CS 7180 Project Report Assignment 1

Team Members:

1. Aditya Varshney
2. Luv Verma

Github Link: https://github.com/adityav1810/cs7180-homework1

Abstract

Our project delves into the use of generative adversarial networks (GANs) for Image Super Resolution (SR). **Our foundation is the pioneering work of Ledig et al., who introduced the SRGAN in their study "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network".** We recreate, retrain, and fine-tune this network to understand its pros and cons, testing it on diverse images. Initially, the network is trained using the DIV2K dataset and later evaluated on the Set14 dataset.

The original paper highlights the importance of the vgg19-based perceptual loss in capturing an image's fine textures and details. We put this to the test using low-resolution images from the Set14 dataset. Lastly, our project aims to determine how closely the peak signal-to-noise ratio of an image aligns with human perception.

Introduction

The quest to transform low-resolution images into high-resolution counterparts has long intrigued computer vision researchers. This task presents challenges such as data ground truth availability, loss of information during downscaling, computational demands, and others. In the early 2000s, these hurdles favored the exploration of dense networks, spurring keen research interest.

Pioneering this field, Dong et al. [1] applied a 3-layer Convolutional Network to Super Resolution (SR). Their approach cleverly learned sparse coding methods without explicit definition, eliminating the need for several manual preprocessing steps. Yet, there was room for improvement.

Enter Zhang et al. [2], who integrated a Residual Dense Block (RDB) to enhance texture learning. RDB's integration into SR tasks marked a crucial advancement, addressing the memory loss challenge between a network's deeper and shallower layers. Now, information could flow more rapidly and efficiently, enabling better feature extraction and computational efficiency.

Drawing inspiration from RDB's success, Tai et al. [4] introduced MemNet. It comprised a memory block with numerous residual units overseen by a gate unit. This gate unit adaptively gauged the significance of preceding residual units, equipping the model with both long and short-term memory functionalities. The outcome? Enhanced capability to recreate sharp image edges.

Furthering the cascade model, Ahn et al. [3] devised a cascading block. This block funneled into superior layers, merging into a single convolutional layer. The outcome was a more interconnected layer system, facilitating faster information propagation and improved image property learning.

Historically, many SR techniques employed bicubic interpolation to upscale low-resolution images, forming their high-resolution counterparts. However, a turning point came with Goodfellow et al. [6] and their landmark research on Generative Adversarial Networks (GANs). This innovative method introduced a unique duet: a generator

network aiming to replicate the input image and a discriminator network distinguishing between the generator's output and the genuine high-resolution image. The process continued until the discriminator could no longer differentiate between the two.

Ledig et al. [5], building on this, applied GANs to Super Resolution. Their generator embraced a ResNet-inspired architecture, equipping the model to discern both local and global features effectively.

As the 2000s progressed, the availability of high-resolution datasets and more potent computing resources reshaped SR strategies. Initially, methods mainly upscaled images using bicubic interpolation, then processed them to achieve a high-resolution result. These ground truth images, derived from bicubic interpolation, weren't always optimal. Notably, works from [1] to [4] achieved upscaling, but only to a maximum factor of 3.

Inspired by both [6] and [2], Ledig et al. [5] brought forth the SRGAN. This model boasted a generator network based on residual concepts, enhancing its scalability, and enabling it to upscale images by a factor of 4. A significant leap was the training on the high-resolution DIV2K dataset. This shift removed the need for bicubic interpolation in creating ground truth images. Instead, the method reversed; bicubic interpolation downscaled images for the generator. This change enhanced the visual quality of output images, rendering them more visually pleasing and photorealistic.

Our work is structured into distinct sections: "Methods and Architecture" covers preprocessing, the intricacies of the generator and discriminator networks, loss functions, and the chosen hyperparameters. "Results and Discussion" offers a deep dive into our findings, especially our analysis of the DIV2K and Set14 datasets. We also address "Areas of Uncertainty and Further Investigations," shedding light on challenges encountered and wrapping up our conclusions. Notably, our results emphasized that high PSNR values, while indicative of similarity, don't always capture the perceptual richness of images.

Methods and Architecture

We implement the SRGAN architecture using Pytorch. The training is initially done on the DIV2K dataset which has 800 High resolution images. The model architecture is kept unchanged from the original paper and has been shown below.

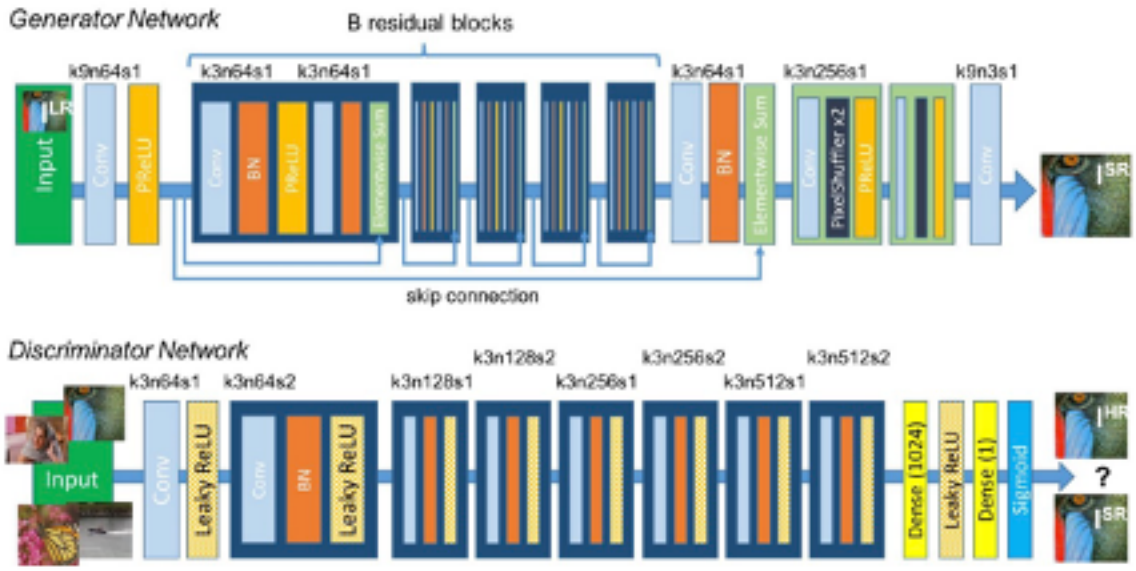


Figure 4: Architecture of Generator and Discriminator Network with corresponding kernel size (k), number of feature maps (n) and stride (s) indicated for each convolutional layer.

Figure 1: SRGAN Network Architecture

Preprocessing

The cropping technique is crucial in preparing datasets for Super-Resolution tasks, especially when training models that require fixed-size input images. Information about the scale factor, denoting the extent of image reduction, and the patch size, defining the dimensions of cropped regions, is preserved. In the context of an image sample, access is made to both the lower-resolution (LR) and higher-resolution (GT) images. The LR image's dimensions are computed, and random coordinates within it are generated to identify the top-left corner of the cropping region. Leveraging the scale factor, corresponding coordinates in the GT image are ascertained. Subsequently, square patches of the specified size are cropped from both the LR and GT images. These cropped patches are then presented as the new samples.
Furthermore, we employ augmentation techniques such as flipping and rotation to induce generalization in the model. The input image and its corresponding GT are randomly flipped vertically or horizontally. The same process is followed for rotating the image.

For training purposes, the model is supplied with both low-resolution (LR) and high-resolution (HR) image pairs to learn the mapping from LR to HR images. The pixel values within the images are normalized to the range of [-1, 1]. Followed by the image

augmentations described above. Alternatively, for the testing phase, we perform the same preprocessing but skip the augmentation step.

Generator Network

The input image is first passed through a Conv2D Layer with a 9 x 9 kernel, stride = 1, and 64 feature maps attached to a PReLU activation. The image is then passed through 16 RDBs, each containing a Conv2D layer attached to the BatchNorm and PReLU layer. These blocks contain local residuals (skip connections) to each successive RDB. After this, the image again is passed through a Conv2D Layer with a 3 x 3 kernel , stride =1 and 64 feature maps attached with another batch norm layer. This is where the global residual is connected. The obtained image is then sent through an upsampling layer which increases the resolution of the input using sub-pixel convolution layers.

Discriminator Network

8 consecutive convolutional layers are used with an increasing number of 3 x 3 kernel layers which are increased by a factor of 2 starting from 64 ending at 512 kernels, inspired from the VGG network. The resulting feature maps are passed through 2 dense layers and a sigmoid activation which outputs real/fake classification.

The discriminator is learned using the standard adversarial min-max problem however the loss function for the generator is changed to a perceptual loss function instead of the standard MSE. The perceptual loss is defined as the weighted sum of content and adversarial loss where in the content loss is derived using the feature map present on the layer before the max pool layers in the VGG19 network and the standard adversarial loss of the generator is then added to it to compute the total perceptual content loss.

Pretraining with L2 Loss

The model undergoes an initial phase of training, commonly referred to as pre-training. This phase primarily focuses on minimizing the Mean Squared Error (L2 loss) between the generated high-resolution images and the actual high-resolution ground truths.

Fine-tuning with Perceptual and Adversarial Losses

After pre-training, the model enters a fine-tuning phase where it is refined further using a combination of losses: perceptual, adversarial, and total variance loss. Perceptual Loss is calculated using the VGG network by comparing features of the generated and real images. Total Variance Loss encourages smoothness in the generated images.

Hyperparameters Used

Residual Blocks (res_num): 16

Batch Size: 16

Loss Coefficients:

- L2 Loss Coefficient (L2_coeff): 1.0
- Adversarial Loss Coefficient (adv_coeff): 1e-3
- Total Variation Loss Coefficient (tv_loss_coeff): 0.0

Training Epochs:

- Pre-training Epochs (pre_train_epoch): 500/1500
- Fine-training Epochs (fine_train_epoch): 2000/4000

Super-resolution Scale (scale): 4

Patch Size: 24. During training, images were divided into patches of size 24x24 for effective processing.

Feature Layer (feat_layer): 'relu5_4'. This parameter indicates the specific layer from which features were extracted for perceptual loss calculations.

VGG Rescale Coefficient (vgg_rescale_coeff): 0.006

Model Paths:

- Default Generator Path: 'model/SRGAN_4000.pt'
- Alternative Generator Path: 'model/SRGAN_1500.pt'

Results and Discussions

In this study, we evaluated the performance of the SRGAN model on a spectrum of images, leveraging the PSNR metric to ascertain the similarity between the generated and the reference high-resolution images.

Analysis of the DIV2K Dataset

Subset Used: Our investigation started with a subset of the DIV2K dataset, specifically the DIV2K_valid_LR_bicubic. We compared its super-resolved images generated from our model with the corresponding high-resolution counterparts.

Observations:

High PSNR Values: We noted that certain images exhibited a high PSNR value, notably above 35. However, upon closer inspection, these images manifested reduced color variations, as evidenced by Figure 2, where many pixels demonstrated similar attributes. This suggests that simpler images with less variation can achieve high PSNR scores with ease.

Perceptual Richness vs. PSNR: Interestingly, images that were perceptually richer, encompassing a myriad of colors and details, generally resulted in lower PSNR scores, approximately around 22 dB (refer to Figure 3). It's essential to note that a PSNR score near 18 dB is typically perceived as inferior. This observation corroborates the fact that PSNR, being a pixel-wise error metric, might not always reflect human perception of image quality accurately. Images with a wealth of details or colors can be inherently challenging for super-resolution models. Subtle discrepancies in these visually rich regions can amplify pixel-wise errors, thereby affecting the PSNR scores, even if the images appear qualitatively superior to the human eye.

Analysis of the Set14 Dataset

- **Dataset Introduction**: Progressing from the DIV2K dataset, we further evaluated the model's performance on another benchmark dataset, Set14, as referenced in the original SRGAN paper.
- **Training Epoch Variation**: Our results, depicted in Figures 3 and 4, indicate that the number of training epochs plays a pivotal role in the model's performance. Specifically:
  - When trained for an extended duration (pre-training epochs = 1500, fine-train epochs = 4000), the resultant PSNR was commendably higher.
  - Conversely, reducing the pre-training epochs to 500 but increasing the fine-train epochs to 15000 didn't yield the same level of performance.

Super-Resolved (Output)
Size: (2040, 1356)
Pixels: 2766240
PSNR: 22.01 dB

Ground Truth

Super-Resolved (Output)
Size: (2040, 1128)
Pixels: 2301120
PSNR: 20.67 dB

Ground Truth

Super-Resolved (Output)
Size: (2040, 1356)
Pixels: 2766240
PSNR: 21.22 dB

Ground Truth

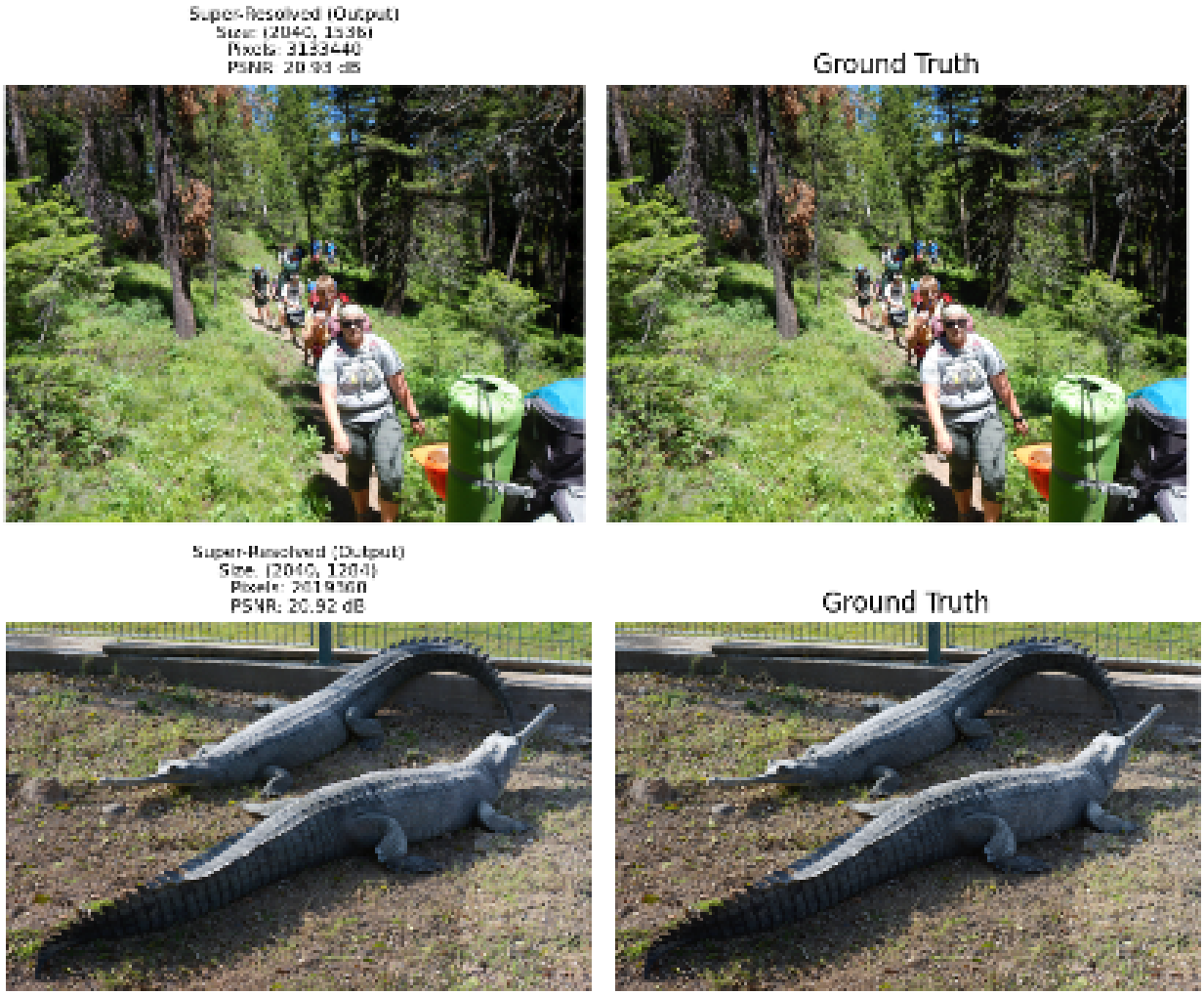
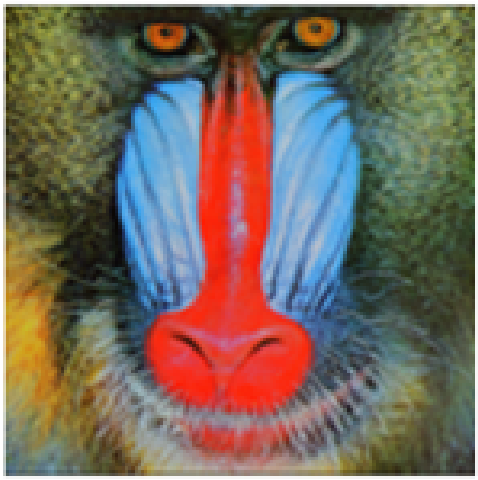
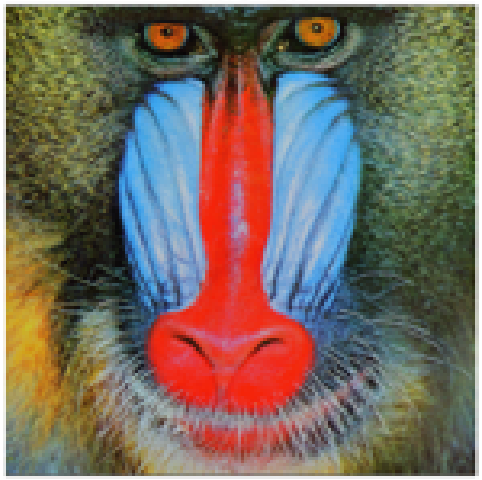


Figure 2 : Results: DIV2K validation dataset (bicubic). Perceptual and PSNR Comparison

Super-Resolved (Output)
Size: (1960, 1960)
Pixels: 3841600
PSNR: 20.75 dB



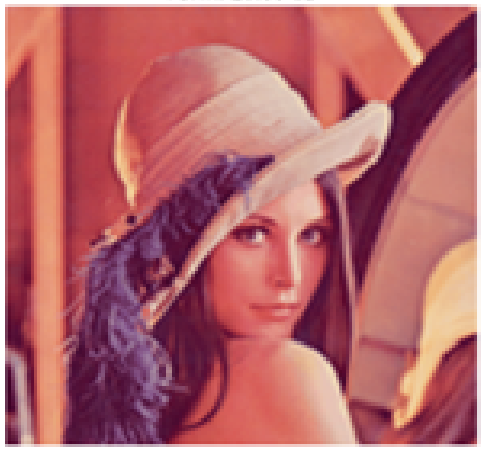
Ground Truth
Size: (490, 490)
Pixels: 240100



Super-Resolved (Output)
Size: (1960, 1960)
Pixels: 3841600
PSNR: 25.28 dB

Ground Truth
Size: (490, 490)
Pixels: 240100

Super-Resolved (Output)
Size: (1960, 1960)
Pixels: 3841600
PSNR: 28.68 dB



Ground Truth
Size: (490, 490)
Pixels: 240100

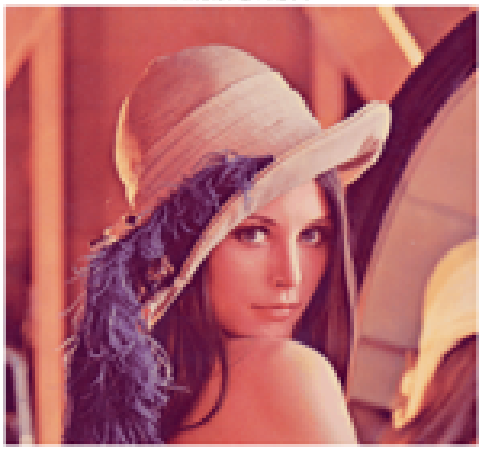
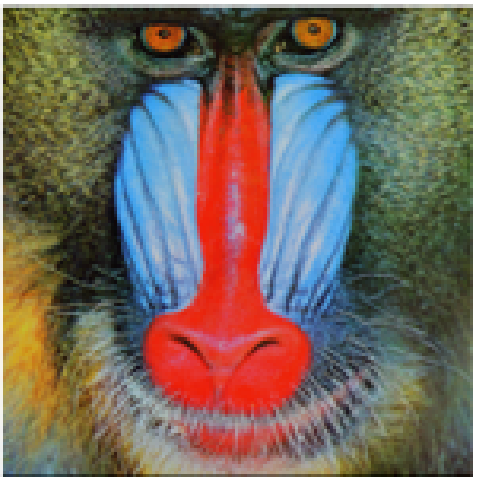


Figure 3: Results: Set14 dataset (Test Set). Perceptual and PSNR Comparison. Pretraining epochs = 1500, Finetraining epochs = 4000

In Figure 4, below, the same images as in Figure 3, however, it has lower PSNRs because of the lesser training.

Super-Resolved (Output)
Size: (1960, 1960)
Pixels: 3841600
PSNR: 21.58 dB

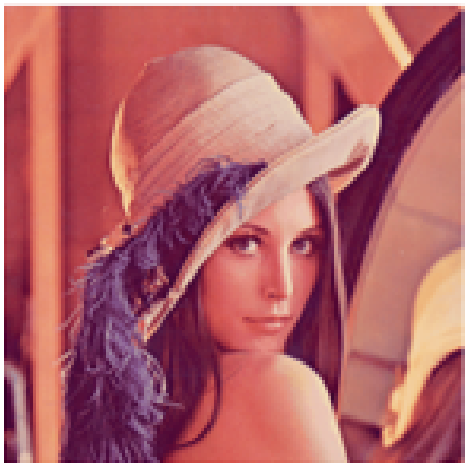


Ground Truth
Size: (490, 490)
Pixels: 240100

Super-Resolved (Output)
Size: (1960, 1960)
Pixels: 3841600
PSNR: 22.03 dB

Ground Truth
Size: (490, 490)
Pixels: 240100

Super-Resolved (Output)
Size: (1960, 1960)
Pixels: 3841600
PSNR: 30.51 dB



Ground Truth
Size: (490, 490)
Pixels: 240100

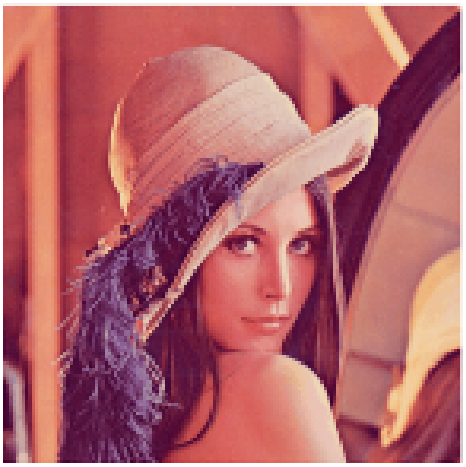


Figure 4: Results: Set14 dataset (Test Set). Perceptual and PSNR Comparison. Pretraining epochs = 500, Finetraining epochs = 1500

Areas of Uncertainty and Further Investigations

During the course of our study, several observations raised questions that we believe warrant further exploration:

Impact of Unseen Data on Super-Resolution

One intriguing observation was the model's response to low-resolution images, particularly those that were blurred. When the network encountered images it had not been trained on, the resultant super-resolved images were often still blurred. This raises pertinent questions:

Training Adequacy: Does the network require more extensive training to address such challenges effectively?

Data Familiarity: Would the model produce sharper super-resolved images if it had been exposed to similar blurred images during training?

Metric Limitations

PSNR Discrepancies: The referenced paper [5] reported an **average PSNR of 26.02 on Set 14**. In contrast, our results, even for blurry outputs, ranged between **20-26 PSNR** (as seen in Figure 5). This discrepancy underscores potential limitations in relying solely on PSNR as a quality metric.

Figure 5: Results: Set14 dataset (Test Set). Perceptual and PSNR Comparison. Pretraining epochs = 1500, Finetraining epochs = 4000. PSNR achieved for the blurry Image above is 26.34db, which is higher than average PSNR given in the paper(Refer: [5], Table 2[5])

Challenges

Hardware Constraints

Despite employing a high-performance a100 GPU, complemented by 32 CPU cores and 48 GB of RAM, the training durations were extensive. Specifically, for the model configuration outlined in Figure 2 (with 1500 epochs of pre-training and 4000 epochs of fine-tuning), it took an entire week to complete the training process.

Idle Time Overheads

A notable portion of this duration wasn't strictly related to computational processing. Periods when we couldn't access the GPU due to either operational needs or other queued tasks further extended the effective training time. Balancing the need for computational power with the availability of resources was a constant challenge.

Training Data Dilemma

During our research, we encountered a significant challenge regarding the choice of training data. Our final selection encompassed approximately 4.15 GB, equating to around 800 images, with an average size of 4.7 MB each. This extensive dataset, while offering diverse image representations, also presented potential limitations. We speculate that a smaller, more curated dataset might have potentially yielded improved results, emphasizing the principle that in some contexts, data quality can surpass sheer quantity.

Conclusion

1. While PSNR is commonly employed to measure image similarity, its values don't always reflect perceptual quality. For instance, in our DIV2K dataset analysis, images with fewer colors showed higher PSNR values, while detailed, perceptually-rich images scored lower.
2. The extent of training has a pronounced effect on model performance. As seen in our Set14 dataset evaluation, lengthier training sessions resulted in better PSNR outcomes, emphasizing the trade-off between training duration and output quality.
3. The model's response to unfamiliar or blurred images points to potential areas for deeper exploration. Such outcomes raise questions about the breadth of the model's training and its ability to adapt to novel data.
4. Our research journey was not without challenges. Despite using high-end GPUs, training durations were extensive. Additionally, the data selection process illuminated the balance between the size and quality of training datasets, suggesting that a more curated dataset might yield better results.
5. Comparing our findings with the referenced paper, it's clear that a more diverse set of evaluation metrics is needed. Sole reliance on PSNR may not offer a comprehensive assessment of a super-resolution model's capabilities.

Citations

[1] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang, "Learning a Deep Convolutional Network for Image Super-Resolution", ECCV, 2014.

[2] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, Yun Fu, "Residual Dense Network for Image Super-Resolution", CVPR, 2018.

[3] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn, "Fast, Accurate, and Lightweight Super-Resolution with Cascading Residual Network", ECCV, 2018.

[4] Tai, Y., Yang, J., Liu, X., & Xu, C. (2017). MemNet: A persistent memory network for image restoration. 2017 IEEE International Conference on Computer Vision (ICCV).

[5] Christian Ledig, Lucas Theis, Ferenc Husza r, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network", CVPR, 2017.

[6] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A.C., & Bengio, Y. (2014). Generative Adversarial Networks. arXiv: Machine Learning.

[7] https://github.com/dongheehand/SRGAN-PyTorch/tree/master