Large Kernel Distillation Network for Efficient Single Image Super-Resolution

Chengxing Xie\textsuperscript{1}\textsuperscript{*} Xiaoming Zhang\textsuperscript{1}\textsuperscript{*} Linze Li\textsuperscript{1} Haiteng Meng\textsuperscript{1} Tianlin Zhang\textsuperscript{2} Tianrui Li\textsuperscript{1} Xiaole Zhao\textsuperscript{1}\textsuperscript{†}
\textsuperscript{1}Southwest Jiaotong University, China
\textsuperscript{2}National Space Science Center, Chinese Academy of Science, China

zxc0074869@gmail.com, xiaoming.zhang@my.swjtu.edu.cn, zxlation@foxmail.com

https://github.com/stella-von/LKDN

Abstract

Efficient and lightweight single-image super-resolution (SISR) has achieved remarkable performance in recent years. One effective approach is the use of large kernel designs, which have been shown to improve the performance of SISR models while reducing their computational requirements. However, current state-of-the-art (SOTA) models still face problems such as high computational costs. To address these issues, we propose the Large Kernel Distillation Network (LKDN) in this paper. Our approach simplifies the model structure and introduces more efficient attention modules to reduce computational costs while also improving performance. Specifically, we employ the re-parameterization technique to enhance model performance without adding extra cost. We also introduce a new optimizer from other tasks to SISR, which improves training speed and performance. Our experimental results demonstrate that LKDN outperforms existing lightweight SR methods and achieves SOTA performance.

1. Introduction

Single image super-resolution (SISR) is an essential problem in low-level computer vision (CV) that involves reconstructing a high-resolution (HR) image from its low-resolution (LR) counterpart. After the introduction of deep learning to super-resolution by SRCNN [9], there has been a significant surge in the development of deep-learning-based SR models. Due to their impressive ability to reconstruct high-resolution images from low-resolution observations, these algorithms have gained popularity in the CV community. Although deeper and larger models are often considered the optimal approach for designing SR models with strong representation ability [31, 57], there is a growing emphasis on developing lightweight models that can approximate the performance of larger models with greatly reduced parameters and less computational complexity.

Among the numerous design approaches for lightweight super-resolution models, information distillation connections [19] have been identified as a highly effective method. This approach fuses features of varying hierarchies to facilitate the transmission of more distinctive features into the network, resulting in efficient feature reuse and achieving a better balance between reconstruction accuracy and computational efficiency.

RFDN [32] reevaluated the information multi-distillation network [19] and introduced multi-feature distillation connections that are highly adaptable and lightweight, and it won the champion of AIM 2020 Efficient SR Challenge [54]. Meanwhile, BSRN [30] achieved the first place in the model complexity track in NTIRE 2022 Efficient SR Challenge [28] by incorporating residual feature distillation connections with effective attention modules and re-parameterizing the redundant convolution of RFDN by using blueprint separable convolution [14] (BSConv). Besides, RFLN [25] emerged as the champion of the runtime track by improving the efficiency of RFDN through the use of residual local feature blocks that reduce network fragments while maintaining model capacity. The newly proposed VAPSR [60] is designed with a concise structure and fewer parameters while achieving SOTA performance. By enhancing the pixel attention mechanism [59], incorporat-
ing large kernel convolutions, and implementing efficient depth-wise separable large kernel convolutions. Additionally, VAPSR achieves comparable performance to RFFDN while utilizing only 28.18% of its parameters and outperforms BSRN.

Although BSRN has made great progress in terms of model parameter and computation, excessive residual connections and complex attention modules (i.e. contrast-aware attention [19] (CCA) and enhanced spatial attention [32] (ESA)) have led to low model computation efficiency. As analyzed in RLFN, the complex ESA module is redundant and it is difficult to quantitatively analyze which parts are truly useful. While VAPSR achieves better performance than BSRN, the model’s running speed is slower due to the presence of a large number of inefficient element-wise multiplications. By removing redundant modules in the network and introducing more efficient ones, we can build a more efficient SR network.

In this paper, we propose a novel lightweight SR network, named large kernel distillation network (LKDN), which builds upon the baseline model of BSRN. Our approach simplifies the model structure and employs a more efficient attention module, called large kernel attention (LKA), to improve model performance and computational costs. We demonstrate the effectiveness of LKA in lightweight SR tasks. Additionally, we leverage the re-parameterization technique to further enhance the performance without adding any additional computational cost. To achieve faster convergence and state-of-the-art (SOTA) performance, we incorporate the recently proposed Adan optimizer [52], which has shown success in various tasks such as high-level CV, natural language processing (NLP), and reinforcement learning. Our proposed LKDN achieves SOTA performance among existing efficiency-oriented SR networks, as shown in Figure 1. The main contributions of this article are:

1. After analyzing the computational efficiency of BSRN [30] and VAPSR [60], We achieved better performance while reducing the number of parameters and computational consumption through simplifying the model structure and introducing a more efficient attention module.
2. We used the technique of re-parameterization to improve the model performance without introducing any additional inference burden.
3. We introduced a new optimizer that can simultaneously boost the training speed and performance of SISR models.

2. Related Work

2.1. Efficient SR Models

The development of lightweight super-resolution networks has received increasing attention in recent years due to their practical applications in resource-constrained scenarios such as mobile devices and embedded systems. Many lightweight SR models have been proposed to reduce the computational cost and memory footprint while maintaining the model capacity and achieving satisfactory performance. The common strategies for lightweight SR networks include network pruning [15, 36, 58], parameter sharing [23, 45], knowledge distillation [12, 16, 56], depth-wise separable convolutions [2, 30, 41, 43], attention mechanisms [13, 19, 32, 59, 60], efficient upsampling methods [26, 40, 47, 59] and re-parameterization technique [3, 55]. In addition to the aforementioned techniques, in terms of network structure design, information distillation connections [19, 25, 30, 32] has also been demonstrated as an effective approach to building lightweight SR network architectures. We thus maintain the network topology design of information distillation while enhancing the attention mechanism and implementing re-parameterization in LKDN.

2.2. Large Kernel Design

Since the introduction of VGG [42], small convolution kernels such as $3 \times 3$ have been widely used due to their high efficiency and lightweight nature. Transformer [49], as a model that achieves a larger receptive field through global self-attention operation, has achieved excellent performance in NLP. In addition, both global [11] and local [34] vision-Transformers have demonstrated impressive performance in the field of CV. This characteristic has inspired researchers to design better convolutional neural networks (CNNs) by utilizing larger convolution kernels. For example, ConvNeXt [35] uses large convolution kernels to obtain a larger receptive field and achieve comparable performance to Swin-Transformer. RepLKNet [7] scales up kernels to $31 \times 31$ using depth-wise convolution and re-parameterization, achieving comparable or superior results to Swin-Transformer on various tasks. VAN [13] explores the effective application of attention mechanisms in CV and proposes a new large kernel attention (LKA) module. SLaK [33] proposes a recipe for applying extremely large kernels from the perspective of sparsity, allowing for the smooth scaling up of kernels to $61 \times 61$ with improved performance. Drawing inspiration from such designs, we developed a large kernel distillation block (LKDB) with large kernel attention (LKA) to further improve the representation ability of LKDN.

2.3. Re-parameterization

Re-parameterization is a pitical technique for designing lightweight models that can improve performance without increasing the inference burden. ACNet [5] employs asymmetric convolution to strengthen kernel structures, yielding better results than normal convolution. RepVGG [8] decomposes a standard $3 \times 3$ convolution into a multi-branch topology comprising identity mapping,
1 × 1 convolution, and 3 × 3 convolution, enabling traditional VGG-style CNNs to achieve similar performance and faster inference speed than SOTA on several high-level vision tasks. Diverse Branch Block [6] combines diverse branches of varying scales and complexities to enrich the feature space, constructing a convolutional network unit resembling Inception [44]. MobileOne [48] leverages the re-parameterization technique to enhance the model’s accuracy and speed, eventually achieving SOTA performance within effective architectures while being many times faster on mobile devices. As a result, we employed re-parameterization techniques to optimize the feature extraction block in LKDN.

2.4. Adan Optimizer

The Adam optimizer [24] is widely utilized in various deep learning domains. By utilizing first and second order gradient moment estimation, it dynamically adjusts the learning rate of each parameter to achieve faster convergence than stochastic gradient descent (SGD). However, Adam can also suffer from non-convergence [39] and local optima [21, 51]. Recently, Adan optimizer [52] combines modified Nesterov impulse, adaptive optimization, and decoupling weight attenuation. Adan uses extrapolation points to perceive gradient information beforehand, allowing for efficient escape from sharp local minima and increasing model generalization. Based on extensive experiments, it has been shown that the Adan optimizer outperforms existing SOTA optimizers for both CNNs and Transformers. Therefore, we aim to apply the Adan algorithm to lightweight super-resolution tasks.

3. Method

3.1. Network Architecture

Our approach follows the same framework design as BSRN [30], as depicted in Figure 2. It comprises four components: shallow feature extraction, multiple stacked feature distillation blocks, multi-layer feature fusion, and image reconstruction block.

Figure 2. The architecture of large kernel distillation network (LKDN).

Figure 3. The details of each component. (a) LKDB: Large Kernel Distillation Block; (b) BSConv: Blueprint Separable Convolution; (c) LKA: Large Kernel Attention; (d) RBSB: Re-parameterized Blueprint Shallow Block.
be expressed as:

\[ I_{LR}^n = \text{Concat}(I_{LR}^n), \]  

where \( \text{Concat}(\cdot) \) denotes the concatenation operation along the channel dimension, and \( n \) is the number of replicated input image \( I_{LR} \). Then the initial feature extraction is implemented by a \( 3 \times 3 \) BSConv to generate shallow features from the input LR image as: \( F_0 = h_{\text{ext}}(I_{LR}^n) \), where \( h_{\text{ext}}(\cdot) \) denotes the module of shallow feature extraction, and \( F_0 \) denotes shallow feature. The structure of BSConv is shown in Figure 3(b), which consists of a \( 1 \times 1 \) convolution and a depth-wise convolution.

The next part of LKDN is to extract deep features through a stack of LKDBs, which can be formulated as:

\[ F_k = H_{\text{LKDB}}^m(\ldots H_{\text{LKDB}}^1(F_0) \ldots), \quad 1 \leq k \leq m, \]  

where \( H_{\text{LKDB}}^m(\cdot) \) denotes the \( k \)th LKDB, \( m \) is the number of used LKDBs, and \( F_k \) represents the output feature of the \( k \)th LKDB.

After gradually refining by the LKDBs, all the intermediate features are fused and activated by a \( 1 \times 1 \) convolution layer and a GELU [17] activation. A \( 3 \times 3 \) BSConv layer is used to smooth the fused features. The process of multi-layer feature fusion can be formulated as:

\[ F_{\text{fusion}} = H_{\text{fusion}}(\text{Concat}(F_1, \ldots, F_k)), \]  

where \( H_{\text{fusion}}(\cdot) \) denotes the feature fusion module, and \( F_{\text{fusion}} \) is the fused features.

Finally, a skip connection is employed in the model to enhance the residual learning and the SR images are obtained through image reconstruction as:

\[ I_{SR} = H_{\text{rec}}(F_{\text{fusion}} + F_0), \]  

where \( H_{\text{rec}}(\cdot) \) denotes the image reconstruction module, and \( I_{SR} \) is the output of the model. The reconstruction process only includes a \( 3 \times 3 \) convolution and pixel-shuffle operation [40].

3.2. Rethinking the BSRN

The performance of BSRN models has been improved by ESA, CCA, and multiple residual connections. However, the complex structure results in lower computational efficiency. RLFN used a pruning sensitivity analysis tool based on a one-shot structured pruning algorithm to analyze the redundancy of the ESA block and discovered a significant amount of redundancy. Therefore, we removed the ESA and CCA modules of BSRN and introduced more efficient attention modules.

Recent studies [13, 60] have shown that the performance of a model can be improved while maintaining acceptable complexity by using large kernel convolution reasonably. VAN [13] utilizes convolution decomposition to split a large kernel convolution into three parts: a spatial local convolution, a spatial long-range convolution, and a channel convolution. Specifically, a \( K \times K \) convolution is decomposed into a \((2d - 1) \times (2d - 1)\) depth-wise convolution, a \([\frac{K}{d}] \times [\frac{K}{d}]\) depth-wise dilation convolution with dilation \( d \), and a \( 1 \times 1 \) convolution. By decomposing large kernel convolution, the model can capture long-range relationships with minimal computational cost and parameters. Similar to VAPS, we perform convolution decomposition in a different order, and as a result, the LKA module is shown in Figure 3(c) and can be expressed as follows:

\[ X_{\text{atten}} = \text{Conv}_{DW-D}(\text{Conv}_{DW}(\text{Conv}_{1x1}(F))), \]  

\[ \text{Output} = X_{\text{atten}} \otimes F, \]  

where \( \text{Conv}_{DW-D}(\cdot) \) and \( \text{Conv}_{DW}(\cdot) \) denotes dilated depth-wise convolution and depth-wise convolution respectively, \( X_{\text{atten}} \) denotes attention map, \( \otimes \) denotes element-wise product operation, and \( F \) denotes the input feature. By decomposing a large \( 17 \times 17 \) convolution into a \( 1 \times 1 \) point-wise convolution, a depth-wise \( 5 \times 5 \) convolution, and a depth-wise dilation convolution with a kernel size of \( 5 \) and dilation of \( 3 \), our model can reduce complexity and improve performance. Replacing ESA and CCA modules with LKA modules can also further improve inference speed.

3.2.1 Re-parameterization

According to [55], excessive skip connection operations can increase memory access cost and inference time. However, the efficient separable distillation block (ESDB) used in BSRN includes four skip connections. To address this issue, we replace the residual connection of the ESDB with a re-parameterizable skip connection and add the branch of BSConv. To replace the Blueprint Shallow Residual Block (BSRB) in BSRN, we introduce the Re-parameterized Blueprint Shallow Block (RBSB) structure, as shown in Figure 3(d). Specifically, we introduce a re-parameterizable skip connection on the \( 1 \times 1 \) point-wise convolution and the \( 3 \times 3 \) depth-wise convolution. Additionally, we parallel a \( 1 \times 1 \) depth-wise convolution on the \( 3 \times 3 \) depth-wise convolution.

The re-parameterization process consists of two steps. First, the input feature \( F_0 \) is passed through a re-parameterized \( 1 \times 1 \) point-wise convolution:

\[ F_1 = \text{Conv}_{1x1}(F_0) + F_0. \]  

Then, a re-parameterized \( 3 \times 3 \) depth-wise convolution is applied:

\[ F_2 = \text{Conv}_{DW3x3}(F_1) + \text{Conv}_{D1x1}(F_1) + F_1. \]
The expressions for $F_1$ and $F_2$ during inference using the re-parameterization technique are thus as follows:

$$F_1 = \text{Conv}_{1 \times 1}(F_0),$$
$$F_2 = \text{Conv}_{DW3 \times 3}(F_0).$$

(9)

### 3.2.2 Large Kernel Distillation Block

After analyzing the characteristics of BSRN and VAPSR, we developed an even more efficient large kernel distillation block (LKDB). The complete structure of LKDB can be seen in Figure 3(a). It comprises four components: feature distillation, feature fusion, feature enhancement, and feature transformation. In the first stages, given the input $F_{in}$, the feature distillation operation can be described as:

$$F_{d_1}, F_{r_1} = D_1(F_{in}), R_1(F_{in}),$$
$$F_{d_2}, F_{r_2} = D_2(F_{r_1}), R_2(F_{r_1}),$$
$$F_{d_3}, F_{r_3} = D_3(F_{r_2}), R_3(F_{r_2}),$$
$$F_{d_4} = D_4(F_{r_3}),$$

(10)

where $D_i$, $R_i$ denote the $i$th distillation and $i$th refinement layer, respectively. $F_{d_i}$, $F_{r_i}$ represents the $i$th distilled features and $i$th refined features, respectively. In the feature fusion stage, all the distilled features produced by previous distillation steps are concatenated together and then fused by a $1 \times 1$ convolution as:

$$F_{fusion} = H_{fusion}(\text{Concat}(F_{d_1}, F_{d_2}, F_{d_3}, F_{d_4})),$$

(11)

where $H_{fusion}$ denotes the $1 \times 1$ convolution layer, and $F_{fusion}$ is the fused feature. For the feature enhancement stage, we introduce an efficiency large kernel attention (LKA) block as:

$$F_{enhance} = H_{LKA}(F_{fusion}),$$

(12)

where $H_{LKA}$ denotes the LKA module, and $F_{enhance}$ is the enhanced feature. To enhance the performance of the model, we employ a $1 \times 1$ convolution in the feature transformation stage, while a pixel normalization [60] module is incorporated to ensure stable model training as:

$$F_{trans} = \text{Norm}_{\text{pixel}}(H_{trans}(F_{enhanced})),$$

(13)

where $H_{trans}$ denotes the $1 \times 1$ convolution transformation, $F_{trans}$ is the transformed feature, and $\text{Norm}_{\text{pixel}}$ refers to the pixel normalization operation [60]. Finally, a long skip connection is used to strengthen the residual learning ability of the model as:

$$F_{out} = F_{trans} + F_{in}.$$  

(14)

### 4. Experiments

#### 4.1. Datasets and Evaluation Metrics

We utilized a training set of 800 images from DIV2K [1] and 2650 images from Flickr2K [31]. Our evaluation of the models is performed on commonly used benchmark datasets, including Set5 [4], Set14 [53], B100 [37], Urban100 [18], and Manga109 [38]. The training data was augmented with random horizontal flips and 90-degree rotations. The evaluation metrics used are the average peak-signal-to-noise ratio (PSNR) and the structural similarity [50] (SSIM) on the luminance ($Y$) channel.

#### 4.2. Implementation Details

The proposed LKDN model is composed of 8 LKDBs with a distillation structure channel number and attention module channel number set to 56, and is trained with BSB to reduce training time. The mini-batch size and input patch size for each LR input are set to 64 and $48 \times 48$, respectively. We train the model using the common $L_1$ loss function and the Adam optimizer [52], with $\beta_1 = 0.98$, $\beta_2 = 0.92$ and $\beta_3 = 0.99$. We set the exponential moving average (EMA) to stabilize training to 0.999. The learning rate is set to a constant $5 \times 10^{-3}$ for the entire $1 \times 10^6$ training iterations.

We propose a smaller version of LKDN, called LKDN-S, for the NTIRE 2023 Efficient SR Challenge [29]. LKDN-S comprises 5 LKDBs and 42 channels, and is trained with RBSB. We also employ re-parameterization techniques in the up-sample layer of LKDN-S. The training process of LKDN-S involves two stages: an initial training stage and a fine-tuning stage. In the initial training stage, we randomly crop 128 mini-batch HR patches with a size of $256 \times 256$. We train LKDN-S using the common $L_1$ loss function with a learning rate of $5 \times 10^{-3}$ and $9.5 \times 10^5$ iterations. In the fine-tuning stage, we set the patch size of HR images and batch size to $480 \times 480$ and 64, respectively. LKDN-S is fine-tuned using the $L_2$ loss function with a learning rate of $2 \times 10^{-5}$, and a total of $5 \times 10^5$ iterations. The EMA is set to 0.999 and Adam optimizer [52], with $\beta_1 = 0.98$, $\beta_2 = 0.92$ and $\beta_3 = 0.99$ is applied in both stages.

We implement all our models using PyTorch 1.11 and Nvidia GeForce RTX 3080 GPUs.

#### 4.3. Ablation Study

##### 4.3.1 Large Kernel Attention

We conducted ablation studies to verify the efficacy of the LKA module, as presented in Table 1. In this table, $C$ and $A$ denote the input channels of the distillation structure and attention module, respectively. Removing the ESA and CCA modules from BSRN resulted in a significant drop in model performance. However, utilizing the LKA module increased the receptive field of the model, leading to
improved performance while maintaining lower parameter and computation costs than BSRN. Further performance improvements were achieved by adjusting the channel numbers of the distillation structure and attention module. Compared to the original BSRN, LKDN can achieve performance gains of more than 0.1 dB on the Urban100 [18] and Manga109 [18] while maintaining lower parameter and computation costs.

### 4.3.2 Re-parameterization

To demonstrate the effectiveness of the proposed RBSB in Figure 3(d), we replaced the basic blocks in the feature distillation structure for comparison. Figure 4(a) displays the BSRB used in BSRN, while Figure 4(b) illustrates the blueprint shallow block (BSB) obtained by directly removing the residual connections. The evaluation results we conducted on LKDN-S are presented in Table 2, demonstrating that eliminating unnecessary residual connections can improve performance while reducing model complexity. Re-parameterization can thus further improve performance while maintaining model complexity.

### 4.3.3 Optimizer

The previous optimization of SR models primarily relied on the Adam optimizer [24]. However, the Adan optimizer [52], which has recently achieved state-of-the-art results on various vision tasks, has piqued the interest of researchers in the field. Therefore, we investigated the effects of the Adan optimizer on SR tasks. Table 3 shows that using the Adan optimizer yields a training speedup of roughly 10% compared to using the Adam optimizer, with a significant performance improvement on various benchmark datasets. Moreover, as demonstrated in Figure 5, the Adan optimizer leads to faster convergence.

![Figure 4. Non-reparameteried components. (a) BSRB: Blueprint Shallow Residual Block; (b) BSB: Blueprint Shallow Block.](image)

![Figure 5. Convergence comparison between Adan [52] and Adam [24] optimizers, using Urban100 [18] SR(×4).](image)

### 4.4. Comparison with the State-of-the-art Methods

We have evaluated our proposed LKDN with various state-of-the-art lightweight SR methods. These methods have been compared for upscale factors of ×2, ×3, and ×4. Table 5 presents the quantitative comparison results for these methods. With the incorporation of efficient attention modules, LKDN has outperformed other methods in terms of achieving the best performance while maintaining a lightweight model.

Figure 6 presents a qualitative comparison of our proposed method. The results indicate that our model is more capable of reconstructing high-similarity structures than existing methods. As an example, in images "img_11" and "img_92", existing methods typically generate obvious distortion and blurriness, while our approach accurately cap-
Table 5. Quantitative comparison with state-of-the-art methods on benchmark datasets. 'Multi-Adds' is calculated with a $1280 \times 720$ GT image. The best and second best are in red and blue respectively.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicubic</td>
<td>×2</td>
<td>-</td>
<td>-</td>
<td>33.66 / 0.9299</td>
<td>30.24 / 0.8688</td>
<td>29.56 / 0.8431</td>
<td>26.88 / 0.8403</td>
<td>30.80 / 0.9339</td>
</tr>
<tr>
<td>SRCNN [9]</td>
<td>×2</td>
<td>8</td>
<td>52.7</td>
<td>36.66 / 0.9542</td>
<td>32.45 / 0.9067</td>
<td>31.36 / 0.8879</td>
<td>29.50 / 0.8946</td>
<td>35.60 / 0.9663</td>
</tr>
<tr>
<td>FSRCNN [10]</td>
<td>×2</td>
<td>13</td>
<td>6.0</td>
<td>37.00 / 0.9558</td>
<td>32.63 / 0.9088</td>
<td>31.53 / 0.8920</td>
<td>29.88 / 0.9020</td>
<td>36.67 / 0.9710</td>
</tr>
<tr>
<td>VDSR [22]</td>
<td>×2</td>
<td>666</td>
<td>612.6</td>
<td>37.53 / 0.9587</td>
<td>33.03 / 0.9124</td>
<td>31.90 / 0.8960</td>
<td>30.76 / 0.9140</td>
<td>37.22 / 0.9750</td>
</tr>
<tr>
<td>LapSRN [26]</td>
<td>×2</td>
<td>251</td>
<td>29.9</td>
<td>37.52 / 0.9591</td>
<td>32.99 / 0.9124</td>
<td>31.80 / 0.8952</td>
<td>30.41 / 0.9103</td>
<td>37.22 / 0.9740</td>
</tr>
<tr>
<td>DRRN [45]</td>
<td>×2</td>
<td>298</td>
<td>6796.9</td>
<td>37.74 / 0.9591</td>
<td>33.23 / 0.9136</td>
<td>32.05 / 0.8973</td>
<td>31.23 / 0.9188</td>
<td>37.88 / 0.9749</td>
</tr>
<tr>
<td>MemNet [46]</td>
<td>×2</td>
<td>678</td>
<td>2662.4</td>
<td>37.78 / 0.9597</td>
<td>33.28 / 0.9134</td>
<td>32.08 / 0.8978</td>
<td>31.31 / 0.9195</td>
<td>37.72 / 0.9740</td>
</tr>
<tr>
<td>IDN [20]</td>
<td>×2</td>
<td>553</td>
<td>124.6</td>
<td>37.83 / 0.9600</td>
<td>33.30 / 0.9148</td>
<td>32.08 / 0.8985</td>
<td>31.27 / 0.9196</td>
<td>38.01 / 0.9749</td>
</tr>
<tr>
<td>CARN [2]</td>
<td>×2</td>
<td>1592</td>
<td>222.8</td>
<td>37.76 / 0.9590</td>
<td>33.52 / 0.9166</td>
<td>32.09 / 0.8978</td>
<td>31.92 / 0.9256</td>
<td>38.36 / 0.9765</td>
</tr>
<tr>
<td>IMDN [19]</td>
<td>×2</td>
<td>694</td>
<td>158.8</td>
<td>38.00 / 0.9605</td>
<td>33.63 / 0.9177</td>
<td>32.19 / 0.8996</td>
<td>31.27 / 0.9283</td>
<td>38.88 / 0.9774</td>
</tr>
<tr>
<td>PAN [59]</td>
<td>×2</td>
<td>261</td>
<td>70.5</td>
<td>38.00 / 0.9605</td>
<td>33.59 / 0.9181</td>
<td>32.18 / 0.8997</td>
<td>32.01 / 0.9273</td>
<td>37.80 / 0.9773</td>
</tr>
<tr>
<td>LAPAR-A [27]×2</td>
<td>548</td>
<td>171.0</td>
<td>38.01 / 0.9605</td>
<td>33.62 / 0.9183</td>
<td>32.19 / 0.8999</td>
<td>32.10 / 0.9283</td>
<td>38.67 / 0.9772</td>
<td></td>
</tr>
<tr>
<td>RFDN [32]</td>
<td>×2</td>
<td>535</td>
<td>95.0</td>
<td>38.05 / 0.9606</td>
<td>33.68 / 0.9184</td>
<td>32.16 / 0.8994</td>
<td>32.12 / 0.9278</td>
<td>38.88 / 0.9773</td>
</tr>
<tr>
<td>RFLN [25]</td>
<td>×2</td>
<td>527</td>
<td>115.4</td>
<td>38.07 / 0.9607</td>
<td>33.72 / 0.9187</td>
<td>32.22 / 0.9000</td>
<td>32.33 / 0.9299</td>
<td>-</td>
</tr>
<tr>
<td>BSNR [30]</td>
<td>×2</td>
<td>332</td>
<td>73.0</td>
<td>38.10 / 0.9610</td>
<td>33.74 / 0.9193</td>
<td>32.24 / 0.9006</td>
<td>32.34 / 0.9303</td>
<td>39.14 / 0.9782</td>
</tr>
<tr>
<td>VAPSR [60]</td>
<td>×2</td>
<td>329</td>
<td>74.0</td>
<td>38.08 / 0.9612</td>
<td>33.77 / 0.9195</td>
<td>32.27 / 0.9011</td>
<td>32.45 / 0.9316</td>
<td>-</td>
</tr>
<tr>
<td>LKDN</td>
<td>×2</td>
<td>304</td>
<td>69.1</td>
<td>38.12 / 0.9612</td>
<td>33.90 / 0.9202</td>
<td>32.27 / 0.9010</td>
<td>32.53 / 0.9322</td>
<td>39.19 / 0.9784</td>
</tr>
</tbody>
</table>

Table 4 provides a more in-depth analysis of BSRN, VAPSR, and LKDN. The results show that LKDN outperforms BSRN while maintaining a comparable inference speed. Moreover, LKDN achieves faster inference speeds than VAPSR while maintaining superior performance.

4.5. NTIRE 2023 Efficient SR Challenge

The goal of the NTIRE 2023 Efficient SR Challenge [29] is to develop a SISR model that improves one or more aspects such as runtime, parameters, FLOPs, activations, and
5. Conclusion

We propose the Large Kernel Distillation Network (LKDN) as an efficient single-image super-resolution (SISR) solution. Through careful analysis of some state-of-the-art lightweight models, we identified their weaknesses and improved upon them to enhance the performance of LKDN. By incorporating the large kernel design, simplifying the model structure, introducing more efficient attention modules, and employing re-parameterization, we achieve a balance between performance and computational efficiency. A new optimizer is also introduced to accelerate the convergence of LKDN. Extensive experiments demonstrate that LKDN outperforms state-of-the-art efficient SR methods in terms of performance, parameters, and Multi-Adds.

Acknowledgment This work is supported in part by National Natural Science Foundation of China under Grant No. 62102330 and in part by Natural Science Foundation of Sichuan Province under Grant Nos. 2022NSFSC0947 and 2022NSFSC0945.
References

[7] Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In CVPR, pages 11963–11975, 2022. 2


[36] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. ICLR, 2018. 2


