Image Super-Resolution Using SRCNN and WDSR

Yixian Gan[1] and Ting Cheng[1]

[1]Harvard University

May 10, 2023

1 Introduction

Image super-resolution is an emerging field of research that aims to recover a high-resolution image from a low-resolution image. Super-resolution techniques have numerous practical applications. For example, in medical imaging, super-resolution can improve the accuracy of diagnoses by providing higher resolution images of anatomical structures. In remote sensing, super-resolution can help identify objects and features that are too small to be detected by the imaging system.

In the long history of image super-resolution, there are multiple literature that proposed novelty methods to better infer the intermediate pixel values after up-sampling. Techniques like super-sampling or over-sampling we discussed in lecture are examples of these. Historically, these methods have performed decently. In recent years, the huge success and immeasurable potential of neural networks, especially convolutional neural networks (CNN), in high-level computer graphic tasks incentivizes people's interest in applying neural networks in image super-resolution tasks. Various methods have been developed to achieve super-resolution. A very classic super resolution technique, Super-Resolution Convolutional Neural Network (SRCNN) was first introduced in 2014 [1]. SRCNN learns an end-to-end mapping between low-resolution and high-resolution images using a shallow convolutional neural network(CNN). It adapted and generalized example-based learning methods to CNN models. Another popular super resolution technique is Wide Activation for Efficient and Accurate Image Super-Resolution(WDSR)[2]. WDSR is a state-of-the-art approach that is based on the SRResNet architecture and consists of multiple residual blocks, which will be discussed in details later this article.

In this project, we aim to compare the performance of these two techniques and evaluate their ability to improve the quality of low-resolution images using various evaluation metrics such as Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE).

2 Methods

2.1 SRCNN

The SRCNN consists of the following operations as shown in Figure 1:

1. Preprocessing and Feature Extraction: Upscale low-resolution image to desired high-resolution size and extract a set of feature maps from the upscaled low resolution image
2. Non-linear Mapping: Map the feature maps representing low-resolution to high-resolution patches.

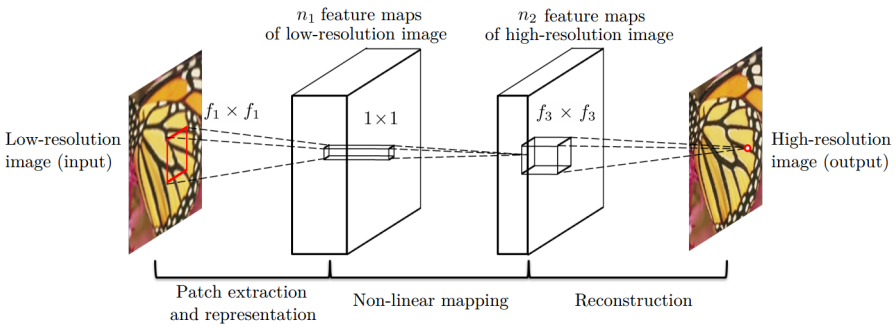


Figure 1: SRCNN Step-by-Step Operations

3. Reconstruction: Produce the high-resolution image from high-resolution patches.

2.2 WDSR

The WDSR model consists of the following components as shown in Figure 2:

1. Convolution Layer
2. Residual Block
3. Additional Convolution Layer(s)
4. Pixel Shuffle (Upsampling)

The WDSR network was built based on the EDSR network by removing the redundant convolution layers before and after upsampling and modifying the residual block structure.

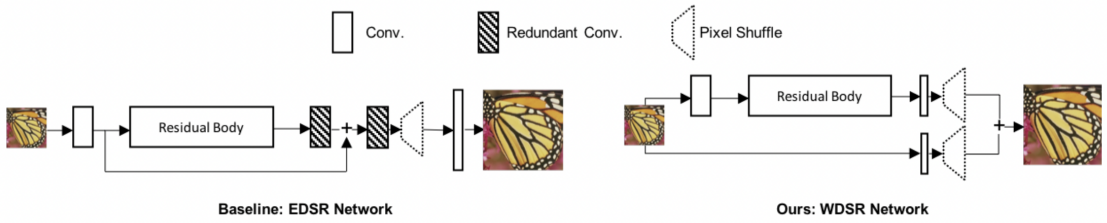


Figure 2: WDSR Model Structure

If we take a closer look at the individual residual block, WDSR-A has a wide activation and WDSR-B has a wider activation and lower-rank convolution layer compared to the original EDSR network.

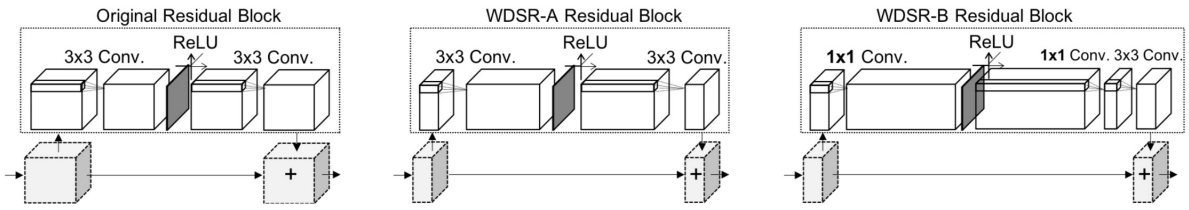


Figure 3: WDSR Individual Residual Block Structure

3 Implementation

3.1 Platform and Libraries

In this project, we implemented all our models on TensorFlow2.0 platform. Python is the major language used, and all codes are developed in Jupyter notebooks. In addition, we used `PIL.Image` module, mainly for results visulization.

3.2 Training Data

For training our model, we used Div2K dataset. Div2K contains 1,000 2K high resolution images of diverse subjects, split by 800-100-100 into train-val-test datasets. It also provides corresponding degraded images as training samples. In our experiment, we selected the image downsized by 4x using Bicubic method as the model inputs.

To address the problem of varying size input, as well as increase the number of training samples, we further decomposed each image in the dataset into multiple 224x224 subsections (patches). These subsections and their degraded counterparts are used as training samples and ground truth. Using 224x224 windows and stride of 300, we obtained 13,000 training samples. The code for preprocessing data is provided in `data_preprocessing.ipynb`.

3.3 Model Structure

SRCNN

In their original paper, Dong et al. proposed several variants of the basic model structure. In our project, we adopted one variant and modified it for better performance. Similar to the original model, out model has a 3-layer structure, with kernel size 9-5-5 for each convolutional layer. But for the first two layers, we increased their number of feature maps from the original 64 and 32 to 128 and 64 respectively. This is out of consideration that the input images in our dataset is much larger than the training image in the original paper, so we want to increase the number of trainable parameters in the model for better expressiveness.

WDSR

Compared to SRCNN, WDSR has more complicated structures, and more tunable hyper-parameters for model training. For example, the original paper proposed two variants of the model structure. The authors also discussed using different number of blocks in the model may have effects on the performance. In our project, we primarily focus on WDSR model as we deem it to be more powerful. We started with the basic version of WDSR - the model consisted with 5 `WDSR-B` type of residual blocks, and then

spent significant amount of time playing with these hyper-parameters to see if we can achieve better performance. More details about our experiments are provided in the next section.

4 Experiments

Yu, Fan, and Huang also proposed several variants of the WDSR model in their paper. In our project, we also played a little bit with these model hyper-parameters, along with some other factors that may affect the training, to get better performance.

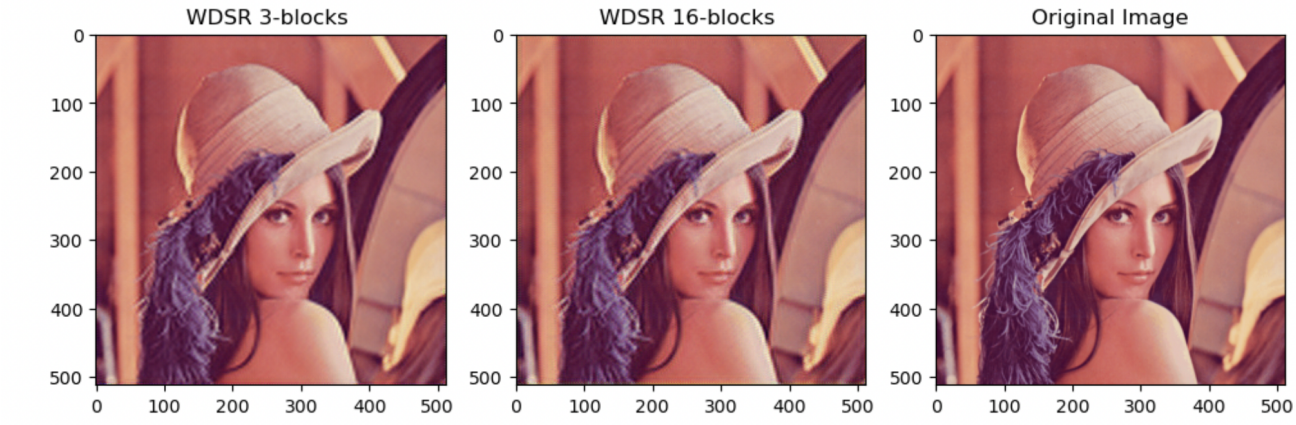
Number of Residual Blocks

As shown in Figure 2, the WDSR model has one main path and one residual path, where the main path is consisted of several residual blocks. Intuitively, we believe that the more blocks in the main path, the deeper the model, and thus the more capacity to learn the training data. Original paper proposed three options, 3, 5, or 8 blocks. Here, we tried them all and decided to go deeper. We also tried out a model with 16 blocks. In each case, the number of parameters, the time required for training, the MSE and PSNR of the final model are documented for model comparison.

# of blocks	3	5	8	16
# of parameters	85K	130K	197K	376K
Training time	13min 8s	13min 24s	17min 24s	58min 30s
MSE	0.06	0.0760	0.0741	0.0912
PSNR [db]	26.1658	25.3057	25.3344	23.6806

Table 1: WDSR Performance with Different Number of Residual Blocks

As shown in Table 1, the training time increases as the number of blocks increase. Initially, the training time increases slowly, but as the number of blocks increases from 8 to 16, the training time increases exponentially. When we have more than 3 blocks, MSE increases and PSNR decreases undesirably, indicating that we have already reached a saturation point (additional residual blocks no longer improve the model's performance).



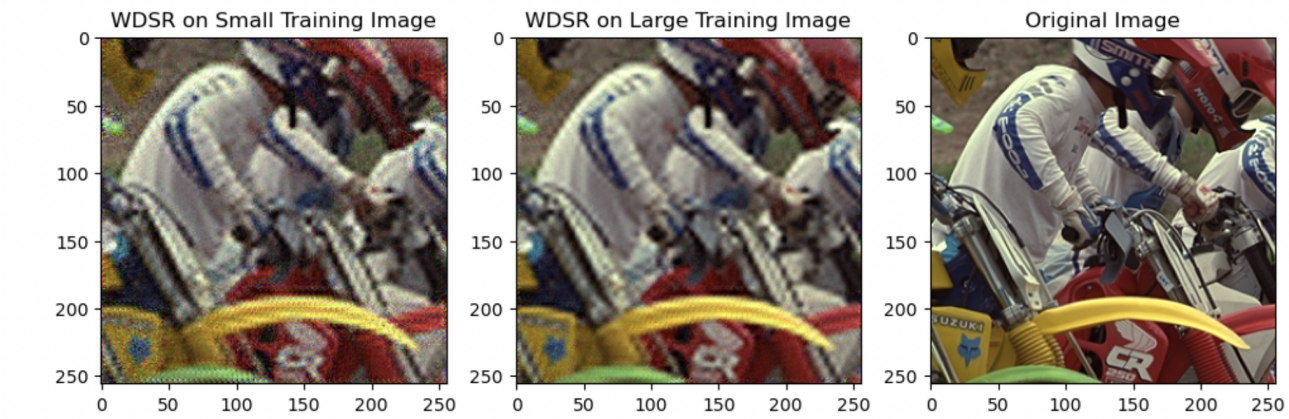
Note: Deeper network suffers more from 'checkerboard effect'.

Training Image Size

Another factor that may affect model performance is the image size of the training data. In the original paper, the authors used 96x96 patches of RGB images. However, we believed that the model may benefit from larger input images, so we used 224x224 patches of RGB images at first. But we also tried 96x96 patches to see if there is any significant difference.

Image Size	Large (3 blocks)	Large (5 blocks)	Small (3 blocks)	Small (5 blocks)
# of parameters	85K	130K	85K	130K
Training time	13min 8s	13min 24s	22min 19s	20min 58s
MSE	0.0623	0.0760	0.0728	0.0753
PSNR [db]	26.1658	25.3057	25.3989	25.2611

Table 2: WDSR Performance with Different Image Size



As shown in Table 2, increasing the size of the training image results in a slightly higher image quality based on the MSE and PSNR metrics and a significantly shorter training time.

Residual Block Structure

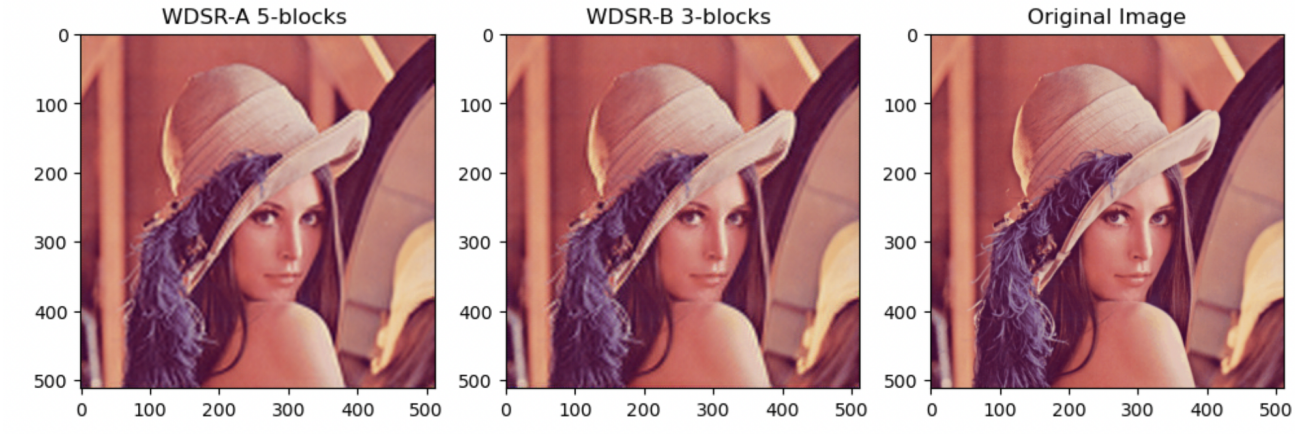
In the original paper, authors also provided two possible structures for the residual block, which they called `WDSR-A` and `WDSR-B`. The main difference is that `WDSR-A` uses 2 convolutional layers and 3x3 kernel for each layer, whereas `WDSR-B` uses 3 layers and 1x1 convolution kernel for the first two layers (see Figure 2 for more details). The authors argued that `WDSR-B` is able to get widen activation with minimal extra parameters, and thus is more favorable. They also claimed that `WDSR-B` structure is able to better ameliorate the "checkerboard effect".

Image Size	WDSR-A (3 blocks)	WDSR-A (5 blocks)	WDSR-B (3 blocks)
# of parameters	240K	388K	85K
Training time	9min 40s	21min 11s	13min 8s
MSE	0.0711	0.0612	0.0623
PSNR [db]	25.5856	27.0784	26.1658

Table 3: WDSR-A and WDSR-B Performance Comparison

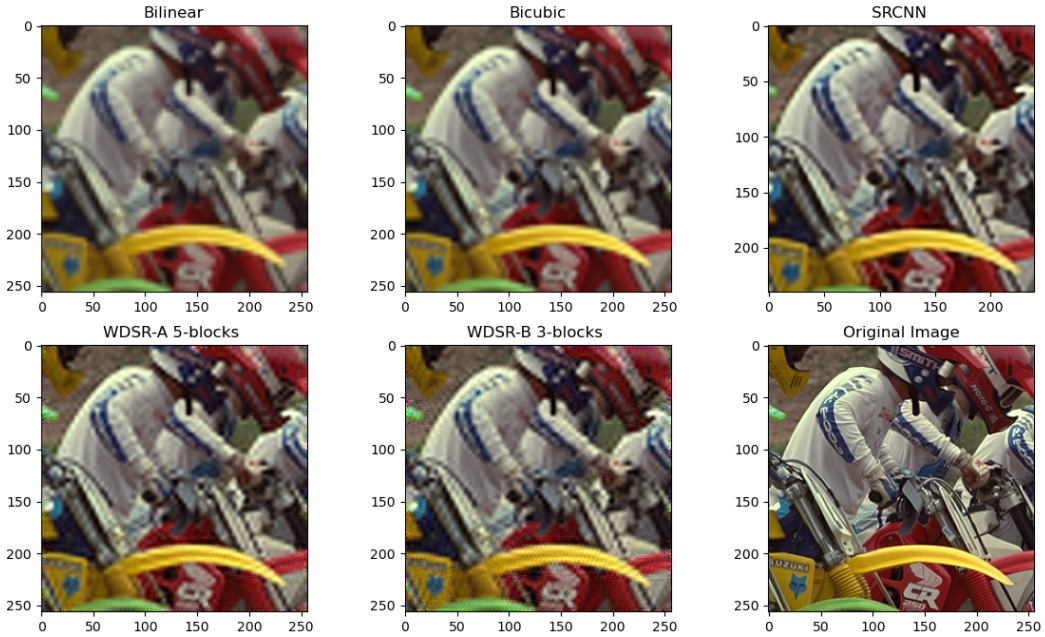
As shown in Table 3, WDSR-B has significantly fewer parameters than WDSR-A. With the same number

of blocks, the MSE and PSNR results indicate that WDSR-B produces slightly better images in this experiment. But it should be noted that WDSR-B does not always outperform WDSR-A, as discussed later.



5 Result Comparison

In addition to SRCNN and WDSR, we performed Bilinear and Bicubic interpolations as the baseline method for comparison.



Bilinear vs Bicubic

As we learned in class, bilinear interpolation means applying a linear interpolation in two directions, which takes weighted average of the surrounding four pixels in the origin image to produce the output.

This method is relatively simple and fast, but it can produce a slightly blurry image with some visible artifacts when the image is upscaled.

Bicubic interpolation, on the other hand, is a more sophisticated method that uses a larger kernel to estimate the values of new pixels based on the surrounding 16 pixels in the original image. Bicubic interpolation produces a smoother and more accurate image, especially when upscaling the image. However, it is computationally more expensive than bilinear interpolation.

In general, bilinear interpolation is a good choice for simple image resizing tasks where speed is important, while bicubic interpolation is a better choice when image quality is the top priority.

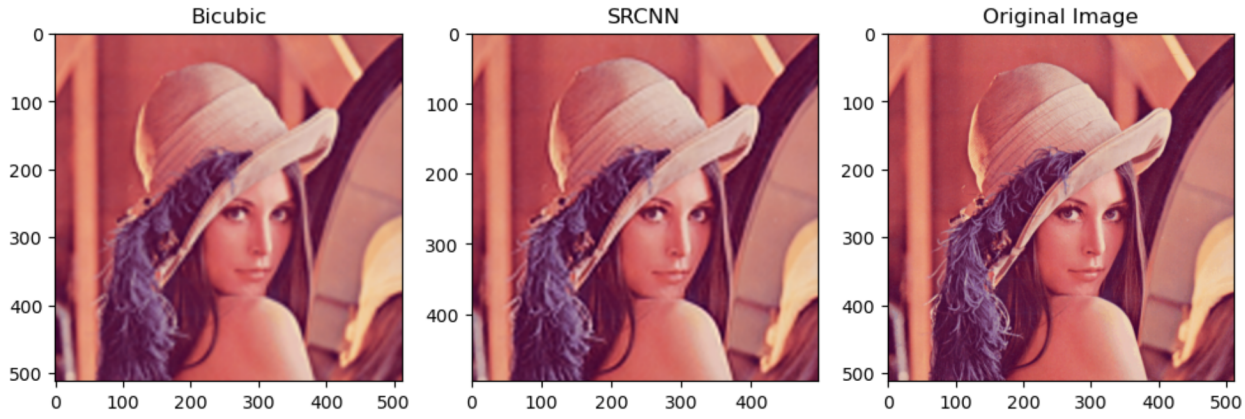
Performance

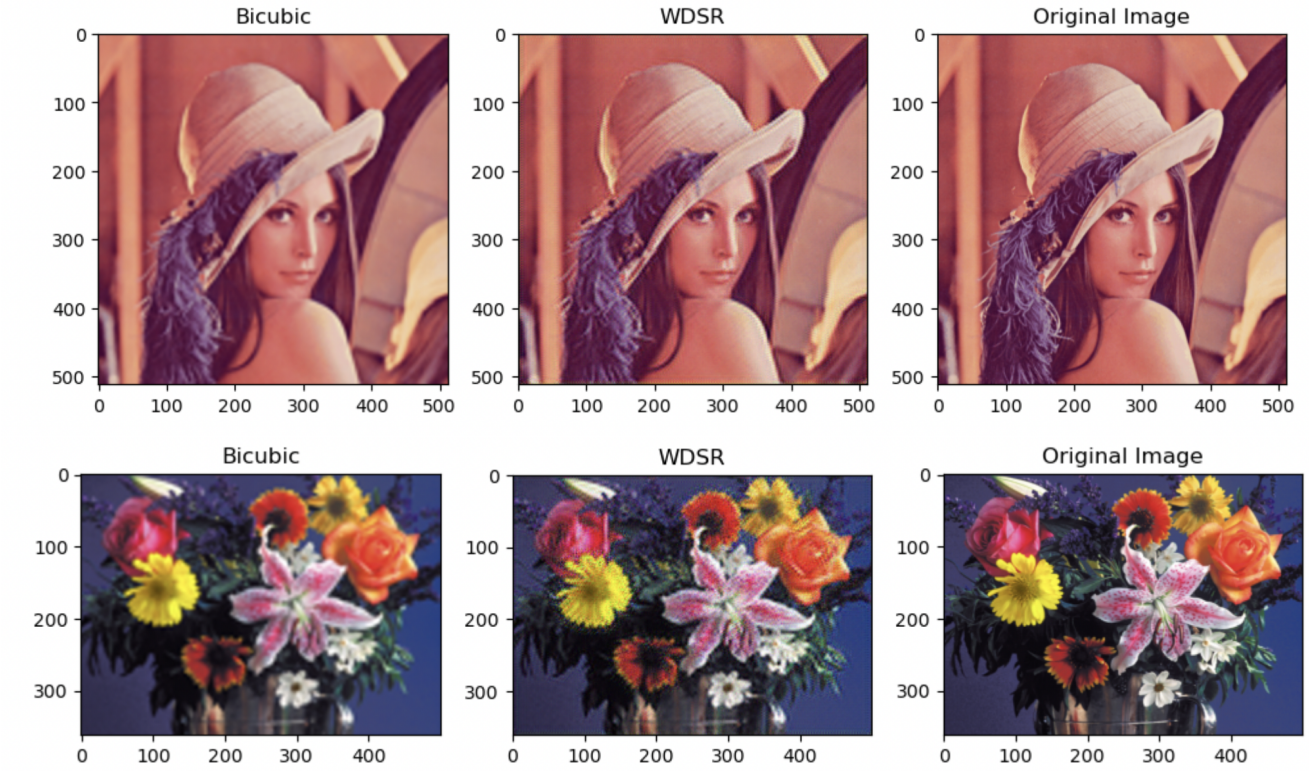
Method	Bicubic	SRCNN	WDSR-A (5 blocks)	WDSR-B (3 blocks)
# of parameters	N/A	241K	388K	85K
Training time	N/A	22min 12s	21min 11s	13min 8s
MSE	0.1021	0.0460	0.0612	0.0623
PSNR [db]	22.7810	29.7038	27.0784	26.1658

Table 4: Performance with Different Networks

Note: We choose the WDSR network with the optimal number of blocks.
As shown in Table 4, surprisingly, based on the MSE and PSNR metrics, SCRNN produces the best-quality image. Although SRCNN is a relatively simple model, it takes the longest time to train. On the other hand, WDSR-A takes longer to train but yields a slightly better quality image compared to WDSR-B.

Demo





As shown in the Demo, the images produced by SRCNN and WDSR are quite similar in terms of quality. However, WDSR is more flexible as it is capable of handling various sizes of original images, which SRCNN cannot.

6 Summary

This project report provides a comparative analysis of the performance of two deep learning-based image super-resolution networks, namely SRCNN and WDSR. Through a series of experiments and evaluations, this study aims to determine which network performs better in terms of image quality and training time. The results indicate that both networks offer significant improvements on low-resolution images. However, despite being a more complex model, WDSR did not produce better images than SRCNN in our experiments. Nevertheless, we found that by reducing the number of parameters, WDSR-B was able to produce high-resolution images with similar quality while requiring only 60% of the training time. Therefore, a key takeaway lesson is that we should consider the task priorities (image quality or training overhead)before choosing the appropriate network for super-resolution tasks in the future.

7 Resources

Our experiments were conducted on a remote computer featuring an Tesla T4 GPU. Our research was primarily based on the following two original papers, "Image Super-Resolution Using Deep Convolutional Networks" and "Wide Activation for Efficient and Accurate Image Super-Resolution" [1][2]. We choose to use the DIV2K dataset to evaluate the performance of the SRCNN and WDSR models [3][4]. We used the code base shared by the original author of the WDSR model as a reference.

References

[1] Chao Dong, Kaiming He, Chen Change Loy, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *arXiv:1501.00092*, 2014.

[2] Jiahui Yu, Yuchen Fan, and Thomas Huang. Wide activation for efficient image and video super-resolution. *30th British Machine Vision Conference*, 2019.

[3] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[4] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017.