Lightweight Bimodal Network for Single-Image Super-Resolution via Symmetric CNN and Recursive Transformer

Guangwei Gao\(^1,3\)\(^*\), Zhengxue Wang\(^1\)\(^*\), Juncheng Li\(^2\)\(^†\), Wenjie Li\(^1\), Yi Yu\(^3\), Tieyong Zeng\(^2\)

\(^1\)Nanjing University of Posts and Telecommunications, Nanjing, China
\(^2\)The Chinese University of Hong Kong, Hong Kong, China
\(^3\)National Institute of Informatics, Tokyo, Japan

\{csggao, cvjunchengli\}@gmail.com, wzx_0826@163.com

Abstract

Single-image super-resolution (SISR) has achieved significant breakthroughs with the development of deep learning. However, these methods are difficult to be applied in real-world scenarios since they are inevitably accompanied by the problems of computational and memory costs caused by the complex operations. To solve this issue, we propose a Lightweight Bimodal Network (LBNet) for SISR. Specifically, an effective Symmetric CNN is designed for local feature extraction and coarse image reconstruction. Meanwhile, we propose a Recursive Transformer to fully learn the long-term dependence of images thus the global information can be fully used to further refine texture details. Studies show that the hybrid of CNN and Transformer can build a more efficient model. Extensive experiments have proved that our LBNet achieves more prominent performance than other state-of-the-art methods with a relatively low computational cost and memory consumption. The code is available at https://github.com/IVIPLab/LBNet.

1 Introduction

Single image super-resolution (SISR) aims to recover the corresponding high-resolution (HR) image with rich details and better visual quality from its degraded low-resolution (LR) one. Recently, convolutional neural networks (CNN) based SISR methods have achieved remarkable performance than traditional methods due to their powerful feature extraction ability. For example, Dong et al. [Dong et al., 2014] pioneered the Super-Resolution Convolutional Neural Network (SRCNN). Later, with the emergence of ResNet [He et al., 2016] and DenseNet [Huang et al., 2017], plenty of CNN-based SISR models have been proposed, like VDSR [Kim et al., 2016a], EDSR [Lim et al., 2017], and RCAN [Zhang et al., 2018]. All these methods show that the deeper the network, the better the performance. However, these methods are difficult to be used in real-life scenarios with limited storage and computing capabilities. Therefore, a model that can achieve better performance while keeping the lightweight of the network has become attractive research. One of the most widely used strategies is to introduce the recursive mechanism, such as DRCN [Kim et al., 2016b] and DRRN [Tai et al., 2017]. The other one is to explore the lightweight structure, including CARN [Ahn et al., 2018], FDIWN [Gao et al., 2022], and PFFN [Zhang et al., 2021a]. Although these models reduce the number of model parameters to a certain extent through various strategies and structures, they also lead to a degradation in performance, thus it is difficult to reconstruct high-quality images with rich details.

Recently, with the continuous progress of Transformer in Natural Language Processing (NLP), how to apply it to computer vision tasks has become a hot topic. Transformer can model the long-term dependence in the image, and this powerful representation ability can help to restore the texture details of the image. However, most methods blindly use Transformer to replace all original CNN structures, this is unreasonable since the ability of CNN to extract local features is irreplaceable. These features can maintain their own stability under different viewing angles, also called local invariance, which is helpful for image understanding and reconstruction. Therefore, we recommend fusing CNN and Transformer to take full use of the advantages of both to achieve efficient SR image reconstruction.

To achieve this, we propose a Lightweight Bimodal Network (LBNet) for SISR. In LBNet, we use both CNN and Transformer to achieve dual-mode coordination reconstruction. As for the CNN part, we focus on local features extraction. Specifically, we propose a novel Local Feature Fusion Module (LFFM), which consists of a series of Feature Refinement Dual-Attention Block (FRDAB). FRDAB uses the channel reduction strategy to reduce the parameters of the model and introduces channel attention and spatial attention mechanisms to reweight the feature information extracted from different branches. Meanwhile, to balance the performance and size of the model, we introduce the parameter sharing strategy to construct the symmetric-like network, the output of the corresponding shared module of the previous stage will be integrated through the channel attention module as the input of the current module. This method can maximize the use of feed-forward features to restore texture details. As for the Transformer part, we propose a Recursive Transformer to learn the long-term dependence of images, thus the texture

\(^*\)Co-first authors
\(^†\)Corresponding author
details can be further refined with global information. In summary, the main contributions are as follows

- We propose an effective Symmetric CNN for local feature extraction and coarse image reconstruction. Among them, Local Feature Fusion Module (LFFM) and Feature Refinement Dual-Attention Block (FRDAB) are specially designed for feature extraction and utilization.

- We propose a Recursive Transformer to learn the long-term dependence of images. This is the first attempt of the recursive mechanism in Transformer, which can refine the texture details by global information with few parameters and GPU memory consumption.

- We propose a novel Lightweight Bimodal Network (LBNet) for SISR. LBNet elegantly integrates CNN and Transformer, enabling it to achieve a better balance between the performance, size, execution time, and GPU memory consumption of the model.

2 Related Works

2.1 CNN-based SISR

Thanks to the powerful feature representation and learning capabilities of CNN, CNN-based SISR methods have made great progress in recent years [Li et al., 2021a]. For example, SRCNN [Dong et al., 2014] applied CNN to SISR for the first time and achieved competitive performance at the time. EDSR [Lim et al., 2017] greatly improved the model performance by using the residual blocks [He et al., 2016]. RCAN [Zhang et al., 2018] introduced the channel attention mechanism and built an 800-layer network. Apart from these deep networks, many lightweight SISR models also have been proposed in recent years. For instance, Ahn et al. [Ahn et al., 2018] proposed a lightweight Cascaded Residual Network (CARN) by using the cascade mechanism. Hui et al. [Hui et al., 2019] proposed an Information Multi-Distillation Network (IMDN) by using the distillation and selective fusion strategy. MADNet [Lan et al., 2020] used a dense lightweight network to enhance multi-scale feature representation and learning. Xiao et al. [Xiao et al., 2021] proposed a simple but effective deep lightweight model for SISR, which can adaptively generate convolutional kernels based on the local information of each position. However, the performance of these lightweight models is not ideal since they disable obtaining larger receptive fields and global information.

2.2 Transformer-based SISR

In order to model the long-term dependence of images, more and more researchers pay attention to Transformer, which was first used in the field of NLP. Recently, many Transformer-based methods have been proposed for computer vision tasks, which also promote the development of SISR. For example, Chen et al. [Chen et al., 2021] proposed a pre-trained Image Processing Transformer for image restoration. Liang et al. [Liang et al., 2021] proposed a SwinIR by directly migrating the Swin Transformer to the image restoration task and achieved excellent results. Lu et al. [Lu et al., 2022] proposed an Effective Super-resolution Transformer (ESRT) for SISR, which reduces GPU memory consumption through a lightweight Transformer and feature separation strategy. However, all these models do not fully consider the fusion of CNN and Transformer, thus difficult to achieve the best balance between model size and performance.

3 Lightweight Bimodal Network (LBNet)

3.1 Network Architecture

As shown in Figure 1, Lightweight Bimodal Network (LBNet) is mainly composed of Symmetric CNN, Recursive Transformer, and reconstruction module. Specifically, Symmetric CNN is proposed for local feature extraction and Recursive Transformer is designed to learn the long-term dependence of images. We define $I_{LR}$, $I_{SR}$, and $I_{HR}$ as the input LR image, the reconstructed SR image, and the corresponding HR image, respectively. At the head of the model, a $3 \times 3$ convolutional layer is applied for shallow feature extraction

$$F_{sf} = f_{sf}(I_{LR}),$$

where $f_{sf}()$ represents the convolutional layer, $F_{sf}$ is the extracted shallow features. Then, the extracted shallow features will be sent to Symmetric CNN for local feature extraction

$$F_{CNN} = f_{CNN}(F_{sf}),$$

where $f_{CNN}()$ represents the Symmetric CNN and $F_{CNN}$ represents the extracted local features. Symmetric CNN is
one of the most important components in LBNet, which consists of several pairs of parameter-sharing Local Feature Fusion Modules (LFFMs) and channel attention modules. All of these modules will be introduced in the next section.

After that, all these features will be sent to the Recursive Transformer for long-term dependence learning

\[ F_{RT} = f_{RT}(F_{CNN}), \]

where \( f_{RT}(\cdot) \) is the Recursive Transformer and \( F_{RT} \) is the feature enhanced by global information. Finally, the refined features \( F_{RT} \) and the shallow features \( F_{SF} \) are added and sent to the reconstruction module for SR image reconstruction

\[ I_{SR} = f_{build}(F_{SF} + F_{RT}), \]

where \( f_{build}(\cdot) \) is the reconstruction module, which composed of a 3 x 3 convolutional layer and a pixel-shuffle layer.

During training, LBNet is optimized with L1 loss function. Given a training dataset \( \{I_{LR}^1, I_{HR}^1\}^N \), we solve

\[ \hat{\theta} = \arg \min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \left\| F_\theta(I_{LR}^i) - I_{HR}^i \right\|_1, \]

where \( \theta \) denotes the parameter set of our proposed LBNet, \( F(I_{LR}^i) = I_{SR} \) is the reconstructed SR image, and \( N \) is the number of the training images.

### 3.2 Symmetric CNN

Symmetric CNN is specially designed for local feature extraction, which mainly consists of some paired parameter-sharing Local Feature Fusion Modules (LFFMs) and Channel Attention (CA) modules. The parameter-sharing of every two symmetrical modules can better balance the parameters and performance. In addition, each pair of parameter-sharing modules will be fused through the channel attention module, so that the extracted features can be fully utilized.

As shown in Figure 1, Symmetric CNN is a dual-branch network. The shallow feature \( F_{SF} \) will first be sent to the top branch and the outputs of each LFFM in the top branch will serve as part of the input of the corresponding LFFM in the down branch. The complete operation can be defined as

\[ F_{LFFM}^{1,i} = f_{LFFM}^{1,i}(F_{SF}), \quad i = 1, \]

\[ F_{LFFM}^{T,i} = f_{LFFM}^{T,i}(F_{LFFM}^{T,i-1}), \quad i = 2, ..., n, \]

\[ F_{LFFM}^{D,i} = f_{LFFM}^{D,i}(F_{LFFM}^{D,i-1} + f_{CA}(F_{LFFM}^{T,i})), \quad i = 2, ..., n, \]

where \( f_{LFFM}^{1,i}(\cdot) \) and \( f_{LFFM}^{D,i}(\cdot) \) represent the \( i \)-th LFFM in the top and down-branch, respectively. \( f_{CA}(\cdot) \) denotes the \( i \)-th channel attention module. It is worth noting that when \( i = 1, F_{LFFM}^{D,1} = f_{LFFM}^{D,1}(F_{LFFM}^{T,1} + f_{CA}(F_{LFFM}^{T,1})). \) Moreover, the weight sharing strategy is applied on the paired modules, thus \( f_{LFFM}^{D,i}(\cdot) = f_{LFFM}^{D,i}(\cdot). \) Finally, the outputs of all these LFFMs are concatenated and a 1 x 1 convolutional layer is used for feature fusion and compression. Therefore, the most effective features extracted at different levels will be sent to the next part to learn the long-term dependence of images.

#### Local Feature Fusion Module (LFFM)

LFFM is the core component of Symmetric CNN. As shown in Figure 2 (a), LFFM is essentially an improved version of DenseBlock [Huang et al., 2017]. Different from DenseBlock, (1) we use FRDAB to replace the original convolutional layer to make it have a stronger feature extraction ability; (2) we introduce a 1 x 1 group convolutional layer before each FRDAB for dimensionality reduction; (3) local residual learning is introduced to further promote the transmission of information. The complete operation of LFFM can be defined as

\[ F_{LFFM}^{1,i} = f_{LFFM}^{1,i}(F_{LFFM}^{T,i-1}), \]

\[ F_{LFFM}^{2,i} = f_{LFFM}^{2,i}(f_{LFFM}^{1,i}(F_{LFFM}^{T,i-1})), \]

\[ F_{LFFM}^{3,i} = f_{LFFM}^{3,i}(f_{LFFM}^{2,i}(F_{LFFM}^{T,i-1})), \]

where \( F_{LFFM}^{1,i} \) and \( F_{LFFM}^{2,i} \) represent the output of the \( i \)-th \( (i = 1, 2, 3) \) FRDAB module in LFFM. \( f_{LFFM}^{1,i}(\cdot) \) means the \( j \)-th \( (j = 1, 2) \) group convolutional layer followed by FRDAB. \( F_{LFFM}^{m-1,i} \) and \( F_{LFFM}^{m,i} \) represent the input and output of the \( m \)-th LFFM module, respectively.

#### Feature Refinement Dual-Attention Block (FRDAB)

As shown in Figure 2 (b), FRDAB is a dual-attention block, which specially designed for feature refinement. Specifically, the multi-branch structure is designed for feature extraction and utilization. In this part, the feature will be sent to two
branches and each branch uses a different number of convolutional layers to change the size of the receptive field to obtain different scales features. \( c/2 \) indicates the operations of halving the outputs. After that, channel attention is used to extract channel statistics for re-weighting in the channel dimension and spatial attention is used to re-weighted the pixel according to the spatial context relationship of the feature map. Finally, the output of these two attention operations is fused by the addition operation. With the help of this method, the finally obtained features will show a stronger suppression of the smooth areas of the input image.

3.3 Recursive Transformer

As we mentioned before, Symmetric CNN is designed for local feature extraction. However, this is far from enough to reconstruct high-quality images since the depth of the lightweight network makes it difficult to have a large enough receptive field to obtain global information. To solve this problem, we introduce Transformer to learn the long-term dependence of images and propose a Recursive Transformer (RT). Different from previous methods, we introduced the recursive mechanism to allow the Transformer to be fully trained without greatly increasing GPU memory consumption and model parameters. As shown in Figure 1, RT is located before the reconstruction module, which consists of two Transformer Modules (TM) and two convolutional layers. The completed operation of RT can be defined as

\[
F_{RT} = f_{3 \times 3}(f_{TM}^1(f_{3 \times 3}(f_{TM}^2(F_{CNN})))), \tag{13}
\]

where \( f_{3 \times 3}(\cdot) \) and \( f_{TM}(\cdot) \) represent the convolutional layer and the TM, respectively. \( \circ \) denotes the recurrent connection, which means that the output of TM will be served as its new input and looped \( S \) times. As for the TM, we only use the encoding part of the standard Transformer structure inspired by ESRT [Lu et al., 2022]. As shown in Figure 3, TM is mainly composed of two layer normalization layers, one Multi-Head Attention (MHA), and one Multi-Layer Perception (MLP). Define the input embeddings as \( F_{in} \), the output embeddings \( F_{out} \) can be obtained by

\[
F_{mid} = F_{in} + f_{MHA}(f_{norm}(F_{in})), \tag{14}
\]

\[
F_{out} = F_{mid} + f_{MLP}(f_{norm}(F_{mid})), \tag{15}
\]

where \( f_{norm}(\cdot) \) represents the layer normalization operation. \( f_{MHA}(\cdot) \) and \( f_{MLP}(\cdot) \) represent the MHA and MLP modules, respectively. Like ESRT, we project the input feature map of MHA into \( Q, K, \) and \( V \) through a linear layer to reduce GPU memory consumption. Meanwhile, feature reduction strategy is also used to further reduce the memory consumption of the Transformer. Following [Vaswani et al., 2017], each head of the MHA must perform a scaled dot product attention, and then concatenate all the outputs and perform a linear transformation to obtain the output. Where the scaled dot product attention can be expressed as

\[
\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V. \tag{16}
\]

This is the first attempt of the recursive mechanism in Transformer. With the help of this strategy, we can fully train and utilize Transformer without increasing the parameters and GPU memory consumption of the model. We will further discuss its effectiveness in the next section.

4 Experiments

4.1 Datasets and Evaluation Metrics

Following previous works, we use DIV2K as training set. For evaluation, we use five benchmark test datasets to validate the effectiveness of LBNet, including Set5 [Bevilacqua et al., 2012], Set14 [Zeyde et al., 2010], BSDS100 [Martin et al., 2001], Urban100 [Huang et al., 2015], and Manga109 [Matsui et al., 2017]. In addition, PSNR and SSIM are used as evaluation indicators to evaluate the performance of SR images on the Y channel of the YCbCr color space.

4.2 Implementation Details

During training, we randomly crop \( 48 \times 48 \) patches from the training set as the inputs, and enhance the images with random rotation and horizontal flipping. The initial learning rate is \( 2 \times 10^{-4} \) and finally dropped to \( 6.25 \times 10^{-6} \) by cosine annealing. Meanwhile, the model is trained by Adam optimizer under the PyTorch framework with a NVIDIA RTX 2080Ti GPU. In the final model, the input and output channels of each part are set to 32, three \( (n = 3) \) LFFMs are used, and Transformer module recurses 2 times \( (S = 2) \). Moreover, we also built a tiny version, named LBNet-T, which only consists of two \( (n = 2) \) LFFMs. Meanwhile, the number of channels in LBNet-T has also been reduced to 18.

4.3 Comparison with Lightweight SISR Models

In Table 1, we compare our LBNet with 11 advanced lightweight SISR models. Most of them achieve the best results at the time in the lightweight SISR task. According to the table, we can clearly observe that our LBNet achieves the best results, and our tiny version LBNet-T also achieves the best results under the same parameter level. In addition, the number of parameters and Mult-Adds of LBNet and LBNet-T are also very low, which proves the efficiency of our models. Meanwhile, we also provide the visual comparison between LBNet and other lightweight SISR models in Figure 5. Obviously, SR images reconstructed by our LBNet have richer detailed textures with better visual effects. This further validates the effectiveness of our proposed LBNet.

4.4 Model Complexity Studies

As can be seen from Table 1, our model achieves an excellent balance between model size, performance, and Mult-Adds. In addition, the execution time of the model is also an important indicator to measure the complexity of the model. To make...
For a more intuitive comparison with other models, we provide the trade-offs between model performance, parameter quantity, and execution time in Figure 4. Obviously, our LBNet achieved the best PSNR results under the premise of comparable execution time and parameters. This further illustrates that LBNet is an efficient and lightweight SISR model.

### 4.5 Ablation Study

**Symmetric CNN Investigations.** In Symmetric CNN, each pair of LFFMs will be fused through a channel attention module. In order to verify the effectiveness of this fusion method, we conduct a series of experiments to study the different feature interaction methods in Symmetric CNN. It is worth noting that we removed the Transformer part of the model to speed up the training time in this ablation study. Table 2 shows the results of these different feature interaction methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Scale</th>
<th>Params</th>
<th>Mult-Adds</th>
<th>PSNR/SSIM Set5</th>
<th>PSNR/SSIM Set14</th>
<th>PSNR/SSIM BSD100</th>
<th>PSNR/SSIM Urban100</th>
<th>PSNR/SSIM Manga109</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDSR [Kim et al., 2016a]</td>
<td>665K</td>
<td>612.6G</td>
<td>33.60/0.9213</td>
<td>29.77/0.8314</td>
<td>28.82/0.7976</td>
<td>27.14/0.8279</td>
<td>32.01/0.9310</td>
<td></td>
</tr>
<tr>
<td>DRRN [Tai et al., 2017]</td>
<td>297K</td>
<td>6796.9G</td>
<td>34.03/0.9244</td>
<td>29.96/0.8349</td>
<td>28.95/0.8004</td>
<td>27.53/0.8378</td>
<td>32.74/0.9390</td>
<td></td>
</tr>
<tr>
<td>IDN [Hui et al., 2018]</td>
<td>553K</td>
<td>56.3G</td>
<td>34.11/0.9253</td>
<td>29.99/0.8354</td>
<td>28.85/0.8013</td>
<td>27.42/0.8359</td>
<td>32.71/0.9381</td>
<td></td>
</tr>
<tr>
<td>CARN [Ahn et al., 2018]</td>
<td>1592K</td>
<td>118.8G</td>
<td>34.29/0.9255</td>
<td>30.29/0.8407</td>
<td>29.96/0.8034</td>
<td>28.06/0.8493</td>
<td>33.43/0.9427</td>
<td></td>
</tr>
<tr>
<td>IMDN [Hui et al., 2019]</td>
<td>701K</td>
<td>71.5G</td>
<td>34.36/0.9270</td>
<td>30.32/0.8417</td>
<td>29.09/0.8046</td>
<td>28.17/0.8519</td>
<td>33.61/0.9445</td>
<td></td>
</tr>
<tr>
<td>AWRSN-M [Wang et al., 2019]</td>
<td>×3</td>
<td>1143K</td>
<td>116.6G</td>
<td>34.42/0.9275</td>
<td>30.32/0.8419</td>
<td>29.13/0.8059</td>
<td>28.26/0.8545</td>
<td>33.64/0.9450</td>
</tr>
<tr>
<td>MADRN [Lan et al., 2020]</td>
<td>930K</td>
<td>88.4G</td>
<td>34.16/0.9253</td>
<td>30.21/0.8398</td>
<td>28.98/0.8023</td>
<td>27.77/0.8439</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>GLADSR [Zhang et al., 2021b]</td>
<td>812K</td>
<td>88.2G</td>
<td>34.41/0.9272</td>
<td>30.37/0.8418</td>
<td>29.08/0.8050</td>
<td>28.24/0.8537</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SMR [Wang et al., 2021]</td>
<td>993K</td>
<td>156.8G</td>
<td>34.40/0.9270</td>
<td>30.33/0.8412</td>
<td>29.10/0.8050</td>
<td>28.25/0.8536</td>
<td>33.68/0.9445</td>
<td></td>
</tr>
<tr>
<td>LAPAR-A [Li et al., 2021b]</td>
<td>594K</td>
<td>114.0G</td>
<td>34.36/0.9267</td>
<td>30.34/0.8421</td>
<td>29.11/0.8054</td>
<td>28.15/0.8523</td>
<td>33.51/0.9441</td>
<td></td>
</tr>
<tr>
<td>LBNet-T (Ours)</td>
<td>407K</td>
<td>22.0G</td>
<td>34.33/0.9264</td>
<td>30.25/0.8402</td>
<td>29.05/0.8042</td>
<td>28.06/0.8485</td>
<td>33.48/0.9433</td>
<td></td>
</tr>
<tr>
<td>LBNet (Ours)</td>
<td>736K</td>
<td>68.4G</td>
<td>34.47/0.9277</td>
<td>30.38/0.8417</td>
<td>29.13/0.8061</td>
<td>28.42/0.8559</td>
<td>33.82/0.9460</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Scale</th>
<th>Params</th>
<th>Mult-Adds</th>
<th>PSNR/SSIM Set5</th>
<th>PSNR/SSIM Set14</th>
<th>PSNR/SSIM BSD100</th>
<th>PSNR/SSIM Urban100</th>
<th>PSNR/SSIM Manga109</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDSR [Kim et al., 2016a]</td>
<td>665K</td>
<td>612.6G</td>
<td>31.39/0.8888</td>
<td>28.01/0.7674</td>
<td>27.29/0.7251</td>
<td>25.18/0.7524</td>
<td>28.83/0.8809</td>
<td></td>
</tr>
<tr>
<td>DRRN [Tai et al., 2017]</td>
<td>297K</td>
<td>6796.9G</td>
<td>31.68/0.8888</td>
<td>28.21/0.7720</td>
<td>27.38/0.7284</td>
<td>25.44/0.7638</td>
<td>29.46/0.8960</td>
<td></td>
</tr>
<tr>
<td>IDN [Hui et al., 2018]</td>
<td>553K</td>
<td>32.3G</td>
<td>31.82/0.8903</td>
<td>28.25/0.7730</td>
<td>27.41/0.7297</td>
<td>25.41/0.7632</td>
<td>29.41/0.8942</td>
<td></td>
</tr>
<tr>
<td>CARN [Ahn et al., 2018]</td>
<td>1592K</td>
<td>90.9G</td>
<td>32.13/0.8937</td>
<td>28.60/0.7806</td>
<td>27.58/0.7349</td>
<td>26.07/0.7837</td>
<td>30.42/0.9070</td>
<td></td>
</tr>
<tr>
<td>IMDN [Hui et al., 2019]</td>
<td>715K</td>
<td>40.9G</td>
<td>32.21/0.8948</td>
<td>28.58/0.7811</td>
<td>27.56/0.7353</td>
<td>26.04/0.7838</td>
<td>30.45/0.9075</td>
<td></td>
</tr>
<tr>
<td>AWRSN-M [Wang et al., 2019]</td>
<td>×4</td>
<td>1254K</td>
<td>72.0G</td>
<td>32.21/0.8954</td>
<td>28.65/0.7832</td>
<td>27.60/0.7368</td>
<td>26.15/0.7884</td>
<td>30.56/0.9093</td>
</tr>
<tr>
<td>MADRN [Lan et al., 2020]</td>
<td>1002K</td>
<td>54.1G</td>
<td>31.95/0.8917</td>
<td>28.44/0.7780</td>
<td>27.47/0.7327</td>
<td>25.76/0.7746</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>GLADSR [Zhang et al., 2021b]</td>
<td>826K</td>
<td>52.6G</td>
<td>32.14/0.8940</td>
<td>28.62/0.7813</td>
<td>27.59/0.7361</td>
<td>26.12/0.7851</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SMR [Wang et al., 2021]</td>
<td>1006K</td>
<td>89.1G</td>
<td>32.12/0.8932</td>
<td>28.55/0.7808</td>
<td>27.55/0.7351</td>
<td>26.11/0.7868</td>
<td>30.54/0.9085</td>
<td></td>
</tr>
<tr>
<td>LAPAR-A [Li et al., 2021b]</td>
<td>659K</td>
<td>94.0G</td>
<td>32.15/0.8944</td>
<td>28.61/0.7818</td>
<td>27.61/0.7366</td>
<td>26.14/0.7871</td>
<td>30.42/0.9074</td>
<td></td>
</tr>
<tr>
<td>LBNet-T (Ours)</td>
<td>410K</td>
<td>12.6G</td>
<td>32.08/0.8933</td>
<td>28.54/0.7802</td>
<td>27.54/0.7358</td>
<td>26.00/0.7819</td>
<td>30.37/0.9059</td>
<td></td>
</tr>
<tr>
<td>LBNet (Ours)</td>
<td>742K</td>
<td>38.9G</td>
<td>32.29/0.8960</td>
<td>28.68/0.7832</td>
<td>27.62/0.7382</td>
<td>26.27/0.7906</td>
<td>30.76/0.9111</td>
<td></td>
</tr>
</tbody>
</table>

**Table 1:** Average PSNR/SSIM comparison. The best and second best results are highlighted with red and blue, respectively.

**Figure 4:** Model complexity study on Set5 (×4).

To verify the effectiveness of our FRDAB, we replace FRDAB with some commonly used feature extraction modules in lightweight SISR models, like IMDB [Hui et al., 2019] and RCAB [Zhang et al., 2018]. It can be seen from Table 3 that although FRDAB will bring a little increase of parameters and Multi-Adds, its performance has been significantly improved. This gain is considerable, which benefits for lightweight models construction. This fully proves that FRDAB is an effective feature extraction module.
In order to learn the long-term dependence of images, we introduced the Transformer and proposed a Recursive Transformer (RT). To verify the effectiveness of the proposed RT, we remove RT and provide the results in Table 5. According to the table, we can clearly observe that our LBNet can reconstruct realistic SR images with accurate edges.

**Recursive Transformer Investigations.** In order to learn the long-term dependence of images, we introduced the Transformer and proposed a Recursive Transformer (RT). To verify the effectiveness of the proposed RT, we remove RT and provide the results in Table 5. According to the table, we can clearly observe that the introduced RT will increase the execution time and GPU memory consumption of the model. However, the increase in Multi-adds and execution time is insignificant. In addition, the experiment also shows that the PSNR/SSIM results of the model with RT have been significantly improved. It shows that the introduced Transformers can improve the learning capabilities of the model thus improving the model performance.

Different from previous Transformers, we introduced the recursive mechanism in the Transformer Module (TM). This design can make the Transformer more fully utilized without increasing the number of model parameters. To verify the effectiveness of the recursive mechanism in Transformer, we conduct a series of experiments with different recursion times. In Table 6, TM-N denotes TM recursed N times and TM-0 represents the flat model that no recursive mechanism is used. Obviously, as the number of recursion times increases, the performance of the model will further improve. Meanwhile, we also notice that when the number of recursion times is 3 (TM-3), the improvement of model performance is not obvious. Therefore, we set the recursion times as 2 ($S = 2$) in the final model to achieve a better balance between mode performance, Multi-Adds, and execution time.

**Comparison with Other Transformer-based Methods.** Recently, some Transformer-based methods have been proposed for SISR. In Table 4, we provide a detailed comparison between SwinIR [Liang et al., 2021] and ESRT [Lu et al., 2022]. According to the results, we can clearly observe that our LBNet achieved competitive results with fewer parameters and Multi-Adds. Although the PSNR result of LBNet is 0.14dB worse than SwinIR, it is worth noting that SwinIR uses an additional dataset (Flickr2K) for training. This is one of the key factors to further improve model performance. All these results further verify the effectiveness of LBNet.

### Table 4: Comparison with other Transformer-based methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Params</th>
<th>Multi-Adds</th>
<th>PSNR/SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SwinIR</td>
<td>897K</td>
<td>49.6G</td>
<td>32.44/0.8976</td>
</tr>
<tr>
<td>ESRT</td>
<td>751K</td>
<td>67.7G</td>
<td>32.19/0.8947</td>
</tr>
<tr>
<td>LBNet (Ours)</td>
<td>742K</td>
<td>38.9G</td>
<td>32.29/0.8960</td>
</tr>
</tbody>
</table>

### Table 5: Study of Recursive Transformer (RT) on Set5 dataset ($\times 4$).

<table>
<thead>
<tr>
<th>Method</th>
<th>Params</th>
<th>Multi-Adds</th>
<th>Running time</th>
<th>PSNR/SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o RT</td>
<td>365K</td>
<td>38.9028G</td>
<td>0.0168s</td>
<td>32.07/0.8929</td>
</tr>
<tr>
<td>with RT</td>
<td>742K</td>
<td>38.9032G</td>
<td>0.0274s</td>
<td><strong>32.23/0.8949</strong></td>
</tr>
</tbody>
</table>

### Table 6: Study on the recursion times of Transformer Module (TM) on Set5 ($\times 4$). The final version is highlighted.

<table>
<thead>
<tr>
<th>Method</th>
<th>Params</th>
<th>Multi-Adds</th>
<th>Running time</th>
<th>PSNR/SSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>TM-0</td>
<td>741.7K</td>
<td>38.9032G</td>
<td>0.0274s</td>
<td>32.23/0.8949</td>
</tr>
<tr>
<td>TM-1</td>
<td>741.7K</td>
<td>38.9036G</td>
<td>0.0356s</td>
<td>32.27/0.8958</td>
</tr>
<tr>
<td>TM-2</td>
<td>741.7K</td>
<td><strong>38.9039G</strong></td>
<td><strong>0.0401s</strong></td>
<td><strong>32.29/0.8960</strong></td>
</tr>
<tr>
<td>TM-3</td>
<td>741.7K</td>
<td>38.9043G</td>
<td>0.0516s</td>
<td>32.30/0.8960</td>
</tr>
</tbody>
</table>

5 Conclusions

In this paper, we proposed a Lightweight Bimodal Network (LBNet) for SISR via Symmetric CNN and Recursive Transformer. Specifically, we proposed an effective Symmetric CNN for local feature extraction and proposed a Recursive Transformer to learn the long-term dependence of images. In Symmetric CNN, Local Feature Fusion Module (LFFM) and Feature Refinement Dual-Attention Block (FRDAB) are designed to ensure sufficient feature extraction and utilization. In Recursive Transformer, the recursive mechanism is introduced to fully train the Transformer, so the global information learned by the Transformer can further refine the features. In summary, LBNet elegantly integrates CNN and Transformer, achieving a better balance between the performance, size, execution time, and GPU memory consumption of the model.

### Acknowledgments

This work was supported in part by the National Key R&D Program of China (No.2021YFE0203700), the National Natural Science Foundation of China (No.61972212), the Natural Science Foundation of Jiangsu Province (No.BK20190089), the Six Talent Peaks Project in Jiangsu Province (No.RJFW-01), and the CRF (No.8730063).
References


