Fast, Accurate, and Lightweight Super-Resolution with Cascading Residual Network

Ziyang Chen
Australian National University
u6908560@anu.edu.au

Han Zhang
Australian National University
u7235649@anu.edu.au

Abstract

Single image super-resolution (SISR) is a hot area of computer vision. The methods based on convolutional neural network (CNN) performs well, but require calculations and operations, which is not suitable for practical applications. In response to this, the CARN family models were proposed, which is a fast, light weight and accurate model. We reproduced the CARN family models and trained the CARN family models using the DIV2K dataset. We tested the models on same natural image datasets and an unnatural image dataset collected by us. The models performed well on natural images but unstable on unnatural images. We also compared some different loss functions and found that it has little effect on the performance of the model.

1. Introduction

As a computer vision task, super-resolution (SR) refers to the use of one or more low-resolution (LR) images to obtain high-resolution (HR) images. High resolution means that the pixel density in the image is high, which can provide more details, and these details are indispensable in many practical applications, such as shooting high-resolution medical and satellite images. The Cascading Residual Network (CARN) [2] solves the SISR problem, which uses a single LR image to restore HR.

In recent years, CNN-based methods have given outstanding performance in SISR tasks. The effect of the early simple neural network structure needs to be improved. Using a more complex network structure and a deeper convolutional layer can greatly improve the performance of the SISR task, but this is achieved at the expense of time and high computational complexity. Therefore, it is necessary to build a lightweight deep learning model to make it suitable for real-world applications, reduce the number of parameters and operations and increase the calculation speed.

In response to the above problems, the CARN family was proposed. First, the CARN model was proposed to improve performance, and then expanded to CARN-M model to optimize speed and number of operations. In short, CARN based on the cascading module neural network effectively improves performance on SR tasks, while the CARN-M algorithm combined with efficient residual block and recurrent networks is very effective on SR tasks. At the same time, these models only use A small number of operations and parameters.

2. Problem Statement

The SISR task is a computer vision task that restores the HR image through a single LR image. It is a many-to-one mapping, so it is usually difficult to implement. However, SISR is very useful because it is expected to break through the limitation of resolution, so it is a very active area. Recently CNN-based methods have performed well on SISR tasks. SRCNN is the first attempt of a deep learning (DL) method in the super-resolution problem. It has a simple structure, but the effect needs to be improved. Other CNN-based methods such as SRDenseNet [10], MDSR [8] and RDN [13] use more complex network structure and deeper convolutional layer, the effect is outstanding, but the cost is time and computational intensity, thus not suitable for real scenes.

To construct a fast, accurate, and lightweight super-resolution model and reduce required operations, the CARN family was proposed to reduce operation and calculation overhead and improves speed while ensuring accuracy.

3. Methodology

For the super-resolution task, we propose two novel models: CARN and CARN-M. CARN is designed for good performance and CARN-M is an extension based on CARN for high efficiency.

3.1. Cascading Residual Network

Inspired by FSRCNN [4], CARN takes LR images as input and outputs HR images restored by SR. As shown in Figure 1 [2], the design of the middle part of the CARN

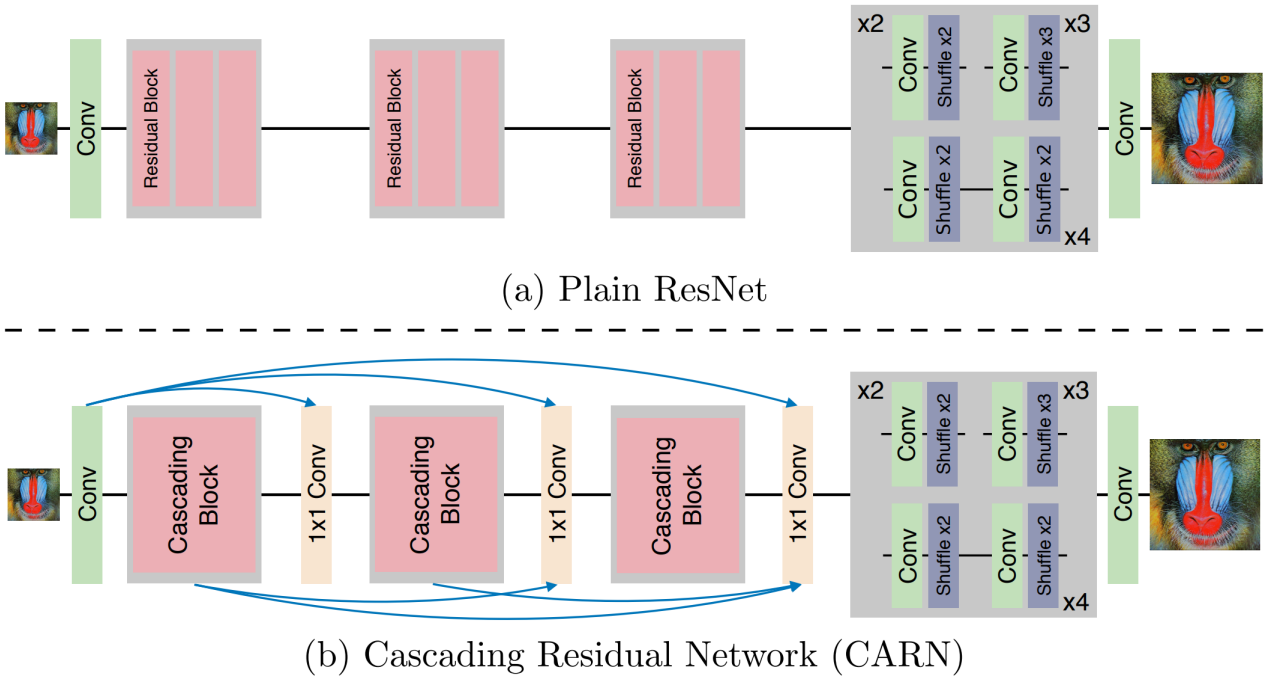

Figure 1: (a) the plain ResNet model, (b) CARN model which the global cascading connection is indicated by the blue arrows.

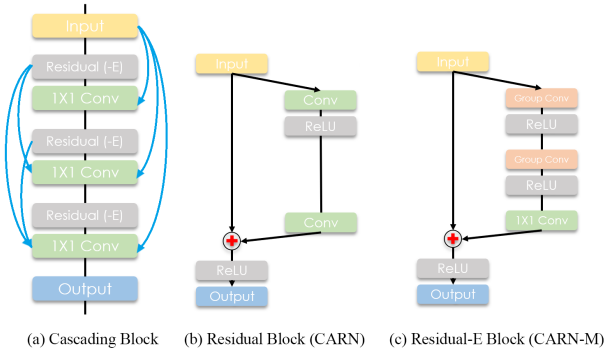

Figure 2: (a) Cascading block. (b) Residual block used in CARN model. (c) Residual-E block used in CARN-M model.

model is based on the ResNet [5], with the residual blocks in ResNet replaced by cascading blocks. For the three cascading blocks, before entering each cascading block, the output of the previous cascading block must be fused and compressed with a 1x1 convolution kernel. There are three residual blocks in each cascading block, and the three residual blocks have the similar structure with the cascading block. Figure 2 (a) and (b) shows the detailed block structures of CARN. This structure combines the features of multiple layers and behaves as multi-level shortcut connections, which can learn multi-level representations and the information transmission is efficient [2].

3.2. Efficient Cascading Residual Network

CARN-M is similar with CARN but more efficient. The main difference of CARN-M and CARN is that CARN-M uses the efficient residual (residual-E) block as shown in Figure 2 (c). The residual-E block is similar with MobileNet [6], but instead of depthwise convolution, residual-E uses group convolution. The residual-E block consists of two group convolutions followed by a 1×1 convolution layer. Group convolution allows the model to be balanced between performance and efficiency by adjusting the size of the groups.

4. Experiments

4.1. Dataset

Following [2], we used DIV2K [1] dataset to train the model. It contains 800 training images, 100 validation images and 100 test images, the types of images are also rich. We also used Set5 [3], Set14 [12], B100 [9] and Urban100 [7] for testing and visualizing. Given that most of the pictures in the above data set are natural images, we also collected a dataset which contains only unnatural image to verify the performance of the model on unnatural images. The images in this dataset are from Pixabay with Pixabay License.

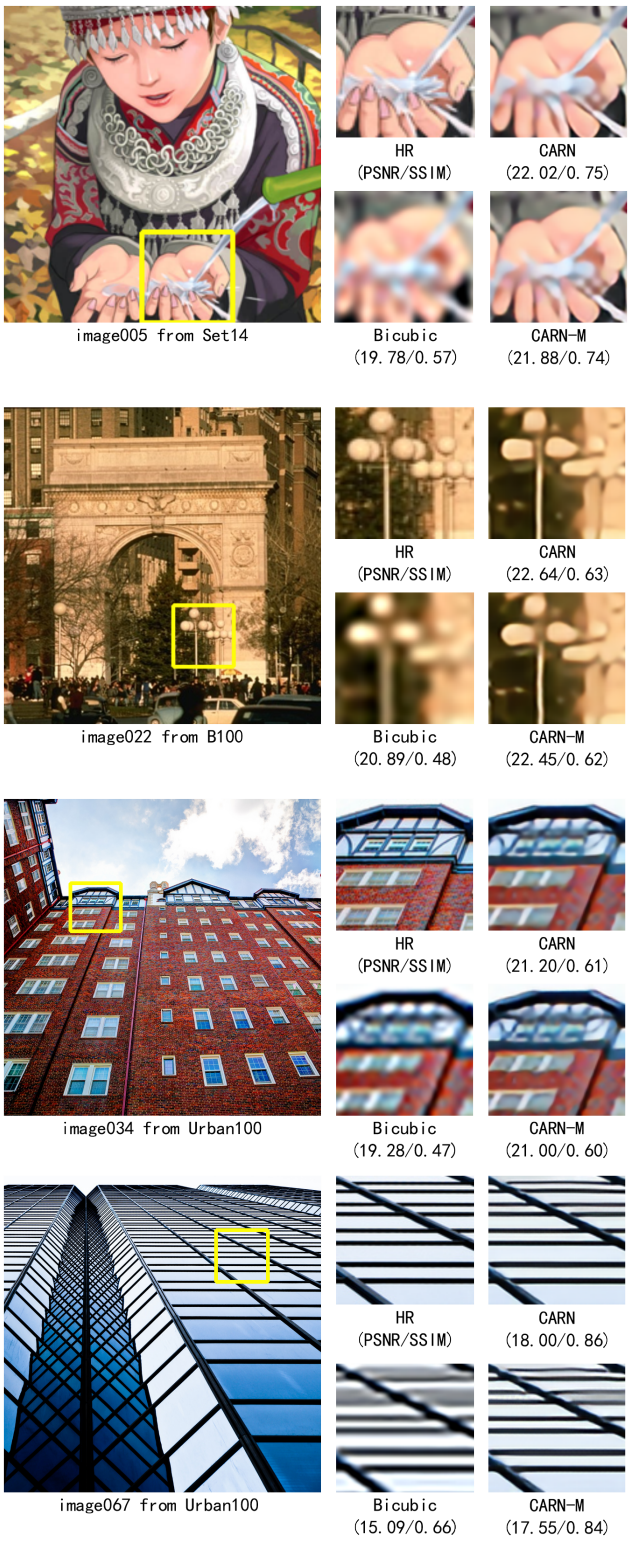

Figure 3: SR results with X4 scale, 600000 steps

Table 1: CARN and CARN-M performance on different datasets. Red figures are the highest PSNR/SSIM for this dataset on this scale while the blue figures are the second highest.

Scale	Dataset	LR (PSNR/SSIM)	CARN (PSNR/SSIM)	CARN-M (PSNR/SSIM)
X2	B100	27.37/0.8110	30.02/0.8939	30.56/0.8985
	Urban100	24.53/0.8061	30.28/0.9232	29.64/0.9163
	Set5	30.44/0.8982	35.61/0.9494	35.32/0.9473
	Set14	27.32/0.8264	31.23/0.9036	31.02/0.9009
	Unnatural	25.50/0.7834	25.76/0.8063	25.82/0.8080
X3	B100	25.47/0.7170	27.73/0.8066	27.57/0.8033
	Urban100	22.57/0.7114	26.50/0.8462	26.01/0.8354
	Set5	27.80/0.8347	32.23/0.9125	31.95/0.9096
	Set14	25.20/0.7395	28.14/0.8275	27.91/0.8233
	Unnatural	24.43/0.7267	25.09/0.7764	25.20/0.7771
X4	B100	24.24/0.6359	26.26/0.7336	24.63/0.6653
	Urban100	21.25/0.6248	24.55/0.7793	23.44/0.7305
	Set5	25.86/0.7639	30.12/0.8764	29.84/0.8709
	Set14	23.72/0.6584	26.50/0.7645	26.27/0.7595
	Unnatural	23.56/0.6788	24.47/0.7382	23.92/0.7224

4.2. Evaluation

Following [2], we used two commonly used metrics to quantify the results: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity index (SSIM) [11]. At the same time, we also compare the results based on subjective judgments.

4.3. Initial Results

We trained the CARN model and CARN-M model using DIV2K dataset. Due to device limitation, we trained the model for 150000 steps. The patch size and batch size are both 64, and the loss function is L1 loss. (Missing CARN-M)

We tested the trained models on datasets Set14, B100 and Urban100, and used the built-in method of scikit-image SciKit library to calculate PSNR and SSIM. The results are shown as Figure 3. We also tried the model provided by the author, whoever the result is different with ours. In our results, the overall PSNR and SSIM are lower, even for the original bicubic images. This may be because we used a different method to calculate the PSNR and SSIM. However, both PSNR and SSIM have been greatly improved.

We also tested the speed of the two model. Our test platform has an Intel i7-8750H CPU and a Nvidia 1050Ti GPU. Running the CARN-M model to process x4 scale of Urban100 dataset costed 75.59s while CARN costed 79.67s. On B100 dataset CARN costed 20.16s. However, CARN-M model uses fewer computing resources compared with CARN model.

4.4. Performance on unnatural images

We used the DIV2K data set as the training set. We tested the performance of the model on the B100, Urban100, Set14 and Set5 datasets. The images in DIV2K are all natural images, so we collected an unnatural image dataset (Unnatural) and verified the performance of the models on unnatural image SR task.

Table 1 shows the average PSNR and SSIM of the test datasets. In most cases, CARN performs better than CARN-M. Overall, CARN and CARN-M both improved the average PSNR and average SSIM on each data set and each scale. However, the PARN/SSIM improvement of CARN and CARN-M on the unnatural data set is not as good as the B100, Urban100 and Set5 data sets with only natural images, and the Set14 data set with only one unnatural image. Figure 4 shows some examples of x4 scale results for Unnatural dataset. For some images, the PSNR of some SR images is even lower than that of LR images, such as img014. However, the SSIM for all the images are improved. Subjectively, img014 has a lot of lines and textures. Although the SR image looks clearer, the lines and textures do not match the HR image very well. CARN and CARN-M have unstable performance on unnatural images, but they perform well in improving SSIM. Adding unnatural images in the training set may improve performance in unnatural images.

4.5. Compare different loss function

We tested different loss functions: L1, smooth L1 and SME. Table 2 shows the x4 scale performance of CARN model with different loss functions on each dataset. L1 loss

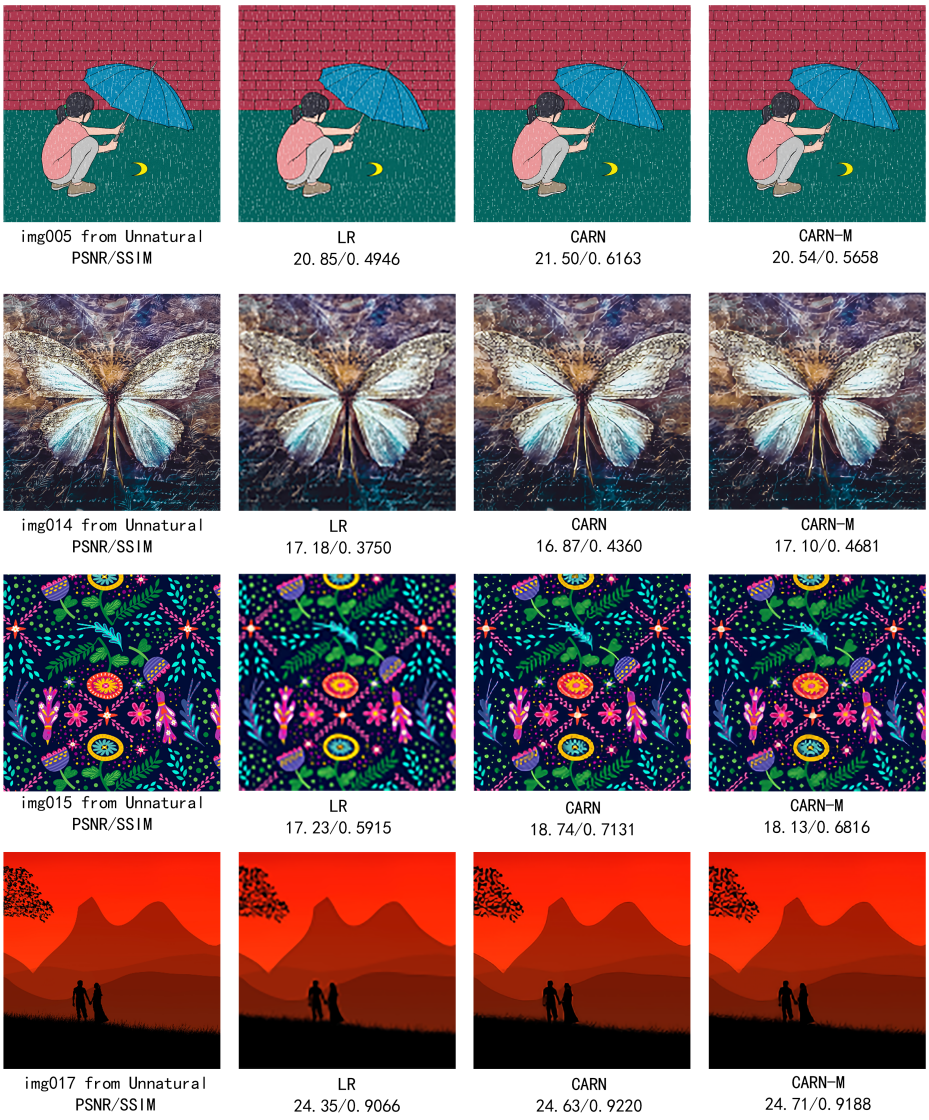

Figure 4: Some sampels from unnatural dataset

Table 2: Comparison of L1, MSE, SmoothL1 loss function on X4 scale

Dataset	LR (PSNR/SSIM)	L1 (PSNR/SSIM)	MSE (PSNR/SSIM)	SmoothL1 (PSNR/SSIM)
Set5	25.86/0.7639	29.87/0.8713	29.79/0.8696	29.75/0.8704
Set14	23.72/0.6584	26.32/0.7609	26.32/0.7591	26.27/0.7593
Urban100	21.25/0.6248	24.24/0.7694	24.24/0.7671	24.23/0.7681
B100	24.24/0.6359	26.12/0.7308	26.12/0.7296	26.10/0.7299
Unnatural	23.56/0.6788	24.34/0.7367	24.34/0.7309	24.27/0.7331

performs slightly better than SME and SmoothL1 in terms of PSNR and SSIM. As for training speed, we used the same platform as 4.3. and run CARN training for 1000 iterations. L1 took 740.00 seconds, MSE took 735.33 seconds, and SmoothL1 took 749.32 seconds. Overall, these three loss functions have little effect on performance.

5. Conclusion

We reproduced the CARN and CARN-M network models. We trained the model using the DIV2K dataset and compared the performance of CARN and CARN-M on B100, Urban100, Set14, Set5 and an unnatural image dataset collected by us. Overall, the performance of CARN is usually better than CARN-M, while CARN-M costs fewer computing resources than CARN. We found that the CARN and CARN-M models show good versatility on natural images, but they are unstable on unnatural images. At the same time, we explored the impact of L1, MSE, and SmoothL1 loss functions on model training and performance, and found that the model performs better when using the L1 loss function, but the difference is very small.

Going forward, we plan to add unnatural images and more types of natural images to the training set to enhance the versatility and performance of the model. Moreover, we plan to apply this network to video SR tasks and consider the time correlation of multiple frames to enhance network performance.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1122–1131, 2017. 2

[2] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Fast, accurate, and lightweight super-resolution with cascading residual network. *Computer Vision – ECCV 2018 Lecture Notes in Computer Science*, page 256–272, 2018. 1, 2, 4

[3] Marco Bevilacqua, A. Roumy, Christine Guillemot, and Marie-Line Alberi-Morel. Low-complexity single image super-resolution based on nonnegative neighbor embedding. 09 2012. 2

[4] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network, 2016. 1

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2

[6] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017. 2

[7] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5197–5206, 2015. 2

[8] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1132–1140, 2017. 1

[9] 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 *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 416–423 vol.2, 2001. 2

[10] Tong Tong, Gen Li, Xiejie Liu, and Qinquan Gao. Image super-resolution using dense skip connections. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4809–4817, 2017. 1

[11] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 4

[12] Jianchao Yang, John Wright, Thomas S. Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, 2010. 2

[13] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(7):2480–2495, 2021. 1

Personal Reflection

This term project gave me an idea of what a research project should look like. Through reproduction and experimentation, I have a deeper understanding of the super-resolution field and had some new ideas on this field. The process of the research is full of fun, and I really enjoy it.

I also realized the lack of my own knowledge, and I still need to improve in understanding models and designing experiments. Next, I will read more literature and learn more knowledge, hoping to improve my ability in future projects and make improvements to the project.

Confidential Peer Review

In doing this project, to the best of my judgement, I confirm that Ziyang Chen mainly contributed to the code for the model and made the slides. He also organized the report to LATEX format. His overall contribution is about 55%. I (Han Zhang) mainly contributed to the report and did the experiments. I also wrote some codes, organized the readme file and made some pages of the slides. My contribution is about 45%. We both contributed to the presentation.