symmetry

MDPI

Article

A Dynamic Fusion of Local and Non-Local Features-Based Feedback Network on Super-Resolution

Yuhao Liu [1,*] and Zhenzhong Chu [2]

[1] College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China
[2] School of Mechanical Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China
* Correspondence: liuyuhao@shmtu.edu.cn

Abstract: Many Symmetry blocks were proposed in the Single Image Super-Resolution (SISR) task. The Attention-based block is powerful but costly on non-local features, while the Convolutional-based block is good at efficiently handling the local features. However, assembling two different Symmetry blocks will generate an Asymmetry block, making the classic Symmetry-block-based Super-Resolution (SR) architecture fail to deal with these Asymmetry blocks. In this paper, we proposed a new Dynamic fusion of Local and Non-local features-based Feedback Network (DLNFN) for SR, which focus on optimizing the traditional Symmetry-block-based SR architecture to hold two Symmetry blocks in parallel, making two Symmetry-blocks working on what they do best. (1) We introduce the Convolutional-based block for the local features and Attention-based network block for non-local features and propose the Delivery–Adjust–Fusion framework to hold these blocks. (2) we propose a Dynamic Weight block (DW block) which can generate different weight values to fuse the outputs on different feedback iterations. (3) We introduce the MAConv layer to optimize the In block, which is critical for our two blocks-based feedback algorithm. Experiments show our proposed DLNFN can take full advantage of two different blocks and outperform other state-of-the-art algorithms.

Keywords: single-image super-resolution; self-attention; feedback network; dynamic weight; deep convolutional network

Citation: Liu, Y.; Chu, Z. A Dynamic Fusion of Local and Non-Local Features-Based Feedback Network on Super-Resolution. *Symmetry* **2023**, *15*, 885. https://doi.org/10.3390/sym15040885

Academic Editors: Jiankang Zhang, Bin Li and Alexander Zaslavski

Received: 16 February 2023
Revised: 1 April 2023
Accepted: 7 April 2023
Published: 9 April 2023

1. Introduction

Single-Image Super-Resolution (SISR) aims to reconstruct a high-resolution (HR) image from a single low-resolution (LR) input image. The mapping between LR and HR images is not bijective, leading to a challenging and ill-posed problem. The recent success of deep neural network based image SR achieved significant success [1], such as Convolutional [2], Feedback [3], and Attention [4]-based HR.

The most recent advance mainly introduces only one block to extract features from LR images to generate SR images; different blocks have own unique advantages: the Convolutional-based SR [2] is adept at extracting local features from the input LR images (receptive field is limited by kernel size), while the Attention-based SR [4] is adept at non-local features from the input LR images but costly. Both kinds of methods achieved remarkable results on PSNR or SSIM, but the performance of these methods is varied with the input LR image's features.

Addressing the above shortage, we try to make the best of both Attention-based SR and Convolutional-based SR. We introduce the Attention-based block to focus on extracting the non-local features, while the Convolution-based block focuses on extracting the local features. This strategy can reinforce the advantages and avoid the disadvantages of both blocks.

However, simply connecting two different blocks in parallel cannot improve the performance: (1) Simply introducing two blocks will double the network scale, which

leads to difficult training. To solve this problem, we introduce the feedback strategy for our network. The Feedback-based SR can reduce the network scale while achieving remarkable performance. (2) How do we merge two blocks' feature maps together to generate the desired outputs? Chen's work [5] indicated attention favor low-frequency patterns at low level and high-frequency patterns at higher levels, which means different level features need different weight values to fuse. So simply adding the corresponding feature maps together (can be seen as fixed weighting add) cannot vary the weight values during different levels. Another famous method is simply one Convolutional layer fusion, but Convolutional layer fusion cannot deal with non-local features and will lead to one branch gradient vanishing if bad initialization for the Convolutional layer. (3) How to generate desired input feature maps for two blocks? The two branches-based feedback block is sensitive to the input feature maps. It is expected that the input features need to be easily separated to feed into two blocks, and the first iteration input for the feedback block should be in the same form as the following iteration inputs, making the feedback block focus on extracting similar features during all the iterations. The deeper input block can ease the challenge, but it will lead to a vanishing gradient problem.

Addressing the three challenges mentioned above, in this paper, we proposed the Dynamic fusion of Local and Non-local features-based Feedback Network (DLNFN) for Super-Resolution (SR). (1) We take the Feedback-based SR as the overall structure, and introduce the Cross-Scale Non-Local block (CSNL) from Mei's work [4] as the Attention-based block and the Feedback-block (FB) from Li's work [3] as the Convolution-based-block, and the two blocks placed in parallel, We also proposed the Delivery–Adjust–Fusion framework to hold these blocks, making them working on what they do best. (2) We introduce a Dynamic Weighting block (DW block) to evaluate the weight values for two branch feature maps under different inputs for each feedback iteration and sum the corresponding feature maps together. (3) We introduce a Mutual Affine Convolutional (MAConv) layer from Liang's work [6] as the input layer. The MAConv layer can enhance feature expressiveness without increasing receptive field, model size, and computation; therefore, one MAConv can get a close performance using fewer model sizes than one traditional Convolutional layer. By cascading some MAConv layers as the input layer, we can obtain a deeper input layer that is easy to train.

In summary, the main contributions of this paper are three-folds:

1. We introduce the Convolution-based block and Attention-based block as the base block of our Feedback-based SR; the Convolution-based block focuses on extracting local feature, and the Attention-based block focus on extracting non-local feature. We also propose the Delivery–Adjust–Fusion framework to hold these blocks, making them work on what they do best.
2. We proposed a Dynamic Weighting block (DW block) to generate the right weight values for different inputs under different iterations, and fuse both branches' feature maps together.
3. We introduce the MAConv layer as the input block, which is critical for our two branch-based feedback algorithms. By cascading 4 MAConv layers as the input layer, we can obtain a deeper input layer while easy to train.

The experiments show our proposal outperforms other state-of-the-art algorithms.

2. Related Works

In this section, we will give a brief introduction to the famous SR algorithms related to our work.

The Convolutional-based SR can extract local features efficiently. The receptive field is limited by kernel size (usually 3×3), but it can be extended by cascading many Convolutional layers. The SRCNN [7] is well-known as the first deep Convolutional-based SR, which is only 3 Convolutional layers. VDSR [8] is a very deep Convolutional-based SR with 20 Convolutional layers, so VDSR can extract deeper features which will improve the SR performance, but difficult to converge. SRCNN and VDSR are well-known classical

SISR algorithms, the performances are not superior but are still an inspiration for other algorithms.

The Feedback-based SR is a famous network structure that can go very deep without increasing the number of parameters. DRCN [9] is a typically Feedback-based SR, in which up to 16 recursions, the outputs of all recursions are fed into the end (Reconstruction Net) so different levels of feature maps can be used to generate SR images. SRFBN [3] is a newly proposed Feedback-based SR algorithm, the feedback block of SRFBN is with G projection groups sequentially with dense skip connections among them, totally T iterations. Each projection group includes a Deconvolutional layer that follows a Convolutional layer, the Deconvolutional layer can up-sampled features (LR - HR) while the Convolutional layer can down-sampled (HR - LR) to get rid of useless information, this strategy can generate powerful high-level representations. However, both DRCN and SRFBNs failed to consider the Non-local features.

There are many works on optimizing the Convolutional-based SR to reduce the cost. Adder Neural Networks (AdderNets) utilize additions to calculate the output features, thus, avoiding massive energy consumption of Conventional multiplications, but cannot be directly introduced into SR due to the different calculation paradigm. Song et. al. [10] thoroughly analyze the relationship between an adder operation and the identity mapping and insert shortcuts to enhance the performance, proposing the AdderSR. The experiments show AdderSR can achieve comparable performance to that of the CNN baselines with an about 2.5$\times$ reduction in energy consumption. Liang et. al. [6] proposed a new Mutual Affine Convolution (MAConv) layer which enhanced feature expressiveness without increasing receptive field, model size, and computation burden. The MAConv exploits channel interdependence, by splitting channels with an affine transformation module whose input is the rest of channel splits. The experiments show that the proposed MAConv achieves the best performance with fewer parameters and FLOPs. Both AdderSR and MAConv layers can reduce the cost, which is very helpful for us in designing larger-scale networks. However, we cannot simply replace all the Convolutional layers with them, as they were not designed to improve the performance (PSNR/SSIM).

Attention, which can extract non-local features, has been studied extensively in the previous research [11,12]. There are many simple but efficient Attention-based models, such as Channel Attention [11], Spatial Attention [13,14], and Pixel Attention [15]. These Attention-based models can extract long-range (non-local) features under different dimensions. To improve the SR model's performance further, many larger Attention-based models were introduced.

Mei et al. [4] proposed the Cross-Scale Non-Local (CSNL) attention module, which can deal with different scale non-local features. The CSNL is a cross-scale feature correlation-based module by introducing a down-sample scaling factor, so the CSNL can deal with different scale non-local features. Niu et al. [16] proposed a new Holistic Attention Network (HAN), which consists of a Layer Attention Module (LAM) and a Channel-Spatial Attention Module(CSAM) to model the holistic inter-dependencies among layers, channels, and positions. These attention models are powerful but costly due to the quadratic computational cost of the input size and introduced too much noise [17].

Introducing sparse representation into the Attention-based model can alleviate the problem of noise and cost. Mei et al. [18] introduce sparse representation into the Attention-based model, proposing a novel Non-Local Sparse Attention (NLSA) with dynamic sparse attention pattern while reducing the computational cost from quadratic to asymptotic linear with respect to the input size. Xia et al. [17] proposed a novel Efficient Non-Local Contrastive Attention (ENLCA) into SR, which also introduced sparse into the module. The ENLCA merely requires linear computation and space complexity with respect to the input size. Both NLSA and ENLCA significantly reduce the cost while achieving comparable performance as the state-of-the-art Attention-based module. However, both of these models did not consider the different scale similar features and only considered the non-local features.

Fusing different blocks together is a difficult task. Behjati et al. [19] introduced a novel procedure called residual attention feature group (RAFG), in which both parallelizing attention and residual block are linearly fused. They also propose a directional variance attention network (DiVANet), which is a computationally efficient yet accurate network for SISR. Our previous work [20] introduced a heavy-block LNFSR that introduced three different blocks; we also proposed an Up-Fusion-Delivery layer to fuse three blocks together. The proposed LNFSR achieved the best performance, but the network is too large, and the fusion method is crude and needs further optimization. Chen et al. [5] introduced an Attention Dropout Module, which dynamically produces sum-to-one attention for its internal branches. This strategy is ingenious and inspired our work, but the proposed Attention Dropout Module is lightweight, which only focuses on the non-local features, and it is not designed for the Feedback-based network, so the strategy cannot be introduced into our work directly.

Motivated by the recent work analyzed above, we proposed the Dynamic fusion of Local and Non-local features-based Feedback Network (DLNFN) for SR, expecting the Attention-based block (for non-local features) and the Convolutional-based block (for local features) can work on what they do best. The fusion method is inspired by Chen's work [5] but making great improvement for our work. We also introduce the MAConv layer, which was optimized to reduce the cost in Liang's work [6], to replace the Convolutional layers in the critical place.

3. The Dynamic Fusion of Local and Non-Local Features-Based Feedback Network (DLNFN) for SR

In this section, we introduce our proposed Dynamic fusion of Local and Non-local features-based Feedback Network (DLNFN) for SR. Firstly, we will introduce the network overall Architecture of our proposed DLNFN in Section 3.1. Secondly, we will introduce the main block (DLN block) of our proposed DLNFN in Section 3.2. At last, we will discuss the implementation details not mentioned above in Section 3.3. The acronyms and notations used in this Section are listed in Appendix A (Table A1).

3.1. The Network Overall Architecture of Our DLNFN

In this section, we will give a detailed introduction to the network architecture of our proposed DLNFN, the overall architecture is shown in Figure 1. Our DLNFN consists of 3 parts: Input Feature Extraction block (In block), Dynamic fusion of Local and Non-local features-based Feedback block (DLN block), and Reconstruction block (Rc block). Let us denote I_{LR} as the input Low-Resolution (LR) image, I_{HR} as the corresponding High-Resolution (HR) image, and I_{SR} as the Super-Resolution (SR) image, which is the output of the DLNFN.

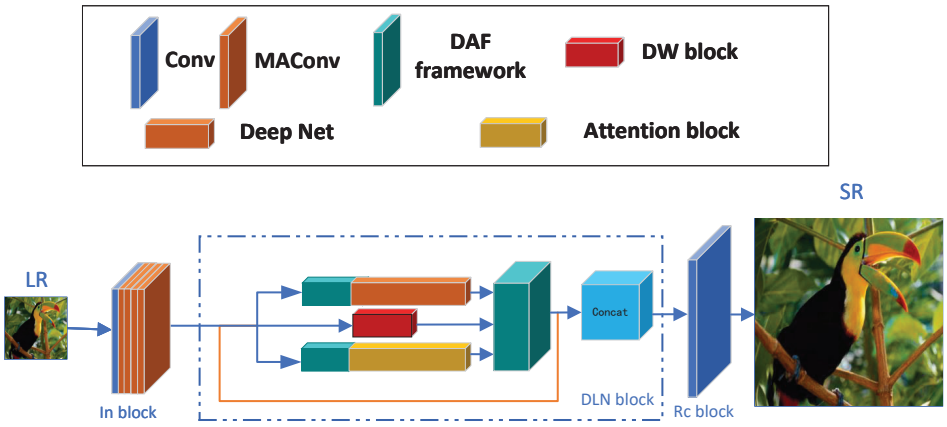

Figure 1. The network overall architecture of our LNFSR.

Input Feature Extraction block (In block): The In block is the first block extracting features from the input LR image. Due to the In block being far from the output, it is easy to be vanishing gradient problem (especially for Feedback-based SR, which can be considered as a very deep network if we expend Feedback block into a series of blocks). Precious work usually takes only one Convolutional layer to reduce the vanishing gradient problem [3,4], but this trick isn't suitable for our DLNFN because the DLN block is three blocks in parallel, which is sensitive to the feature maps from the In block. The first iteration's input (the In block outputs) must be similar to the following iteration's inputs (the outputs of the DLN block in different iteration). The suitable output for In block will release the DLN block from compensating different iteration's inputs, so powerful In block can help the DLN block focus on extracting similar features during different iterations.

To enhance the feature extraction capacity and expressiveness for the In block, we introduce the MAConv layer [6] into the In block of our DLNFN. The MAConv layer can enhance feature expressiveness without increasing receptive field, model size, and computation, so one MAConv can get close performance but fewer model size, as well as computation complexity, than the traditional Convolutional layer. By cascading some MAConv layers for our In block, we can obtain a deeper input layer while easy to train.

The In block cascades one Convolutional layer and 4 MAConv layers, with PReLU following each layer. The Convolutional layer is to expand the channel from 3 (for RGB image) of the input LR images to the desired channels (64 for our DLNFN), and the 4 MAConv layers extract useful feature maps for the following block (DLN block).

If we denote the In block as $In(\cdot)$ and the input of In block is the LR image (I_{LR}), the output of In block is shown in Equation (1):

$$F_{In} = In(I_{LR}) \tag{1}$$

The output of In block F_{In} are fed into the following block (DLN block).

Dynamic fusion of Local and Non-local features-based Feedback block (*DLN* block): The DLN block is the Feedback block for our DLNFN, which serves as the main block of our DLNFN. The structure is shown in Figure 1. During each iteration, the output of the DLN block is the up-scale feature map (the same dimensions as the HR image); this trick will reduce the load of the following block (Rc block). The *DLN* block is denoted as $DLN(\cdot)$, and the input of the *DLN* block is the output of In block (F_{In}), so the output of *DLN* block $DLN(\cdot)$ is shown in Equation (2):

$$F_{DLN} = DLN(F_{In}) \tag{2}$$

For the *DLN* block: (1) we introduce two base blocks to focus on extracting different kinds of features: we introduce the Cross-Scale Non-Local block (CSNL) from Mei's work [4], as the Attention-based block, to focus on extracting non-Local features. We also introduce the Feedback-block (FB) from Li's work [3], as the Convolutional-based block, to focus on extracting local features. Both blocks are placed in parallel, as Figure 1, making both blocks work on what they do best. (2) We introduce a Dynamic Weight block (DW block) to score two blocks' weight values for each iteration's input, so different inputs under different iterations can get different weight values to sum two blocks' feature maps together.

Figure 2 is the structure of our *DLN* block, we will give a detailed description of the *DLN* block in Section 3.2.

Reconstruction block (Rc block): The Rc block is simply one Convolutional layer, which reconstructs the SR image by assembling the up-scale feature maps of the DLN block's output. The Rc block is denoted as $Rc(\cdot)$ and the input of Rc block is the output of DLN block (F_{DLN}), so the output of Rc block is the final *SR* image (I_{SR}), denotes as in Equation (3):

$$I_{SR} = Rc(F_{DLN}) \tag{3}$$

The *Rc* block is only one Convolutional layer with its input as the up-scale feature maps. So: (1) the *Rc* block will release from up-scale feature maps, only focusing on assembling features into the *SR* outputs. (2) the gradient can be delivered to the previous blocks (In block and *DLN* block) quickly during the back-propagation process, which can reduce the vanishing gradient problem of the previous blocks.

The loss function: The DLNFN is optimized to minimize the loss function $L(\Theta)$; we choose the Mean Absolute Error (MAE, denoted as L_1) as the loss function to measure the difference between the output *SR* image and the ground-truth *HR* image. Given a training image pair $<I_{LR}, I_{HR}>$, where I_{HR} is the *HR* image and I_{LR} is the corresponding *LR* image, the loss function is defined as:

$$L(\Theta) = \|DLNFN(I_{LR}) - I_{HR}\|_1 \tag{4}$$

where:

$$DLNFN(I_{LR}) = I_{SR} = Rb(DLN(In(I_{LR}))) \tag{5}$$

3.2. The Dynamic Fusion of Local and Non-local Features-Based Feedback Block (DLN Block)

In this section, we will give a detailed introduction to the Feedback block of our proposed DLNFN: the Dynamic fusion of Local and Non-local features-based Feedback block (*DLN* block). The detailed structure of our proposed DLN block is described in Figure 2.

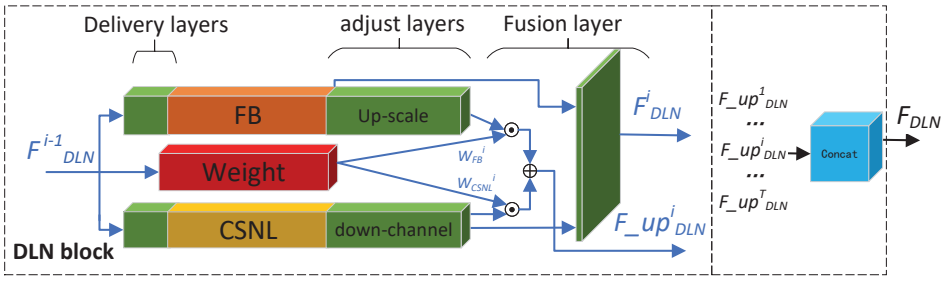

Figure 2. Detailed structure of the DLN block.

The Structure of DLN block: Figure 2 gives us a detailed illustration of our proposed DLN block. The DLN block is a Feedback-based block, with totally T iterations. For each iteration (such as the i-th iteration), the previous iteration's output (F_{DLN}^{i-1} if $2 \leqslant i \leqslant T$ or F_{In} if $i = 1$), with its dimensions as the LR, is considered as the input of the i-th iteration. The outputs of the i-th iteration are two fold: F_{DLN}^{i} and $F\_up_{DLN}^{i}$. The output of F_{DLN}^{i}, with its dimensions as the LR, is considered as the input of the next iteration. In addition, the output of $F\_up_{DLN}^{i}$, with its up-scaled feature maps as SR's dimensions, is considered as the output of the current iteration of the DLN block.

We concatenate the output of all iterations ($F\_up_{DLN}^{1}, \cdots, F\_up_{DLN}^{i}, \cdots, F\_up_{DLN}^{T}$) in channel dimension as the final output of the DLN blocks F_{DLN}. The final output of the DLN block is defined as

$$F_{DLN} = concat(F\_up_{DLN}^{1}, \cdots, F\_up_{DLN}^{T}) \tag{6}$$

where the function $concat(\dots)$ is to concatenate all the inputs on feature dimension.

We take all the feature maps of all iterations as the output to provide an abundant different hierarchy of feature maps for the Rc block to generate a more accurate SR image. Due to different iteration outputs, one can extract different hierarchies from the feature maps [21], and the SR image can be generated from different hierarchies of the feature maps.

The blocks in DLN: There are three blocks in our DLN: the FB block [3], the CSNL block [4] and the Dynamic Weight block (DW block): (1) the FB block focus on extracting the local feature and generate desired local features, which is more efficient and lower cost

than Attention-based block (the CSNL block). So we take the FB block to focus on the local feature high efficiently, releasing the load of the CSNL block. (2) the CSNL block is an Attention-based block that is powerful on both non-local and local features but costly. In our DLN block, the CSNL block will only need to focus on the non-local features, this will reduce the cost of the CSNL block. (3) The DW block is used to generate different weight values for two branches (FB branch and CSNL branch) during each iteration. The two branches' weight values varied with different inputs at different locations in the neural network [5], so the DW block is introduced to generate dynamic weight values to fuse the two branches' output together.

The Delivery–Adjust–Fusion framework (DAF framework): In Figure 2, the DAF framework (the green block and blue lines among them) is the framework for the DLN block, which is used to carry three blocks (the FB block and the CSNL block for feature extraction, and the DW block for weighting), giving two feature extraction blocks fully playing to their strengths. The DAF framework consists of the Delivery layer, Adjust layer, and Fusion layer.

The Delivery layer: the Delivery layer is simply one Convolutional layer, which is ahead of the FB and CSNL block. The Delivery layer is focused on extracting desired information for the following block from the input (F_{DLN}^{i-1} if $2 \leqslant i \leqslant T$ or F_{In} if $i = 1$): the Delivery layer ahead of the FB is focusing on extracting the local features, while the Delivery layer ahead of the CSNL is focusing on extracting the non-local features. The input are fed into the DW block without any preprocessing to score the local and non-local weights directly.

The Adjust layer: we introduce the Adjust layer to adjust the outputs of the FB block and CSNL block into the same channel and two different scales for the following layer (Fusion layer). The outputs are twofold: the output with its dimensions as the LR (denote as F_{FB}^{i} for FB block and F_{CSNL}^{i} for the CSNL block in the i-th iteration), and the output with its up-scale dimensions as the HR (denote as $F\_up_{FB}^{i}$ for FB block and $F\_up_{CSNL}^{i}$ for the CSNL block in the i-th iteration). The FB block's output is only the F_{FB}^{i}, so one Deconvolutional layer after the FB block is to up-scale the output of F_{FB}^{i} to $F\_up_{FB}^{i}$. While the CSNL block's outputs are F_{CSNL}^{i} and $F\_up_{CSNL}^{i}$ with $2\times$ channels, so one Convolutional layer with $1\times$ output channels after the CSNL block, is introduced to reduce the output channel from $2\times$ channels to $1\times$ channels. At last, the outputs (F_{FB}^{i}, F_{CSNL}^{i}, $F\_up_{FB}^{i}$ and $F\_up_{CSNL}^{i}$) are feed into the Fusion layer.

The Fusion layer: we introduce the Fusion layer to fuse two channel outputs together for different scale inputs. We fuse the F_{FB}^{i} and F_{CSNL}^{i} into F_{DLN}^{i}, and fuse the $F\_up_{FB}^{i}$ and $F\_up_{CSNL}^{i}$ into $F\_up_{DLN}^{i}$. We take different strategies for different scale's outputs: (1) The F_{FB}^{i} and F_{CSNL}^{i} are fed into one Convolutional layer to generate output F_{DLN}^{i}, then F_{DLN}^{i} are considered as the input of the DLN block for next iteration. (2) The $F\_up_{FB}^{i}$ and $F\_up_{CSNL}^{i}$ are added with the weight values generated by the DW block. The outputs of DW block denotes as W_{FB}^{i} and W_{CSNL}^{i} (W_{FB}^{i} is the weight values for FB block, and W_{CSNL}^{i} is the weight values for CSNL block), so the output $F\_up_{DLN}^{i}$ are computed as Equation (7):

$$F\_up_{DLN}^{i} = W_{FB}^{i} \cdot F\_up_{FB}^{i} + W_{CSNL}^{i} \cdot F\_up_{CSNL}^{i} \tag{7}$$

Finally, all the outputs of $F\_up_{DLN}^{i}$ ($F\_up_{DLN}^{1}$, $\cdots$, $F\_up_{DLN}^{T}$) are concatenated as the output of F_{DLN}.

The Dynamic Weighting method: there are three typical feature fusion methods, as shown in Figure 3. (1) Figure 3b is Convolutional-based fusion, the Convolutional layer can learn optimal parameters according the training data. This strategy, which is flexible without a priori knowledge, is a widely used fusion strategy, but there are two shortages: (a) The Convolutional layer only focuses on the local features (receptive field depends on the kernel size, usually 3×3). (b) Inappropriate initial values for the Convolutional layer will result in difficulty in training. In extreme situations, it will cause one branch gradient to vanish. (2) Figure 3c is adding two branches' outputs with fixed weight values (usually the same weight values as 0.5); this strategy can get rid of the one branch gradient

vanishing problem, and ingenious weight values will greatly improve the performance of the network, but there are two shortage: (a) Fixing the weight values heavily rely on experience and a priori knowledge, simply set same weight values as 0.5 may not suitable for all the network. (b) The fixed weight values are not fit for the Feedback-based feature fusion due to the weight values varied under different inputs and iterations.

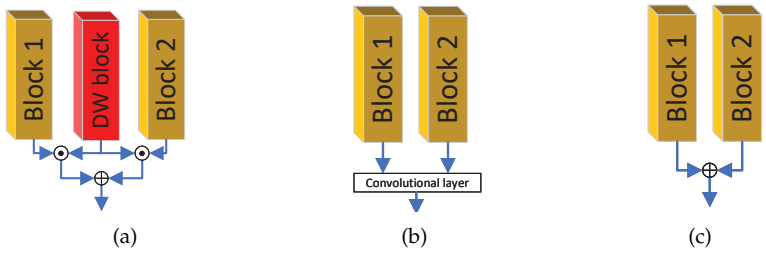

Figure 3. Three different types of feature fusion: (**a**) dynamic weighting add. (**b**) Convolutional fusion. (**c**) fix weights add.

Based on the above analyses, we introduce the Dynamic Weighting block (DW block) to generate proper weight values for the current iteration and fusion of two branches according to the weight values. The advantage of the dynamic weighting add-based fusion method, as in Figure 3a, are as follows: (1) If properly designed the DW block, the weight value generation can consider multiple factors, such as local and non-local features. (2) It can get rid of the one-branch gradient vanishing problem due to the add operation would not reduce the gradient propagation. (3) The DW block is suitable for the Feedback-based SR to generate different weight values during different iteration.

We only introduce the dynamic weighting add for $F\_up^i_{FB}$ and $F\_up^i_{CSNL}$ to generate the output $F\_up^i_{DLN}$, but take the Convolutional layer on F^i_{FB} and F^i_{CSNL} to generate output F^i_{DLN}. The reason is (1) it cannot share the same dynamic weight for generating $F\_up^i_{DLN}$ and F^i_{DLN} due to the need to consider different information for two fusion procedures: generating the weight values for $F\_up^i_{DLN}$ need to consider the input of F^{i-1}_{DLN}, while generating F^i_{DLN} for the next iteration's input need to predict the next iteration's two blocks' requirements. Therefore, we need two DW blocks to generate 2 groups of weight values respectively, this will increase the model parameters and computation cost. (2) Predicting the next iteration's requirement is heavy work, so we take one Convolutional layer to fuse the F^i_{FB} and F^i_{CSNL}, generating acceptable feature maps for the Delivery layer.

The analyses on the DW block are as follows: (1) The MAConv layer can extract the local features like the classic Convolutional layer with the same receptive field (kernel size $= 3 \times 3$) but fewer parameters and computation cost, so it's helpful to reduce the DW block's cost. (2) The PA+AvgPool+FC can extract information from the non-local features. The PA (Pixel Attention) block [15] generates attention coefficients for all pixels of the feature map, generating 3D output. Then the AvgPool + FC extracts the non-local features, which can be considered as a Channel Attention-Like block (slightly different from classic Channel Attention [13], we drop the max pooling for simple and modify the last FC's output channel = 2), generating 1D weight vector. (3) We place the MAConv (for the local feature) and the PA + AvgPool+FC (for the non-local feature) sequentially, so feature maps flow as 3D Local (convolution)-3D Non-local (pixel)-1D Non-local (channel)-output.

The Dynamic Weight block (DW block): the architecture of DW block is shown in Figure 4. The DW block is simply 4 layers to generate two weight values (W^i_{FB} and W^i_{CSNL}) for the two branches of up-scaled outputs (the $F\_up^i_{FB}$ and $F\_up^i_{CSNL}$) in the i-th iteration. Therefore, we introduce the MAConv+PReLU to evaluate the weight of local futures (FB block's output: $F\_up^i_{FB}$) and PA+AvgPool+FC to evaluate the weight of non-local features (CSNL block's output: $F\_up^i_{CSNL}$). The DW block is considered a lightweight block, so the DW block wouldn't increase the load of the DLN block.

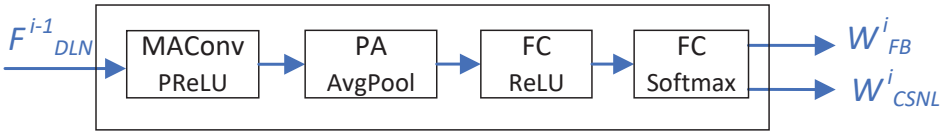

Figure 4. The architecture of Dynamic Weight block (DW block).

The DW block is inspired by A^2N of Chen's work [5], but there are some differences: (1) Our block is Feedback-based while A^2N is n blocks sequentially; (2) the DW block of our DLN block is considered both local and non-local feature, so DW block introduced both Convolutional-based block and Attention-based blocks (PA and CA-like), while the A^2N's Attention Dropout Module only generating the dynamic attention weights with CA-like block.

3.3. Other Implementation Details

Following are the other implementation details not mentioned above:

(1) We follow the implementation details of the FB and CSNL blocks. We take the activation function as PReLU for all the Convolutional and Deconvolutional blocks except for the DW block and the Rc block. We take the feature-map channels = 64 and feedback iteration = 9 for our DLNFN.

(2) We take MAE loss (L_1 loss) to optimize our proposed DLNFN, Adam optimizer to optimize the network parameters with $\beta 1 = 0.9$, $\beta 2 = 0.999$ and the initial learning rate = 0.0001, we reduce the learning rate by multiplying 0.5 for every 150 epochs, total 500 epochs. The network is implemented with the PyTorch framework.

4. Experimental Results

4.1. Datasets, Reuse Strategy, and Evaluation Metrics

We perform all the experiments on the DIV2k database for training (a total of 1000 images, where 800 images as the train set, 100 images as a valid set, and 100 images as the test set); we take all the train set (800 HR images) for training all models (our DLNFN and all the comparing algorithms). The image reuse strategy is performed as follows: Firstly, each HR image is randomly cropped into one small patch. Secondly, each HR patch is randomly rotated by 0°, 90°, 180°, and 270°, and horizontally flipped to augment the train images. Lastly, the BiCubic method is performed to generate the LR patch from the HR patch. During each epoch, all training images are performed 20 times (10 times in Section 4.2 to reduce the training cost) under the image reuse strategy. We set the input patch (LR patch) size= 48×48 for our proposed DLNFN to balance the performance and cost. For evaluation, all the SR results are first transformed into YCbCr space and evaluated by PSNR and SSIM [22] metrics on the Y channel only.

4.2. Ablation Study

In this Section, we will perform an ablation study on our DLNFN. To reduce the training cost, all the experiments on the ablation study are performed on a light weight network and half data augmenting: We set the feature channel = 32 and feedback iteration = 3 as a light weight DLNFN, denoted as DLNFN-L. During each epoch, all the training images are performed 10 times. For fair comparison, the comparing algorithms on ablation study are under the same strategy.

Do the Local and Non-local features improve performance: We perform an ablation study to determine whether the Local and Non-local feature-based Network will improve the performance: We only active the FB block, denotes as DLNFN-Local, to determine the performance on single local feature-based Network, and only active the CSNL block, denotes as DLNFN-Non-local, to determine the performance on single non-local feature-based Network. We also perform experiments on the classic FB and CSNLN algorithm to git rid of the framework's impact, we choose the SRFBN-L [3] as a light weight FB-based algorithm, and we set the feature channel = 64, feedback iteration = 3 for the CSNLN,

denotes as CSNLN-L, as a light weight CSNL-based algorithm, the performances (PSNR) are list in Table 1.

In Table 1, our lightweight DLNFN performs best, demonstrating the efficiency of the Local and Non-local feature-based Network. Our DLNFN-based structure can improve the performance of FB-based block (outperform the SRFBN-L) but reduce the performance of CSNL-based block(underperform the CSNLN-L), illustrating the overall framework do impact the performance greatly, and different blocks have their own optimal overall framework.

Table 1. The ablation study on the Local and Non-local features at ×2 scale on Set5 (the best performance is shown in **red**).

Algorithm	Local-Feature-Based		Non-Local-Feature-Based		Local and Non-Local Features-Based
	SRFBN-L	DLNFN-Local	CSNLN-L	DLNFN-Non-local	DLNFN-L (our)
PSNR	37.77	37.84	37.91	37.85	38.02

Whether the Dynamic fusion method improves performance: We perform an ablation study to determine whether the Dynamic fusion method will improve the performance. Our proposed Dynamic fusion method, as Figure 3a, is computed as Equation (7). We replace the Dynamic fusion into one Convolutional layer as the Convolutional-based method (as Figure 3b, denote as Conv-DLNFN-L), and We fix both the weight values = 0.5 as the fix weights add method (as Figure 3c, denote as Fix-DLNFN-L). The performances (PSNR) are listed in Table 2.

Table 2. The ablation study on the Dynamic fusion at ×2 scale on Set5 (the best performance is shown in **red**).

Algorithm	Conv-DLNFN-L	Fix-DLNFN-L	DLNFN-L (our)
PSNR	37.90	37.75	38.02

In Table 2, our Dynamic fusion-based DLNFN performs best. The performance of the Dynamic fusion method (DLNFN-L) outperforms the Convolutional-based method (Conv-DLNFN), and the fix weights add method (Fix-DLNFN) with a large gap. The fix weights add method performed worst, illustrating the unsuitable weighting values will perform the opposite effect.

Whether the MAConv layer improves performance: We perform an ablation study to determine whether the MAConv layer improves performance. Our DLNFN-L is 4 MAConv layers for the In block, we choose 3 compared algorithm as follow: 2 MAConv layers for the In block (denoted as DLNFN-2MA), 2 Convolutional layers for the In block (denotes as DLNFN-2Conv), 4 Convolutional layers for the In block (denotes as DLNFN-4Conv). The performances (PSNR/SSIM) are listed in Table 3.

Table 3. The ablation study on the MAConv layer at ×2 scale on Set5 (the best performance is shown in **red**).

Algorithm	convolutional Layer Based		MAConv Layer Based	
	DLNFN-2Conv	DLNFN-4Conv	DLNFN-2MA	DLNFN-L (4MA, our)
PSNR	37.96	37.97	38.00	38.02

In Table 3, the MAConv layer-based algorithms (2 MAConv layers and 4 MAConv layers) outperform the corresponding Convolutional layer based algorithms (2 Convolutional layers and 4 Convolutional layers) for the In block, which means the MAConv layer can improve the performance. The 4 MAConv layers perform best, so we take the In block as 4 MAConv layers.

4.3. Comparisons With State-of-the-Arts

In this section, we will give a quantitative comparison of our DLNFN with other famous state-of-the-art SR algorithms, we choose SRCNN [7], VDSR [8], EDSR [23], RCAN [24], SRFBN [3], A^{2}N [5], DiVANet+ [19], and CSNLN [4] as the state-of-the-art algorithms considered in this experiment. The SRCNN, first proposed as the Convolutional based SR algorithm, is considered the baseline of the SR method. We have downloaded the official PyTorch-based models for RCAN, SRFBN, and CSNLN algorithms, so we performed the official model in this section. We retrained the SRCNN, VDSR, and EDSR algorithms. We dropped their unique training tricks (such as noise and Gauss blurring) and retrained them under the same training strategy as our DLNFN, a total of 1000 epochs. We perform the upscale factor range in $\times 2$ and $\times 3$ for all the SR algorithms; we did not perform our DLNFN on a larger scale (such as $\times 4$ and $\times 8$) due to time and GPU memory consumption. We report the performance on five famous standard benchmark databases: Set5 [25], Set14 [26], B100 [27], Urban100 [28], and Manga109 [29]. For a fair comparison, we also list the number of parameters to show the effectiveness of algorithms. We evaluated all the SR results on PSNR and SSIM [22]. The results are listed in Table 4.

Table 4. The performance (PSNR/SSIM) of the considered state-of-the-art algorithms (the best performance is shown in **red** and the second-best performance is shown in **blue**).

Algorithm	Scale	Params	Set5		Set14		B100		Urban100		Manga109	
			PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
SRCNN [7]	×2	0.02M	36.45	0.9527	32.32	0.9043	31.04	0.8838	29.11	0.8887	34.82	0.9627
VDSR [8]	×2	0.67M	37.58	0.9587	33.16	0.9133	31.94	0.8964	31.05	0.9164	37.44	0.9737
A^{2}N [5]	×2	1.04M	38.06	0.9608	33.75	0.9194	32.22	0.9002	32.43	0.9311	38.87	0.9769
SRFBN [3]	×2	2.14M	38.11	0.9609	33.82	0.9196	33.29	0.9010	32.62	0.9328	39.08	0.9779
DiVANet+ [19]	×2	0.9M	38.23	0.9618	33.88	0.9201	32.36	0.9018	32.67	0.9330	39.15	0.9780
CSNLN [4]	×2	3.06M	**38.28**	**0.9616**	**34.12**	**0.9223**	32.40	0.9024	**33.25**	**0.9386**	**39.37**	0.9785
RCAN [24]	×2	15.44M	38.27	0.9614	**34.12**	0.9216	**32.41**	**0.9027**	33.34	0.9384	39.44	0.9786
EDSR [23]	×2	40.73M	38.25	0.9613	33.97	0.9205	32.36	0.9020	32.98	0.9361	39.17	0.9781
DLNFN (our)	×2	5.82M	**38.28**	**0.9617**	**34.22**	**0.9231**	**32.41**	**0.9026**	**33.39**	**0.9390**	**39.41**	**0.9786**
SRCNN [7]	×3	0.02M	32.52	0.9052	29.09	0.8160	28.10	0.7781	25.84	0.7869	29.62	0.8999
VDSR [8]	×3	0.67M	33.76	0.9225	29.96	0.8347	28.85	0.7986	27.32	0.8324	32.41	0.9356
A^{2}N [5]	×3	1.04M	34.47	0.9279	30.44	0.8437	29.14	0.8059	28.41	0.8570	33.78	0.9458
SRFBN [3]	×3	2.83M	34.70	0.9292	30.51	0.8461	29.24	0.8084	28.73	0.8641	34.18	0.9481
DiVANet+ [19]	×3	0.9M	34.66	0.9289	30.53	0.8452	29.26	0.8077	28.66	0.8610	34.02	0.9473
CSNLN [4]	×3	6.01M	**34.74**	**0.9300**	**30.66**	**0.8482**	**29.33**	0.8105	**29.13**	**0.8712**	**34.45**	**0.9502**
RCAN [24]	×3	15.63M	**34.74**	0.9299	30.65	0.8482	29.32	0.8111	29.09	0.8702	34.44	0.9499
EDSR [23]	×3	43.68M	**34.74**	0.9297	30.50	0.8461	29.24	0.8095	28.76	0.8651	34.01	0.9481
DLNFN (our)	×3	9.59M	**34.84**	**0.9307**	**30.78**	**0.8498**	**29.36**	**0.8119**	**29.45**	**0.8764**	**34.72**	**0.9515**

We list the performance of our DLNFN and other compared algorithms at a scale factor of $\times 2$ and $\times 3$. Our proposal performed best in all the state-of-the-art algorithms, illustrating the Local and Non-local features-based network outperform other single feature-based networks. In the $\times 2$ scale, our DLNFN outperformed most of the other compared algorithms. Our DLNFN algorithm performs best in 3 databases (Set14, B100, and Manga109). In Set5, our DLNFN draw with the CSNLN in 'PSNR' metric, is better than CSNLN in the 'SSIM' metric. In B100, our DLNFN draws with the RCAN in the 'PSNR' metric, but worse than RCAN in the 'SSIM' metric. In $\times 3$ scales, our DLNFN outperformed best than all the other compared algorithms, the reason for this is that our DLN block can generate the upscaled feature maps directly, reducing the load on the Rc block to generate SR images.

We take the newly proposed evaluation metric: Diffusion Index (DI) [30] (A larger DI indicates more pixels are involved), evaluate the resection field for our DLNFN and CSNLN (second-best performance) in scale $\times 2$, the $DI = 19.96$ for CSNLN (with 12 iterations) while the $DI = 20.42$ for our DLNFN (with 9 iterations), illustrated our DLNFN involved more pixels than CSNLN, but fewer feedback iterations. The reason we suspect this is that our Delivery–Adjust–Fusion framework can rapidly deliver features to the right place, so the larger resection field is fully evaluated.

We must acknowledge that there are still shortcomings in our proposed algorithm: Our DLNFN is relatively more parameter numbers than some comparing algorithms, such as DiVANet+ [19], which is balance-oriented (balance the cost and performance), while our proposed DLNFN is performance-oriented (higher PSNR and SSIM).

4.4. Visualized Analysis

In this section, we give a visualized analysis of our DLNFN. We chose the SRCNN [7], VDSR [8], EDSR [23], RCAN [24], SRFBN [3], A^{2}N [5], and CSNLN [4] as the comparing algorithms, the HR and LR image is considered as the benchmark and our DLNFN is placed in the right-bottom. We chose two typical sections under $\times 2$ and $\times 3$, in Figure 5.

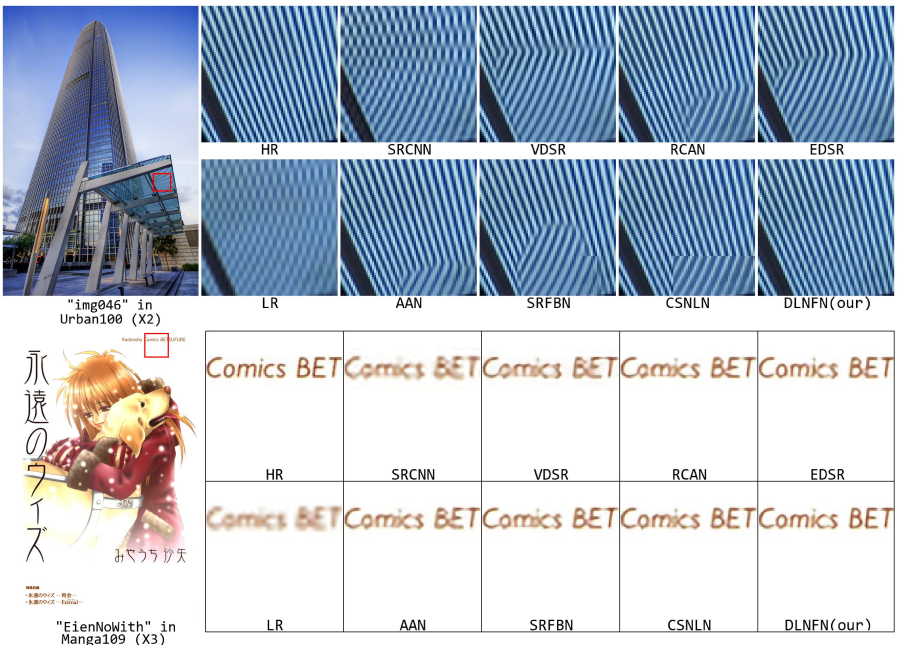

Figure 5. Visualized comparison on our DLNFN with other comparing Algorithms.

The first picture in Figure 5 is 'img046' in the Urban100 database under scale $\times 2$. The glass has a line-like shape, which has typically long-distance similar features. The direction of the line in HR is from the top left to the bottom right, and only our DLNFN generated the right SR image. The A^{2}N performed second. The SRCNN generates the wrong image. In addition, the other compared algorithms generate the wrong line direction.

The Convolutional-based algorithms (SRCNN, VDSR, EDSR, and SRFBN) performed worse (larger area wrong SR image patch) than the Attention-based algorithms(RCAN and CSNLN), illustrating the Attention-based algorithms can take fully used of the non-local feature (long distance simply features), which can help to estimate local feature's pattern.

The second picture in Figure 5 is the 'EienNoWith' image in the Manga109 database under scale $\times 3$. The word 'Comics BET' is too small in $\times 3$ LR image, so all the algorithms fail to generate clearly letters. However, our DLNFN can generate recognizable letters ('Comics BET'). For the comparing algorithms, the SRCNN, VDSR, EDSR, and SRFBN failed to generate clear letters 'i', 'c, 'and 'E,' the RCAN failed to generate clear letters 'i' and 'c,' the CSNLN generate second best SR patch but still worse than our DLNFN in letter 'i' and 'E.' The English letter is typically for local features, but Attention-based algorithms (RCAN CSNLN) still outperformed the Convolutional-based algorithms (SRCNN, VDSR, EDSR, and SRFBN), illustrating the Attention-based algorithm's power, but it was not superior to Convolutional-based algorithms in all aspects.

According to the visualized analysis above, the Attention-based algorithm (RCAN and CSNLN) is more powerful than Convolutional-based algorithms but is not superior to Convolutional-based algorithms in all aspects, so Convolutional-based algorithms can compensate the Attention-based algorithms, making all block working on what they do best. By introducing the FB block (focus on local feature) and CSNL block (focus on non-local feature), our proposed DLNFN can generate clearer SR images and outperform other state-of-the-art algorithms.

5. Conclusions

In this paper, we proposed a Dynamic fusion of Local and Non-local features-based Feedback Networks (DLNFN) for Super-Resolution (SR). The main contributions of this paper are three-fold: (1) We introduce the Convolution-based block to focus on extracting local features and the Attention-based block to focus on extracting non-local features. We also propose the Delivery–Adjust–Fusion framework to hold these blocks, making them work on what they do best. (2) We proposed a Dynamic Weighting block to generate weight values for different inputs under different iterations, and fuse both branches' feature maps together. (3) We introduce the MAConv layer as the input block, which is critical for our two branch-based feedback algorithms. By cascading 4 MAConv layers as the input layer, we can obtain a deeper input layer while easy to train. Experiments show our proposed DLNFN can take full advantage of two different blocks, and outperform other state-of-the-art algorithms. However, our proposed DLNFN only considered the fuse method for two blocks' output, but fail to consider the split method for blocks' input. So our future work is trying to introduce a split method, such as the Clustering method [31], to generate suitable inputs for two blocks, further improving the performance.

Author Contributions: Conceptualization, Y.L.; funding acquisition, Z.C.; methodology, Y.L.; writing—original draft, Y.L.; writing—review & editing, Z.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China, under Grant No. U2006228 and No. 61972241. This work was supported by Shanghai Soft Science Research Project No. 23692106700. This work was supported by the Natural Science Foundation of Shanghai under Grant No. 22ZR1427100.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

For better reading this paper, we provide part of the acronyms and notations used in this paper in Table A1.

Table A1. The Acronyms and Notations.

Acronyms and Notations	Description
SISR	Single Image Super-Resolution
SR	Super-Resolution
HR	High-Resolution
LR	Low-Resolution
I_{SR}	Super-Resolution image
I_{LR}	Low-Resolution image
I_{HR}	High-Resolution image
$L(\Theta)$	the loss function
$In(\cdot)$	the In block
F_{In}	the output features of the In block
$DLN(\cdot)$	the DLN block
F_{DLN}	the output features of the DLN block
$Rc(\cdot)$	the Reconstruction block
$DLNFN(\cdot)$	our proposed DLNFN algorithm
F^i_{DLN}	the i-th iteration output of the DLN block with dimension as LR
$F\_up^i_{DLN}$	the i-th iteration output of the DLN block with dimension as SR
F^i_{FB}	the output of FB branch with dimension as LR in the i-th iteration
F^i_{CSNL}	the output of CSNL branch with dimension as LR in the i-th iteration
$F\_up^i_{FB}$	the output of FB branch with dimension as SR in the i-th iteration
$F\_up^i_{CSNL}$	the output of CSNL branch with dimension as SR in the i-th iteration
W^i_{FB}	the weight value of FB branch in the i-th iteration
W^i_{CSNL}	the weight value of CSNL branch in the i-th iteration
$concat(\dots)$	concatenate all the inputs on feature dimension

References

1. Wang, Z.; Chen, J.; Hoi, S.C. Deep Learning for Image Super-resolution: A Survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *43*, 3365–3387. [CrossRef] [PubMed]
2. Mao, X.; Shen, C.; Yang, Y. Image Restoration Using Very Deep Convolutional Encoder-Decoder Networks with Symmetric Skip Connections. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Barcelona, Spain, 5–10 December 2016; Curran Associates, Inc.: New York, NY, USA; pp. 2802–2810.
3. Li, Z.; Yang, J.; Liu, Z.; Yang, X.; Jeon, G.; Wu, W. Feedback Network for Image Super-Resolution. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 3867–3876. IEEE: Piscataway, NJ, USA. [CrossRef]
4. Mei, Y.; Fan, Y.; Zhou, Y.; Huang, L.; Huang, T.S.; Shi, H. Image Super-Resolution with Cross-Scale Non-Local Attention and Exhaustive Self-Exemplars Mining. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 5690–5699. [CrossRef]
5. Chen, H.; Gu, J.; Zhang, Z. Attention in attention network for image super-resolution. *arXiv* **2021**, arXiv:2104.09497.
6. Liang, J.; Sun, G.; Zhang, K.; Van Gool, L.; Timofte, R. Mutual affine network for spatially variant kernel estimation in blind image super-resolution. In Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR), Nashville, TN, USA, 21–24 June 2021; pp. 4096–4105.
7. Dong, C.; Loy, C.C.; He, K.; Tang, X. Learning a Deep Convolutional Network for Image Super-Resolution. In Proceedings of the Computer Vision—ECCV 2014, Zurich, Switzerland, 6–12 September 2014; Springer International Publishing: Cham, Switzerland; pp. 184–199. [CrossRef]
8. Kim, J.; Lee, J.K.; Lee, K.M. Accurate Image Super-Resolution Using Very Deep Convolutional Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1646–1654. [CrossRef]
9. Kim, J.; Lee, J.K.; Lee, K.M. Deeply-Recursive Convolutional Network for Image Super-Resolution. In Proceedings of the IEEE Comput Soc Conf Comput Vision Pattern Recognit (CVPR), Las Vegas, NV, USA, 27–30 June 2016; IEEE: Piscataway, NJ, USA; pp. 1637–1645. [CrossRef]
10. Song, D.; Wang, Y.; Chen, H.; Xu, C.; Xu, C.; Tao, D. Addersr: Towards energy efficient image super-resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 15648–15657.

11. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
12. Li, B.; Xiong, S.; Xu, H. Channel Pruning Base on Joint Reconstruction Error for Neural Network. *Symmetry* **2022**, *14*, 1372. [CrossRef]
13. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In Proceedings of the European conference on computer vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 3–19. [CrossRef]
14. Li, B.; Lu, Y.; Pang, W.; Xu, H. Image Colorization using CycleGAN with semantic and spatial rationality. *Multimed. Tools Appl.* **2023**, *82*, 1–15. [CrossRef]
15. Zhao, H.; Kong, X.; He, J.; Qiao, Y.; Dong, C. Efficient image super-resolution using pixel attention. In Proceedings of the Computer Vision–ECCV 2020 Workshops, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 56–72.
16. Niu, B.; Wen, W.; Ren, W.; Zhang, X.; Yang, L.; Wang, S.; Zhang, K.; Cao, X.; Shen, H. Single image super-resolution via a holistic attention network. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 191–207.
17. Xia, B.; Hang, Y.; Tian, Y.; Yang, W.; Liao, Q.; Zhou, J. Efficient non-local contrastive attention for image super-resolution. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual Online, 22 February–1 March 2022; Volume 36, pp. 2759–2767.
18. Mei, Y.; Fan, Y.; Zhou, Y. Image super-resolution with non-local sparse attention. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 3517–3526.
19. Behjati, P.; Rodriguez, P.; Fernández, C.; Hupont, I.; Mehri, A.; Gonzàlez, J. Single image super-resolution based on directional variance attention network. *Pattern Recognit.* **2023**, *133*, 108997. [CrossRef]
20. Liu, Y.; Chu, Z.; Li, B. A Local and Non-Local Features Based Feedback Network on Super-Resolution. *Sensors* **2022**, *22*, 9604. [CrossRef] [PubMed]
21. Liu, J.; Zhang, W.; Tang, Y.; Tang, J.; Wu, G. Residual Feature Aggregation Network for Image Super-Resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 2359–2368. [CrossRef]
22. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef] [PubMed]
23. Lim, B.; Son, S.; Kim, H.; Nah, S.; Mu Lee, K. Enhanced Deep Residual Networks for Single Image Super-Resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 136–144. [CrossRef]
24. Zhang, Y.; Li, K.; Li, K.; Wang, L.; Zhong, B.; Fu, Y. Image Super-Resolution Using Very Deep Residual Channel Attention Networks. In Proceedings of the European Conference on Computer Vision (ECCV), Rupnagar, India, 26–28 November 2018; pp. 286–301. [CrossRef]
25. Bevilacqua, M.; Roumy, A.; Guillemot, C.; Alberi-Morel, M.L. Low-Complexity Single-Image Super-Resolution based on Nonnegative Neighbor Embedding. In Proceedings of the 23rd British Machine Vision Conference (BMVC), Surrey, UK, 3–7 September 2012. [CrossRef]
26. Zeyde, R.; Elad, M.; Protter, M. On Single Image Scale-Up using Sparse-Representation. In Proceedings of the International Conference on Curves and Surfaces, Avignon, France, 24–30 June 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 711–730. [CrossRef]
27. Martin, D.; Fowlkes, C.; Tal, D.; Malik, J. A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. In Proceedings of the Eighth IEEE International Conference on Computer Vision. ICCV 2001, Vancouver, BC, Canada, 7–14 July 2001; Volume 2, pp. 416–423. [CrossRef]
28. Huang, J.B.; Singh, A.; Ahuja, N. Single Image Super-resolution from Transformed Self-Exemplars. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5197–5206. [CrossRef]
29. Matsui, Y.; Ito, K.; Aramaki, Y.; Fujimoto, A.; Ogawa, T.; Yamasaki, T.; Aizawa, K. Sketch-based manga retrieval using manga109 dataset. *Multimed. Tools Appl.* **2017**, *76*, 21811–21838. [CrossRef]
30. Gu, J.; Dong, C. Interpreting Super-Resolution Networks with Local Attribution Maps. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 9199–9208. [CrossRef]
31. Xiong, S.; Li, B.; Zhu, S. DCGNN: A single-stage 3D object detection network based on density clustering and graph neural network. *Complex Intell. Syst.* **2022**, 9, 1–10. [CrossRef]