SSIM |
|---|---|---|---|---|---|---|
| x4 | EDSR-baseline [30] | - | 32.14 / 0.893 | 28.59 / 0.782 | 27.59 / 0.737 | 26.12 / 0.787 |
| | DI [17] | 50% | 32.04 / 0.891 | 28.50 / 0.779 | 27.52 / 0.735 | 25.90 / 0.780 |
| | Ours | 50% | 32.25 / 0.894 | 28.63 / 0.782 | 27.59 / 0.737 | 26.04 / 0.784 |
| | CARN [2] | - | 32.16 / 0.894 | 28.59 / 0.781 | 27.58 / 0.736 | 26.03 / 0.783 |
| | C-SGD [7] | 22% | 32.08 / 0.893 | 28.52 / 0.780 | 27.56 / 0.735 | 25.93 / 0.780 |
| | Ghost-DW [13] | 19% | 32.14 / 0.894 | 28.59 / 0.781 | 27.57 / 0.735 | 26.02 / 0.782 |
| | GhostSR [39] | 20% | 32.11 / 0.893 | 28.57 / 0.780 | 27.56 / 0.735 | 25.98 / 0.781 |
| | Ours | 50% | 32.20 / 0.894 | 28.61 / 0.781 | 27.57 / 0.736 | 26.05 / 0.783 |
| | RDN [52] | - | 32.47 / 0.899 | 28.81 / 0.787 | 27.72 / 0.742 | 26.61 / 0.803 |
| | GhostSR [39] | 47% | 32.32 / 0.897 | 28.70 / 0.784 | 27.66 / 0.740 | 26.37 / 0.795 |
| | Ours | 47% | 32.39 / 0.897 | 28.78 / 0.785 | 27.68 / 0.740 | 26.41 / 0.795 |

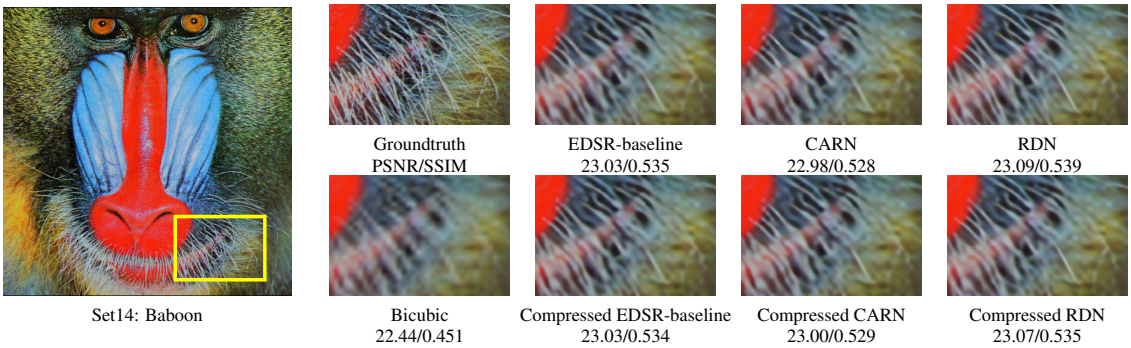



Figure 4: Visual comparisons of baseline models and our compressed models for x4 SISR. Our method can effectively reduce the computation cost of both lightweight and non-compact networks without incurring visual quality loss for SISR.

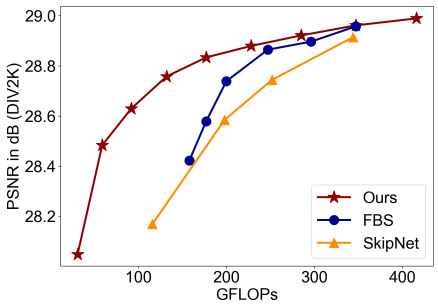



Figure 5: Comparison with SOTA dynamic compression methods, SkipNet [46], FBS [10] for x4 SISR on DIV2K validate set.

cheaper depthwise convolutions or shift operations to reduce FLOPs. For the lightweight CARN network, our method achieves 2x FLOPs reduction while slightly outperforming the baseline on three out of four benchmarks. We attribute this PSNR improvement to the fact that our method can maintain the representation ability by selectively computing a sub-structure based on each input for acceleration. Moreover, due to the two-stage training strategy, our method can leverage the pre-training information, which facilitates the searching stage to learn more effective compression policies for different inputs. On the other hand, the comparative methods achieve very limited (∼20%) FLOPs reduction for CARN, and suffer from PSNR loss. For example, the PSNR of C-SGD drops by 0.1dB on the Urban100 dataset. This is because CARN is compact and static channel pruning greatly reduces the model capacity. Compared with [13, 39], our method also achieves better PSNR values. For more complex and non-compact SR architecture RDN, our method manages to reduce nearly half of FLOPs while yielding higher PSNR/SSIM than [39]. Since our method jointly reduce the spatial and channel wise redundancy, we achieve better PSNR-FLOPs trade-off than other methods focusing only on channel-wise compression.

In Fig.5, we compare with SOTA dynamic compression methods, SkipNet [46] and FBS [10]. PSNR are evaluated with EDSR-baseline for x4 SISR. we plot PSNR against a range of FLOPs reduction ratios. Our method outperforms these two leading approaches under various trade-off between PSNR and FLOPs.

**Qualitative results.** Fig.4 compares the quality of SR images produced by our method versus the baseline for x4 SISR. As observed, the details and textures in images produced by our method demonstrate undetectable visual difference compared to the baseline.

**Comparison with super-efficient SR methods.** In Table 2 and Fig.7, we compare with SOTA super-efficient SR for

Table 2: Quantitative comparison against SOTA **super-efficient** SR methods, including manually designed [9, 45, 5] and NAS-based [26] networks. FLOPs are calculated as the number of multiply-adds needed to convert an image to 720p (1280 × 720) resolution. Red/Blue indicate Best/Second Best in each group. Our compressed models exhibit competitive PSNR-FLOPs Pareto frontier compared with SOTAs.

| Scale | Method | FLOPs | Set5 PSNR / SSIM | Set14 PSNR / SSIM | B100 PSNR / SSIM | Urban100 PSNR / SSIM |
|---|---|---|---|---|---|---|
| x4 | VDSR [22] | 613G | 31.35 / 0.884 | 28.02 / 0.767 | 27.29 / 0.725 | 25.18 / 0.752 |
| | SESR-XL [5] | 6.6G | 31.54 / 0.887 | 28.12 / 0.771 | 27.31 / 0.728 | 25.31 / 0.760 |
| | Ours-L | 6.6G | 31.74 / 0.887 | 28.31 / 0.773 | 27.36 / 0.729 | 25.46 / 0.763 |
| | FEQE-P [45] | 5.6G | 31.53 / 0.882 | 28.21 / 0.771 | 27.32 / 0.727 | 25.32 / 0.758 |
| | FSRCNN [9] | 4.6G | 30.71 / 0.866 | 27.59 / 0.754 | 26.98 / 0.713 | 24.62 / 0.726 |
| | TPSR-NoGAN [26] | 3.6G | 31.10 / 0.878 | 27.95 / 0.766 | 27.15 / 0.721 | 24.97 / 0.746 |
| | Ours-M | 3.6G | 31.53 / 0.884 | 28.19 / 0.770 | 27.29 / 0.727 | 25.24 / 0.756 |
| | SESR-M11 [5] | 1.9G | 31.27 / 0.881 | 27.94 / 0.766 | 27.20 / 0.723 | 25.00 / 0.747 |
| | Ours-S | 1.9G | 31.31 / 0.879 | 28.04 / 0.767 | 27.19 / 0.723 | 25.03 / 0.747 |

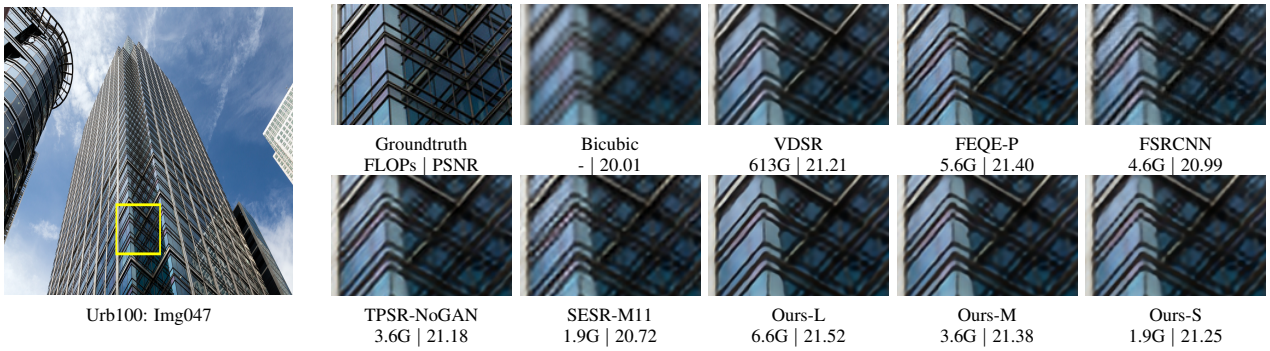



Figure 6: Visual comparison with SOTA super-efficient SR methods for x4 SISR. Our compressed models demonstrate better image quality while requiring similar or fewer FLOPs than other manually designed or NAS-based models.

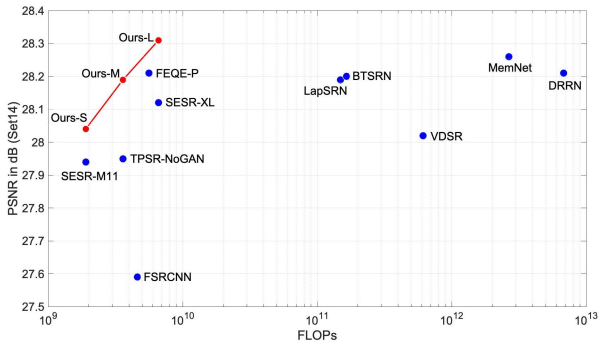


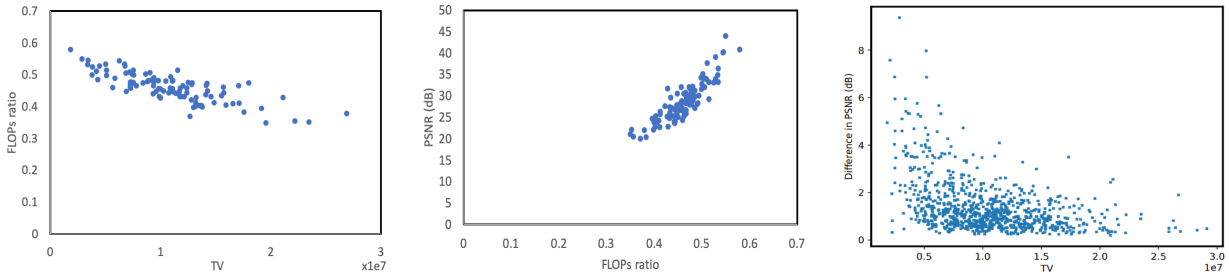

Figure 7: PSNR on Set14 versus FLOPs for different SR networks (x4 SISR). Our method achieves superior PSNR-FLOPs Pareto frontier, compared with SOTA super-efficient SR methods.

x4 SISR, including both manually designed and NAS-based networks. Since we target highly compact SR networks, we apply our method to generate a series of compressed models with extremely large FLOPs reductions. Our super-efficient model are obtained by compressing EDSR-baseline (114 GFLOPs) to 6.6 GFLOPs (Ours-L), 3.6 GFLOPs (Ours-M), and 1.9 GFLOPs (Ours-S). This corresponds to 17x, 32x, and 60x FLOPs reduction, respectively. To the best of our knowledge, SESR [5] is the current best in terms of PSNR-FLOPs trade-off. Notably, Ours-L/S achieves higher PSNR/SSIM than SESR-XL/M11 on all benchmarks, when using the same FLOPs. Compared with FEQE-P [45], Ours-M achieves competitive PSNR/SSIM while requiring 1.6x fewer FLOPs. Moreover, Ours-M outperforms the NAS-based TPSR-NoGAN [26] noticeably when using the same FLOPs. For example, on Set5 dataset, Ours-M gains 0.4dB over TPSR-NoGAN. Finally, we notice that both Ours-L and Ours-M achieve better PSNR than VDSR [22], but require 93x and 170x fewer FLOPs, respectively. Fig.6 shows the image quality of different efficient SR methods. Our method generates better quality SR images with sharper edges and less artifacts than competing methods.

## 5. Results Analysis and Discussion

We conduct ablation studies to further evidence the effectiveness of our method. The results are based on compressing EDSR-baseline for x4 SISR with 50% FLOPs reduction. More results are provided in the supplementary.

**Effect of multi-dimensional compression.** Our method jointly reduces spatial and channel wise redundancy for different input images. To investigate the effectiveness of both components, we perform experiments to dynamically reduce the channel-wise redundancy only, or to reduce the spatial-wise redundancy only. The results are shown in Table 3. As observed, under the same FLOPs reduction, either channel-wise or spatial-wise compression alone yields




(a) (b) (c)

Figure 8: (a) FLOPs ratio for processing each image in DIV2K validation set versus the total variation (TV) values of these images. (b) Achieved PSNR of each image in DIV2K validation set versus FLOPs ratio for processing these images. (c) PSNR difference between two models with substantial capacity versus the TV values for all the DIV2K images.

Table 3: Ablation study of spatial and channel wise dynamic model compression. The proposed multi-dimensional dynamic model compression achieves the highest PSNR by jointly reducing spatial and channel redundancy.

| Channel | Spatial | Set5 | Set14 | B100 | Urban100 |
|---|---|---|---|---|---|
| ✓ | ✗ | 32.16 | 28.56 | 27.55 | 25.98 |
| ✗ | ✓ | 32.20 | 28.61 | 27.57 | 26.00 |
| ✓ | ✓ | 32.25 | 28.63 | 27.59 | 26.04 |

Table 4: Comparison of different feature-maps statistics for guiding routers and saliency predictors. "GMP": global max pooling. "GAP": global average pooling. "SI": spatial information.

| Statistics | Set5 | Set14 | B100 | Urban100 |
|---|---|---|---|---|
| GMP | 32.05 | 28.50 | 27.51 | 25.83 |
| GAP | 32.19 | 28.55 | 27.53 | 25.93 |
| SI (default) | 32.25 | 28.63 | 27.59 | 26.04 |

lower PSNR than jointly compressing multi-dimensions. It is also interesting to note that spatial-wise compression alone leads to higher PSNR than channel-wise compression alone. This result further corroborates the spatial redundancy of intermediate feature-maps in SR networks.

**Comparison of feature-maps statistics.** In our method, we compute the Spatial Information from feature-maps to guide the routers and the channel saliency predictors. In Table 4, we compare SI with global average pooling (GAP) and globabl maximum pooling (GMP) for extracting feature-maps statistics. As observed, SI achieves the highest PSNR on four benchmarks, due to its power to capture the high-frequency spatial details. On the other hand, max pooling operation inevitably causes greater information loss than two other alternatives, it yields the lowest PSNR.

**Understand the learnt compression policy.** We study the correlation between the compression policy and the complexity of input images. In Fig.8(a), we plot the FLOPs (relative to the full model FLOPs, i.e., FLOPs ratio) used to process each image in DIV2K validation versus the complexity of each image measured by total variation (TV) [41]. A larger value of TV implies a more complex image to reconstruct for SISR. In Fig.8(b), we plot the FLOPs ratio versus the PSNR for each image in DIV2K validation. Interestingly, our method learns to assign more FLOPs to process relatively simpler images and achieve better PSNR on these images, while assign fewer FLOPs to process more complex images. This leads to an average of 28.96 dB PSNR on DIV2K validation with 50% FLOPs reduction of EDSR-baseline. To explain this seemingly counter-intuitive compression policy, we conduct the following experiment. We compare two models with substantial capacity difference, one is the full EDSR-baseline and the other is the 0.1x EDSR-baseline. Fig.8(c) shows the PSNR difference between two models versus the TV values for all DIV2K images. As observed, complex images are almost equally hard for both models. While for simpler images, the larger capacity model achieves significantly higher PSNR. The compression policy learnt by our method automatically exploit this property of SR: complex images are processed by fewer FLOPs since the PSNR would not get improved much even with more FLOPs, while simple images are processed with more FLOPs to sustain a high PSNR value. Our method effectively leverage diverse demands for computation from inputs to achieve better FLOPs-PSNR trade-off for SISR.

## 6. Conclusion

This paper proposes Multi-Dimensional Dynamic Model Compression for efficient SISR. For SR networks, we observe that the channel importance exhibit high input-dependency and the spatial redundancy can be reduced by downsampling with minimal information loss. Accordingly, our method learns to jointly reduce spatial and channel wise redundancy and vary the computation cost for different inputs. Our method can be readily applied to a variety of SR networks and trained end-to-end with standard super-resolution loss, in combination with a sparsity criterion. Experiments on several benchmarks and SR models show that our method achieves noticeable FLOPs reduction with negligible quantitative and visual quality loss. Our extremely compressed models achieve superior PSNR-FLOPs trade-off compared with SOTA super-efficient SR methods.

## References


[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 126–135, 2017.

[2] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 252–268, 2018.

[3] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

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

[5] Kartikeya Bhardwaj, Milos Milosavljevic, Alex Chalfin, Naveen Suda, Liam O'Neil, Dibakar Gope, Lingchuan Meng, Ramon Matas, and Danny Loh. Collapsible linear blocks for super-efficient super resolution. *arXiv preprint arXiv:2103.09404*, 2021.

[6] Tao Dai, Jianrui Cai, Yongbing Zhang, Shu-Tao Xia, and Lei Zhang. Second-order attention network for single image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11065–11074, 2019.

[7] Xiaohan Ding, Guiguang Ding, Yuchen Guo, and Jungong Han. Centripetal sgd for pruning very deep convolutional networks with complicated structure. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4943–4953, 2019.

[8] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015.

[9] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *European conference on computer vision*, pages 391–407. Springer, 2016.

[10] Xitong Gao, Yiren Zhao, Łukasz Dudziak, Robert Mullins, and Cheng-zhong Xu. Dynamic channel pruning: Feature boosting and suppression. *arXiv preprint arXiv:1810.05331*, 2018.

[11] Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

[12] Jinyang Guo, Wanli Ouyang, and Dong Xu. Multi-dimensional pruning: A unified framework for model compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1508–1517, 2020.

[13] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. Ghostnet: More features from cheap operations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1580–1589, 2020.

[14] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1664–1673, 2018.

[15] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4340–4349, 2019.

[16] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1389–1397, 2017.

[17] Zejiang Hou and Sun-Yuan Kung. Efficient image super resolution via channel discriminative deep neural network pruning. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3647–3651. IEEE, 2020.

[18] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[19] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2015.

[20] P ITU-T RECOMMENDATION. Subjective video quality assessment methods for multimedia applications. *International telecommunication union*, 1999.

[21] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[22] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1646–1654, 2016.

[23] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Deeply-recursive convolutional network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1637–1645, 2016.

[24] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 624–632, 2017.

[25] Christian Ledig, Lucas Theis, Ferenc Huszár, 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[26] Royson Lee, Łukasz Dudziak, Mohamed Abdelfattah, Stylianos I Venieris, Hyeji Kim, Hongkai Wen, and Nicholas D Lane. Journey towards tiny perceptual super-resolution. In *European Conference on Computer Vision*, pages 85–102. Springer, 2020.

[27] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *ICLR*, 2017.

[28] Yawei Li, Shuhang Gu, Christoph Mayer, Luc Van Gool, and Radu Timofte. Group sparsity: The hinge between filter pruning and decomposition for network compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8018–8027, 2020.

[29] Zhen Li, Jinglei Yang, Zheng Liu, Xiaomin Yang, Gwanggil Jeon, and Wei Wu. Feedback network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3867–3876, 2019.

[30] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.

[31] Mingbao Lin, Rongrong Ji, Yan Wang, Yichen Zhang, Baochang Zhang, Yonghong Tian, and Ling Shao. Hrank: Filter pruning using high-rank feature map. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[32] Shaohui Lin, Rongrong Ji, Chenqian Yan, Baochang Zhang, Liujuan Cao, Qixiang Ye, Feiyue Huang, and David Doermann. Towards optimal structured cnn pruning via generative adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2790–2799, 2019.

[33] Ding Liu, Bihan Wen, Yuchen Fan, Chen Change Loy, and Thomas S Huang. Non-local recurrent network for image restoration. In *Advances in Neural Information Processing Systems*, pages 1673–1682, 2018.

[34] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017.

[35] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, pages 5058–5066, 2017.

[36] Xiaotong Luo, Yuan Xie, Yulun Zhang, Yanyun Qu, Cuihua Li, and Yun Fu. Latticenet: Towards lightweight image super-resolution with lattice block.

[37] David Martin, Charless Fowlkes, Doron Tal, Jitendra Malik, et al. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. Iccv Vancouver:, 2001.

[38] Abdul Muqeet, Jiwon Hwang, Subin Yang, Jung Heum Kang, Yongwoo Kim, and Sung-Ho Bae. Ultra lightweight image super-resolution with multi-attention layers. *arXiv preprint arXiv:2008.12912*, 2020.

[39] Ying Nie, Kai Han, Zhenhua Liu, An Xiao, Yiping Deng, Chunjing Xu, and Yunhe Wang. Ghostsr: Learning ghost features for efficient image super-resolution. *arXiv preprint arXiv:2101.08525*, 2021.

[40] Ben Niu, Weilei Wen, Wenqi Ren, Xiangde Zhang, Lianping Yang, Shuzhen Wang, Kaihao Zhang, Xiaochun Cao, and Haifeng Shen. Single image super-resolution via a holistic attention network. In *European Conference on Computer Vision*, pages 191–207. Springer, 2020.

[41] Leonid I Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: nonlinear phenomena*, 60(1-4):259–268, 1992.

[42] Ying Tai, Jian Yang, and Xiaoming Liu. Image super-resolution via deep recursive residual network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3147–3155, 2017.

[43] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *Proceedings of the IEEE international conference on computer vision*, pages 4539–4547, 2017.

[44] Thomas Verelst and Tinne Tuytelaars. Dynamic convolutions: Exploiting spatial sparsity for faster inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2320–2329, 2020.

[45] Thang Vu, Cao Van Nguyen, Trung X Pham, Tung M Luu, and Chang D Yoo. Fast and efficient image quality enhancement via desubpixel convolutional neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018.

[46] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 409–424, 2018.

[47] Zuxuan Wu, Tushar Nagarajan, Abhishek Kumar, Steven Rennie, Larry S Davis, Kristen Grauman, and Rogerio Feris. Blockdrop: Dynamic inference paths in residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8817–8826, 2018.

[48] Zhonghui You, Kun Yan, Jinmian Ye, Meng Ma, and Ping Wang. Gate decorator: Global filter pruning method for accelerating deep convolutional neural networks. *Advances in neural information processing systems*, 2019.

[49] 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.

[50] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Learning a single convolutional super-resolution network for multiple degradations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3262–3271, 2018.

[51] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 286–301, 2018.

[52] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2472–2481, 2018.

[53] Hengyuan Zhao, Xiangtao Kong, Jingwen He, Yu Qiao, and Chao Dong. Efficient image super-resolution using pixel attention. *arXiv preprint arXiv:2010.01073*, 2020.

[54] Zhuangwei Zhuang, Mingkui Tan, Bohan Zhuang, Jing Liu, Yong Guo, Qingyao Wu, Junzhou Huang, and Jinhui Zhu. Discrimination-aware channel pruning for deep neural networks. In *Advances in Neural Information Processing Systems*, pages 883–894, 2018.