Lightweight Feature Fusion Network for Single Image Super-Resolution

Wenming Yang, Wei Wang, Xuechen Zhang, Shuifa Sun, Qingmin Liao

Abstract—Single image super-resolution(SISR) has witnessed great progress as convolutional neural network(CNN) gets deeper and wider. However, enormous parameters hinder its application to real world problems. In this letter, We propose a lightweight feature fusion network (LFFN) that can fully explore multi-scale contextual information and greatly reduce network parameters while maximizing SISR results. LFFN is built on spindle blocks and a softmax feature fusion module (SFFM). Specifically, a spindle block is composed of a dimension extension unit, a feature exploration unit and a feature refinement unit. The dimension extension layer expands low dimension to high dimension and implicitly learns the feature maps which is suitable for the next unit. The feature exploration unit performs linear and nonlinear feature exploration aimed at different feature maps. The feature refinement layer is used to fuse and refine features. SFFM fuses the features from different modules in a self-adaptive learning manner with softmax function, making full use of hierarchical information with a small amount of parameter cost. Both qualitative and quantitative experiments on benchmark datasets show that LFFN achieves favorable performance against state-of-the-art methods with similar parameters. Code is avaliable at https://github.com/qibao77/LFFN-master.

Index Terms—Super-resolution, convolutional neural network, softmax feature fusion module, spindle block.

I. Introduction

SINGLE image super-resolution (SISR) is an important low-level computer vision task which aims at recovering a high-resolution (HR) image from a low-resolution (LR) image. It is a seriously ill-posed problem since an LR image can be mapped to an infinite number of HR images. Recently, deep convolutional neural network (CNN) has greatly facilitated improvements in this field. Dong *et al.* [1] firstly proposed a three-layer CNN to establish a mapping between LR and HR. Kim *et al.* proposed the well-known VDSR [2], which introduced residual learning and adaptive gradient clipping to alleviate the difficulty of training deep network. In DRCN [3], the recursive network was used to reduce the model parameters and a multi-supervised strategy was adopted to fuse intermediate results. Benefiting from skip-connection can alleviate the vanishing-gradient problem [4], [5], Lim *et al.* [6] built a very deep network MDSR (more than 160 layers) with residual blocks.

Researchers usually deepen and widen the network to achieve better performance. However, even constructed with

W. Yang, W. Wang, X. Zhang and Q. Liao are with the Department of Electronic Engineering, Graduate School at Shenzhen, Tsinghua University, China (E-mail: {yang.wenming@sz, wangwei17@mails, xc-zhang16@mails, liaoqm@}.tsinghua.edu.cn).

S. Sun is with the Department of Hubei Key Laboratory of Intelligent Vision Based Monitoring for Hydroelectric Engineering, China Three Gorges University, China (E-mail: watersun@ctgu.edu.cn).
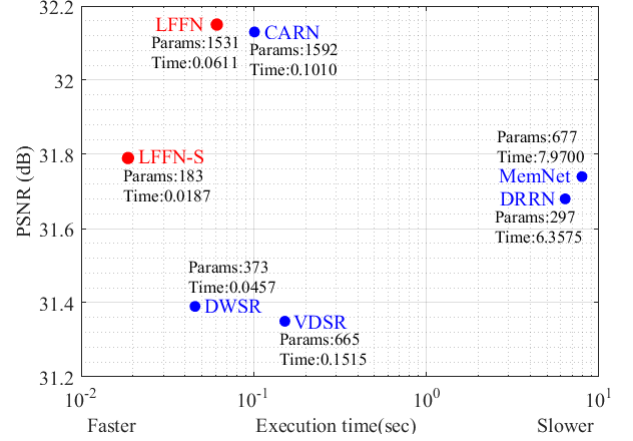

Fig. 1. Speed and accuracy trade-off. The average PSNR and the average inference time for upscaling $\times 4$ on Set5. LFFN and LFFN-S contain B4M15 and B4M4, respectively. The 3*3 convolution of all spindle blocks in LFFN-S is replaced by depthwise convolution.

small convolution kernels, such as 3×3, the network will take up large memories. In order to lighten the deep network, some strategies have been adopted. DRRN [7] employed parameter sharing strategy to reduce parameters, but it still requires large computation objectively. CARN-M [8] adopt group convolution to attack a trade-off between computation and performance of the model. Unfortunately, applying group convolution directly to SISR will obviously impair performance. To address these problems, we propose a lightweight network LFFN to compute the HR image from the original LR image. In LFFN, we introduce a new organization of the inception-residual block [9], named spindle block, which contains a dimension extension unit, a feature exploration unit and a feature refinement unit. The dimension extension unit can learn the feature maps suitable for the next unit, and the architecture can be mitigated by fewer filters in backbone. Inspired by ResNeXt [10] and Xception [11], we introduce a feature exploration unit to explore the linear and nonlinear as well as multi-scale information for 4 different channel groups. This unit can improve the representational power of the model and can further alleviate the architecture due to fewer filters in each group. We also consider using feature maps of different receptive fields to enhance the performance. Taking computation into account and motivated by feature recalibration demonstrated in SENets [12], we develop a softmax feature fusion module (SFFM) to aggregate the features of different levels in a self-adaptive channel-wise convex weighted way rather than the multi-supervised method used in DRCN [3] and

MemNet [13]. The parameters of SFFM are not large, since there is only one dense layer applied to each global feature of different levels. And SFFM can learn how to combine the features that are most conducive to reconstruction.
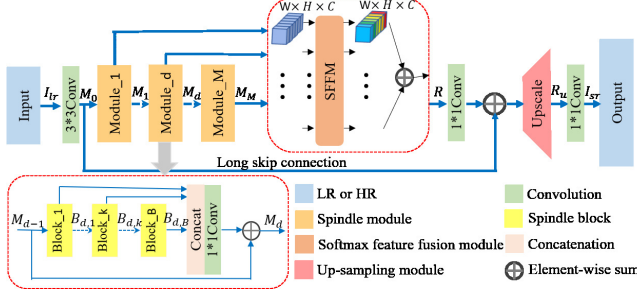
II. PROPOSED METHOD

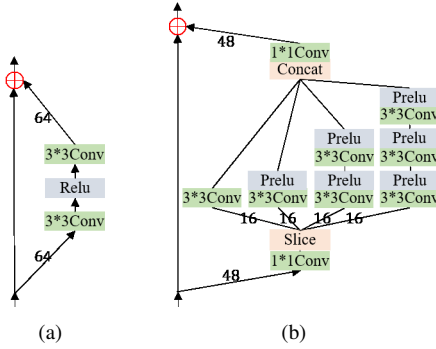Fig. 2. The architecture of our proposed lightweight feature fusion network (LFFN).

Fig. 3. Basic block of different architecture. (a) residual block. (b) spindle block.

A. Network structure

As shown in Fig. 2, the overall architecture consists of $B \times M$ spindle blocks, a softmax feature fusion module (SFFM) and an up-sampling module. We denote I_{lr} and I_{sr} as the input and output of LFFN, respectively. First of all, we use a single 3×3 convolutional layer of 48 filters to extract the feature maps from the original LR image:

$$M_0 = F_{sf}(I_{lr}), \qquad (1)$$

where $F_{sf}(\cdot)$ represents convolution operation and M_0 serves as the input of next part. The next part is M stacked local feature fusion modules. Inspired by MemNet [13] and SR-DenseNet [14], we concatenate feature maps from B stacked spindle blocks to further make full use of local features. We also introduce residual learning for each module to make deep network training easier. This procedure can be expressed as

$$
\begin{aligned}
M_d &= F_d(M_{d-1}) \\
&= F_{df}([B_{d,1}, \cdots, B_{d,k}, \cdots, B_{d,B}]) + M_{d-1},
\end{aligned} \qquad (2)
$$

where F_d denotes the d-th module function and F_{df} is the function of the 1×1 convolution in d-th module. M_d and

$B_{d,k}$ indicate the output of the d-th module and k-th spindle block respectively. More details about spindle block will be given in next section. After extracting complicated features progressively with M modules, we further conduct softmax feature fusion (SFFM).

$$R = F_{sffm}(M_d, \cdots, M_d, \cdots, M_d), \qquad (3)$$

where R is the output feature maps of SFFM, F_{sffm} denotes a composite function. Finally, like [15] and [6] , we utilize ESPCN [16] followed by a convolution layer to upscale the refined feature maps and get the output of LFFN. It is worth mentioning that we replace the 3×3 convolution with 1×1 convolution in upscale module and the last layer to further reduce parameters.

$$I_{sr} = F_{last}(F_{up}(F_{fuse}(R) + M_0)), \qquad (4)$$

where F_{last} and F_{fuse} denote the 1×1 convolution and F_{up} is the function of upscale module.

B. Spindle Block

To reap the benefits of inception residual block [9] and group convolution, we propose a well-designed residual block, named spindle block. The overall block can be formulated as:

$$B_{d,k} = B_{d,k-1} + F_{fru}(F_{feu}(F_{deu}(B_{d,k-1})), \qquad (5)$$

where F_{fru}, F_{feu} and F_{deu} represent compound function of three basic units respectively. And more details about them explicated as follows.

1) dimension extension unit: The number of filters is a critical factor to improve the efficiency of deep networks, which is fixed to 64 in many deep methods for SISR currently. We can lighten the architecture by decreasing the filters, but the performance fluctuates accordingly. Using "bottleneck layer" [17] (1×1 convolution) to compress dimensions resemble pooling operation in channel dimension. We believe that reducing feature channels before non-linear layer can lead to information loss. Here, we expand the dimensions from 48 to 64 before non-linear mapping to maintain performance with fewer parameters.

2) feature exploration unit: As shown in Fig.3(b), we first slice the feature maps into four different 16-dimensional groups. Then, we explore nonlinear information in three groups and linear information in the other group. Specifically, we adopt a sequence of 3×3 convolutional layers followed by parametric rectified linear units (PReLUs) to make full use of the image multi-scale information. Different from [17], [9], we assemble linear and nonlinear information to boost representational power of basic blocks and directly dispose the expanded feature maps instead of reducing dimension by additional 1×1 convolutions.

3) feature refinement unit: Then the concatenate feature maps are sent to a 1×1 convolutional layer which acts as refining features, compressing dimensions and overcoming the impact of the slice operation on weakening the information flow.

Basically, as shown in Fig.3, our spindle block can take advantage of linear and nonlinear and multi-scale information

with fewer parameters than baseline residual block. In particular, when we use the configuration expressed in Fig.3, a spindle block has 30.21% of parameters of a residual block. This ratio can be further decreased to 12.13% by replacing 3×3 convolution in spindle block with depthwise convolution. More analysis will be described in experiment.

C. Softmax Feature Fusion Module

Information in different levels of feature maps can complement each other for reconstruction. In order to gain more abundant and efficient information, we focus on hierarchical features and achieve a fusion mechanism. As shown in Fig.4, we take all intermediate feature maps M_i as input and generate a fusion representation R. And $M_i = [m_{i1}, ..., m_{ij}, ..., m_{iC}]$, $i = 1, ..., M$, $j = 1, ..., C$, where $m_{ij} \in \mathbb{R}^{W \times H}$ denotes the j-th channel of the i-th feature maps M_i, and C is the total number of channels. Inspired by squeeze operation in [12], we apply global average pooling to each channel to obtain the global channel feature $X_i = [x_{i1}, ..., x_{ij}, ..., x_{iC}]$, $X_i \in \mathbb{R}^C$. Then, we follow it with a dense layer to fully exploit inter-channel correlation, as formulated below:

$$Y_i = \alpha_i X_i, \qquad (6)$$

where α_i represent the weight set of i-th dense layer and $Y_i = [y_{i1}, ..., y_{ij}, ..., y_{iC}]$, $Y_i \in \mathbb{R}^C$. We utilize concatenation and slice operation and softmax function to produce the weight of the corresponding channel of different features. This process can be expressed as:

$$W_j = softmax(Y_j), \qquad (7)$$

where $Y_j = [y_{1j}, ..., y_{ij}, ..., y_{Mj}]$, $Y_j \in \mathbb{R}^M$ and $W_j = [w_{1j}, ..., w_{ij}, ..., w_{Mj}]$, $W_j \in \mathbb{R}^M$. The final output of SFFM is obtained as the following formula:

$$r_j = \sum_{i=1}^{M} m'_{ij} = \sum_{i=1}^{M} w_{ij} \cdot m_{ij}, \qquad s.t. \sum_{i=1}^{M} w_{ij} = 1, \quad (8)$$

where $R = [r_1, ..., r_j, ..., r_C]$, $r_j \in \mathbb{R}^{W \times H}$ and $m'_{ij} \in \mathbb{R}^{W \times H}$ denotes the j-th channel of the i-th rescaled feature maps M'_i. SFFM aims to incorporate hierarchical features with as few parameters as possible and each weight vector W_i, $W_i \in \mathbb{R}^C$ in SFFM depends on global features of all intermediate feature maps, which is different from channel attention in SENets [12].
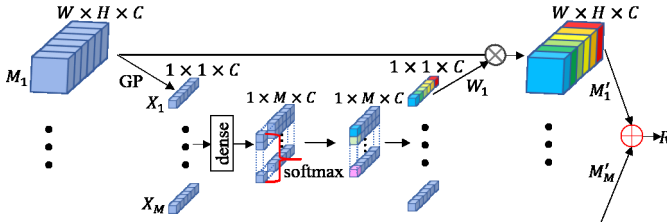
III. EXPERIMENT

A. Implementation details

At first, we pre-train our model on 91 images from Yang *et al.* [18] and 200 images from the Berkeley Segmentation Dataset [19]. To further improve the performance, we use a newly-proposed high-quality image dataset DIV2K [20] which consists of 800 images to fine-tune our pre-trained model. Data augmentation (rotation and flip) is also performed on the 291-image dataset and DIV2K dataset. To produce LR images, we downscale the HR images on particular scaling factors with bicubic interpolation. The proposed method is compared on four widely used benchmark datasets: Set5 [21], Manga109 [22], BSD100 [23], Urban100 [24]. For fair comparison, we evaluate the model with PSNR and SSIM on Y channel (i.e., luminance) of transformed YCbCr space.

In our final architecture LFFN, 15 spindle modules, each contains 4 spindle blocks, are constructed (i.e., B4M15). We initialize all convolutional filters using the method of He *et al.* [25]. We use the L1 loss as our loss function instead of the L2. For optimization, we use the ADAM optimizer [26] by setting $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. We use 16 RGB input patches of size 32×32 from the LR images for training, and the initial learning rate is set to 8×10^{-4} and then decreased to half every 20 epochs. In order to accelerate the convergence, we adopt the adjustable gradient clipping [2] which has been well implemented in tensorflow. Both training stages are configured the same as demonstrated above except that the initial learning rate is set to 4×10^{-4} during the fine-tuning. Training a LFFN roughly takes four days with a GTX 1080Ti GPU on the $\times 2$ model.

B. Model Analysis

Table I shows the effects of spindle block and softmax feature fusion module (SFFM). LFFN-NF is LFFN without softmax feature fusion module (SFFM) and we replace spindle blocks with residual blocks (Fig.3(a)) in LFFN-NS. The three

TABLE I
ABLATION STUDY OF SPINDLE BLOCK AND SFFM FOR SCALE FACTOR $\times 4$ ON DATASET URBAN100

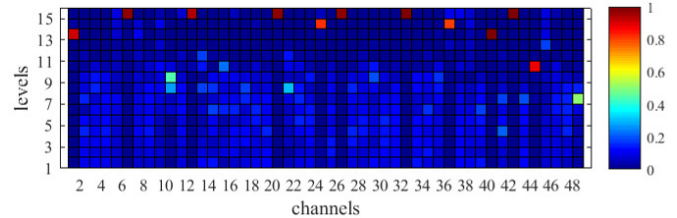|	LFFN-NF	LFFN-NS	LFFN
Spindle Block	✓	✗	✓
SFFM	✗	✓	✓
Params.(k)	1,497	4,770	1,531
PSNR/SSIM	25.97/0.7821	26.20/0.7893	26.24/0.7902

Fig. 5. Information source distribution for different channels of the feature maps used for reconstruction. The figure is drawn by visualizing the weight vector W of intermediate feature maps for the "butterfly" image from Set5 dataset on $\times 4$ enlargement.

Fig. 4. Softmax feature fusion module (SFFM) architecture. $\otimes$ denotes element-wise product. GP represents global average pooling. "dense" represents the fully connected layers applied to global features from different modules.

TABLE II
BENCHMARK SISR RESULTS. AVERAGE PSNR/SSIM FOR SCALE FACTOR ×2, ×3 AND ×4 ON DATASETS SET5, MANGA109, BSD100 AND URBAN100. RED COLOR INDICATES THE BEST PERFORMANCE.

Algorithm	scale	Params(K)	Mult-Adds(G)	Set5		Manga109		BSD100		Urban100	
				PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
VDSR [2]	2	665	612.6	37.53	0.9587	37.22	0.9750	31.90	0.8960	30.76	0.9140
DWSR [27]	2	373	344.0	37.42	0.9568	37.27	0.9719	31.85	0.8944	30.46	0.9162
DRRN [7]	2	297	6796.9	37.74	0.9591	37.60	0.9736	32.05	0.8973	31.23	0.9188
MemNet [13]	2	677	2665.0	37.78	0.9597	37.72	0.9740	32.08	0.8978	31.31	0.9195
CARN [8]	2	1592	222.8	37.76	0.9590	38.28	0.9754	32.09	0.8978	31.92	0.9256
LFFN	2	1522	342.8	37.95	0.9597	38.73	0.9765	32.20	0.8994	32.39	0.9299
LFFN-S	2	173	37.9	37.66	0.9585	37.93	0.9746	31.96	0.8963	31.28	0.9192
VDSR [2]	3	665	612.6	33.66	0.9213	32.01	0.9340	28.82	0.7976	27.14	0.8279
DWSR [27]	3	373	344.0	33.75	0.9209	32.14	0.9323	28.80	0.7972	27.22	0.8293
DRRN [7]	3	297	6796.9	34.03	0.9244	32.42	0.9359	28.95	0.8004	27.53	0.8378
MemNet [13]	3	677	2665.0	34.09	0.9248	32.51	0.9369	28.96	0.8001	27.56	0.8376
CARN [8]	3	1592	118.8	34.29	0.9255	33.47	0.9429	29.06	0.8034	28.06	0.8493
LFFN	3	1534	153.6	34.43	0.9266	33.65	0.9445	29.13	0.8059	28.34	0.8558
LFFN-S	3	185	18.1	34.04	0.9233	32.80	0.9381	28.91	0.8005	27.51	0.8372
VDSR [2]	4	665	612.6	31.35	0.8838	28.83	0.8870	27.29	0.7251	25.18	0.7524
DWSR [27]	4	373	344.0	31.39	0.8829	29.01	0.8855	27.27	0.7246	25.27	0.7552
DRRN [7]	4	297	6796.9	31.68	0.8888	29.18	0.8914	27.38	0.7284	25.44	0.7638
MemNet [13]	4	677	2665.0	31.74	0.8893	29.42	0.8942	27.40	0.7281	25.50	0.7630
CARN [8]	4	1592	90.9	32.13	0.8937	30.45	0.9073	27.58	0.7349	26.07	0.7837
LFFN	4	1531	87.9	32.15	0.8945	30.66	0.9099	27.52	0.7377	26.24	0.7902
LFFN-S	4	183	11.7	31.79	0.8886	29.76	0.8979	27.42	0.7308	25.52	0.7673

networks have the same number of basic blocks (B4M15). Compared with LFFN, the performance of LFFN-NS degraded and the parameters increased by three times, indicating that the proposed spindle block is more effective than residual block. LFFN is obviously superior to LFFN-NF, and the parameters are not increased much, revealing that SFFM is valid for incorporating hierarchical information. Beyond that, as shown in Fig.5, the different channel information of the feature maps used for reconstruction come from all levels. And high-level features play a major role in some channels, while low-level features dominate in other channels, which indicates that aggregating hierarchical features is important for SISR and SFFM can implement it well.

C. Comparisons With State-of-the-Art Methods

We compare LFFN (B4M15) and LFFN-S (B4M4 + depth-wise convolution) with state-of-the-art methods. We also compare parameters and computation (Mult-Adds) of each method. And Mult-Adds is calculated by assuming that the spatial resolution of HR image is 1280×720. As shown in Table II, our LFFN performs favorably against state-of-the-art methods on all datasets. LFFN exceeds Memnet [13] by a margin of 0.41 PSNR while being 30.32 times less compute than Memnet for upscaling ×4 on Set5. Our smallest network LFFN-S has Mult-Adds about 0.44% of MemNet, 0.17% of DRRN and 3.40% of DWSR on ×4 enlargement, respectively, but still achieves comparable performance. Fig.1 shows the execution time of different methods. We use the original codes of state-of-the-art methods to evaluate the runtime on the same machine with 2.1 GHz Intel Xeon CPU and GTX 1080 Ti GPU (12G Memory). LFFN faster, lighter and more accurate than the latest lightweight network CARN [8]. LFFN-S is about 400 times faster and 3.7 times smaller than MemNet

We also provide qualitative comparison in Fig.6. Our smallest network LFFN-S can produce almost the same result as
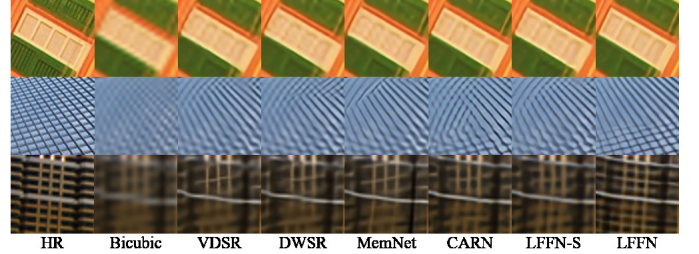

Fig. 6. Visual comparison for ×3 SR on "img013", "img062", "img085" from the Urban100 dataset.

other state-of-the-art methods (e.g., MemNet). Besides, LFFN recovers clearer, more accurate contours and less artifacts than other methods.

IV. CONCLUSION

In this paper, we propose a novel lightweight feature fusion network (LFFN) for single image super-resolution. In order to build a more effective and accurate architecture, we pay more attention to full usage of the feature map information. Whether softmax feature fusion module (SFFM) or the proposed spindle block which serves as the basic building unit can significantly improve the representational capacity of a network with fewer parameters. Experiments well demonstrate the effectiveness of our method.

REFERENCES

[1] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *European conference on computer vision*. Springer, 2014, pp. 184–199.

[2] J. Kim, J. K. Lee, and K. M. Lee, "Accurate image super-resolution using very deep convolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1646–1654.

[3] J. Kim, J. Kwon Lee, and K. Mu Lee, "Deeply-recursive convolutional network for image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1637–1645.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[5] J. Chu, J. Zhang, W. Lu, and X. Huang, "A novel multiconnected convolutional network for super-resolution," *IEEE Signal Processing Letters*, vol. 25, no. 7, pp. 946–950, 2018.

[6] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *The IEEE conference on computer vision and pattern recognition (CVPR) workshops*, vol. 1, no. 2, 2017, p. 4.

[7] Y. Tai, J. Yang, and X. Liu, "Image super-resolution via deep recursive residual network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, no. 2, 2017, p. 5.

[8] N. Ahn, B. Kang, and K.-A. Sohn, "Fast, accurate, and, lightweight super-resolution with cascading residual network," *arXiv preprint arXiv:1803.08664*, 2018.

[9] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning." in *AAAI*, vol. 4, 2017, p. 12.

[10] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017, pp. 5987–5995.

[11] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *arXiv preprint*, pp. 1610–02 357, 2017.

[12] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *arXiv preprint arXiv:1709.01507*, vol. 7, 2017.

[13] Y. Tai, J. Yang, X. Liu, and C. Xu, "Memnet: A persistent memory network for image restoration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4539–4547.

[14] T. Tong, G. Li, X. Liu, and Q. Gao, "Image super-resolution using dense skip connections," in *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, 2017, pp. 4809–4817.

[15] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network."

[16] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1874–1883.

[17] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[18] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE transactions on image processing*, vol. 19, no. 11, pp. 2861–2873, 2010.

[19] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 5, pp. 898–916, 2011.

[20] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, vol. 3, 2017, p. 2.

[21] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," 2012.

[22] Y. Matsui, K. Ito, Y. Aramaki, A. Fujimoto, T. Ogawa, T. Yamasaki, and K. Aizawa, "Sketch-based manga retrieval using manga109 dataset," *Multimedia Tools and Applications*, vol. 76, no. 20, pp. 21 811–21 838, 2017.

[23] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 2. IEEE, 2001, pp. 416–423.

[24] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5197–5206.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[27] T. Guo, H. S. Mousavi, T. H. Vu, and V. Monga, "Deep wavelet prediction for image super-resolution," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2017.