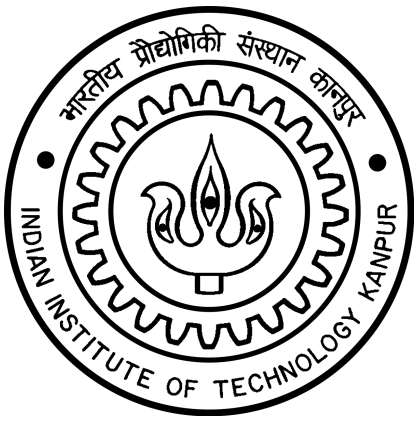


Indian Institute of Technology Kanpur

CS395A

Undergraduate Project - 1

Students:

Harish Rajagopal (160552)
Vishwas Lathi (160808)

Professor:

Vinay Namboodiri
Dept. of Computer Science and Engineering

Project Report

I. Abstract

We propose MAD-SRGAN which is a multi-agent generalisation to SRGAN [1]. SRGAN, a generative adversarial network [2] for image super-resolution is capable of generating photo-realistic natural 4× upscaled images.

SRGAN consists of a single discriminator which tries to discriminate whether generated images are realistic or not, while the generator tries to push the generated image towards the manifold of natural images. MADGAN [3] is a generalisation of GAN to address the problem of mode collapse, which most of the GAN's and their variants suffer. We propose a modification to vanilla SRGAN by combining MADGAN and SRGAN, namely MAD-SRGAN which consists of multiple generators and further variants also include multiple discriminators.

II. MAD-SRGAN

MAD-SRGAN consists of a 4 mini-generators each of which which deep residual network [4] with skip connections, as shown in fig. 2. Each residual block consists of two convolution layers with 3×3 kernel and 64 filters followed by batch-normalization [5] layer and ReLU as the activation function.

To discriminate whether the generated images lie close to natural images manifold, we have a discriminator network. The network uses the DCGAN [6] architecture. It consists of 8 convolution layers with 3×3 filter kernels. The feature maps obtained from the convolution layers are fed as input to two densely connected layers, which outputs the probability of classification using a sigmoid function. The network is trained using high-resolution images which are 4x downscaled and then fed as input to the model while the ground truth remains the upscaled images.

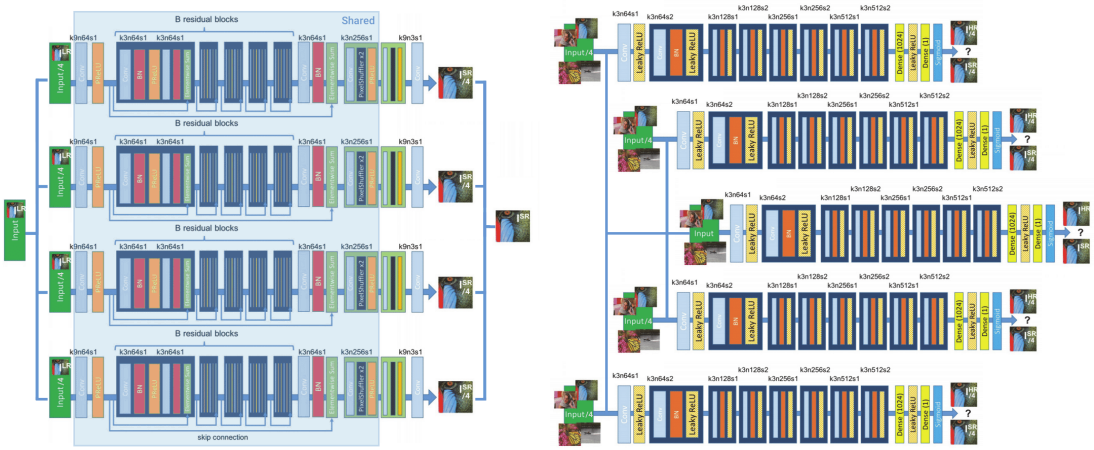
Figure 1: Overlap in outputs

The input image is divided into four equal and slightly overlapping sections, each corresponding to the top-left, top-right, bottom-left and bottom-right portion of the input image (see fig. 1). Each section is then fed to a mini-generator which tries to generate the corresponding part in the high-resolution target image. The output of all the generators is concatenated in the same fashion as the division of the input image. This output is then passed as an input to the discriminator which tries to identify whether the input image is fake or not.

Later on, in our experiments, we tried a multi-discriminator approach where the output of each mini-generator was fed to a mini-discriminator. The model consists of four mini-generators with sharing of parameters as defined earlier. Hence, the current model has four mini-discriminators, each trying to predict whether the section generated by the corresponding mini-generator, is fake or not. Then the outputs of all the mini-generators are combined in the same way as defined above and then given as input to a master discriminator which tries to compare the final upscaled image with the target and decides if it's real or not.

III. Losses

The loss function L consists of three parts, namely the content loss ($L_{content}$), adversarial



(1) Generators (2) Discriminators

Figure 2: MAD-SRGAN Architecture

loss (L_{adv}) and the overlap loss ($L_{overlap}$).

$$L = L_{content} + \alpha L_{adv} + \beta L_{overlap}$$

where α and β are hyperparameters.

The content loss is a perceptual loss [7] calculated on feature maps of the VGG19 network [8], similar to the one used by SRGAN. The adversarial loss is the adversarial loss for the generators by the discriminator, as used in SRGAN. The overlap loss is the MSE loss for the overlapping parts of the four generator outputs (see fig. 1), calculated as follows:

$$L_{overlap} = \sum_{i=1}^{3} \sum_{j=i+1}^{4} MSE(i, j)$$

where $MSE(i, j)$ is the MSE loss for the overlapping parts of images i and j.

In the multi-discriminator setup, the adversarial loss L_{adv} now consists of adversarial loss corresponding to each multi-discriminator and the adversarial loss for the master-discriminator.

$$L_{adv} = L_{adv}^{master} + \sum_{i=1}^{4} L_{adv}^{min\_disc\_i}$$

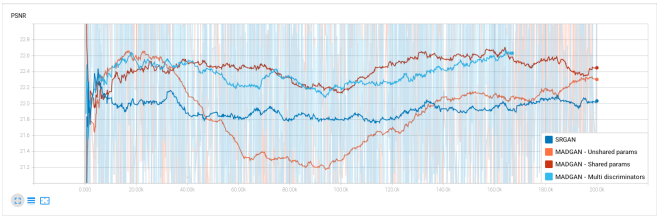
IV. Experiments

We ran our experiments using Tensorflow [9] v1.10. The system setup was a GPU server running Ubuntu 14.04.5 LTS, with a 12-core Intel Core i7 CPU and a single Nvidia GeForce GTX TITAN X (12GB) graphics card. The code was adapted from the GitHub repository SRGAN-tensorflow, which is our reference SRGAN implementation. The model was trained on the RAISE dataset [10] and the TESTIMAGES dataset [11, 12] was used for testing on aliasing patterns.

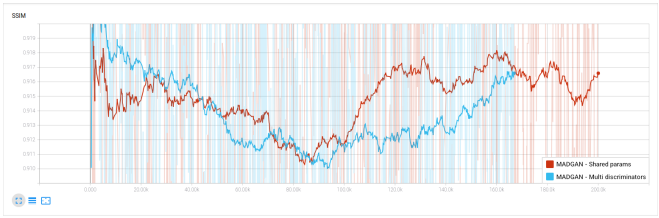
For image pre-processing, we applied random crop and random flipping to generate inputs of 24×24 pixels, with a 4x output. The optimiser used was Adam, with staircase decay for the learning rate. PSNR and multi-scale SSIM [13] were the additional metrics used for evaluating. Additionally, we also used images applied in aliasing tests as test images, to gauge the performance of the model concerning possible aliasing effects. Sample image outputs along with outputs for aliasing test images are shown in the appendix on page 5.

The values for our hyperparameters are:

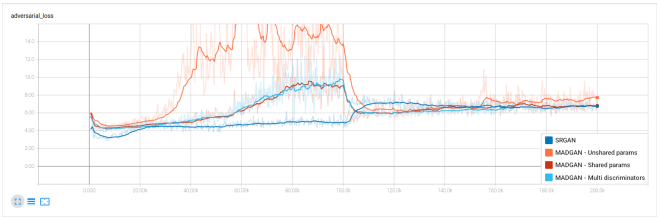
- Batch size: 16
- Residual blocks: 16
- VGG scaling in perceptual loss: 0.0061
- Adversarial loss weight (α): 0.001
- Overlap loss weight (β): 0.1
- Learning rate: 0.0001
- Learning rate decay rate: 0.1
- Learning rate decay after: 100000 steps
- Adam β_1 parameter: 0.9



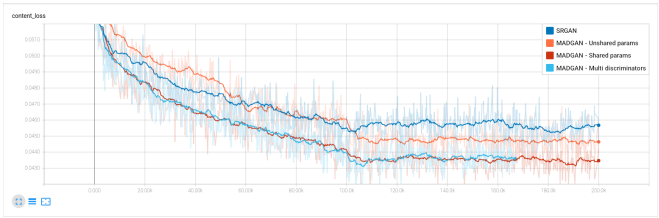
(1) PSNR



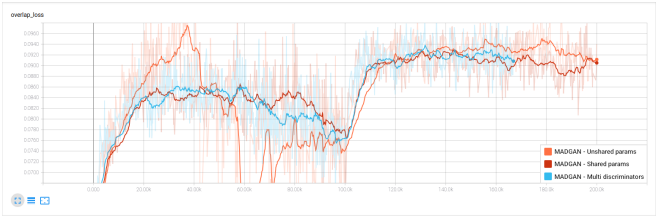
(2) SSIM



(3) Adversarial Loss



(4) Content Loss



(5) Overlap Loss

Figure 3: Results of MAD-SRGAN over SRGAN

- Maximum epochs: 200000

Results visualised via Tensorboard [9] (see fig. 3) show that MAD-SRGAN with no sharing of parameters showed poor performance as compared to SRGAN, resulting in noisy images. Enabling sharing of parameters in the lower layers resulted in a noticeable improvement over keeping parameters separate, and slight improvements over SRGAN. We think that this is because sharing results in the lowering of model parameters and thus helps in the stabilization of training those parameters. Further, this would help focus the optimizer on fine-tuning the higher layers of the generators as per their requirements and not focus too much on getting the more basic processing required by the lower layers right.

Addition of multiple discriminators in the form of one discriminator for each generator along with the global discriminator should help to tune the output of each generator further. However, we observed that this did not result in a noticeable improvement either in the losses, the metrics (PSNR or SSIM), or even when visually inspecting the output images.

References

[1] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," *CoRR*, vol. abs/1609.04802, 2016.

[2] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks," *ArXiv e-prints*, June 2014.

[3] A. Ghosh, V. Kulharia, V. P. Namboodiri, P. H. S. Torr, and P. K. Dokania, "Multi-agent diverse generative adversarial networks," *CoRR*, vol. abs/1704.02906, 2017.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[5] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by

reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.

[6] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR*, vol. abs/1511.06434, 2015.

[7] J. Johnson, A. Alahi, and F. Li, "Perceptual losses for real-time style transfer and super-resolution," *CoRR*, vol. abs/1603.08155, 2016.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[9] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[10] D.-T. Dang-Nguyen, C. Pasquini, V. Conotter, and G. Boato, "Raise – a raw images dataset for digital image forensics," March 2015.

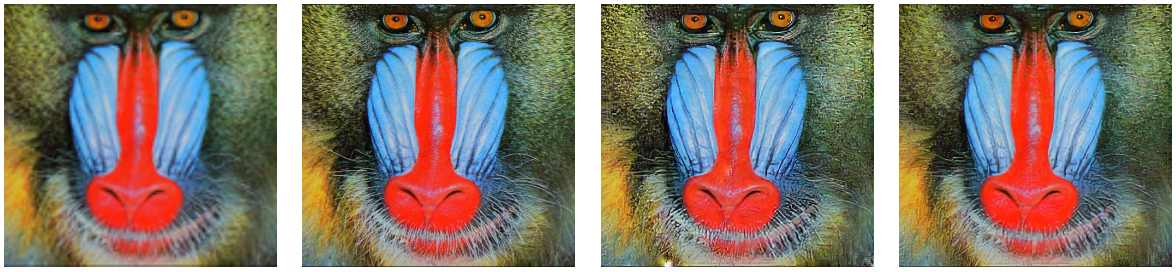
[11] A. N and G. A, "Testimages: A large data archive for display and algorithm testing," *Journal of Graphics Tools*, vol. 17, pp. 113–125, 2015.

[12] A. N and G. A, "Testimages: a large-scale archive for testing visual devices and basic image processing algorithms," in *STAG - Smart Tools & Apps for Graphics Conference*, 2014.

[13] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*, vol. 2, pp. 1398–1402 Vol.2, Nov 2003.

Appendix: Sample Images

Test Images



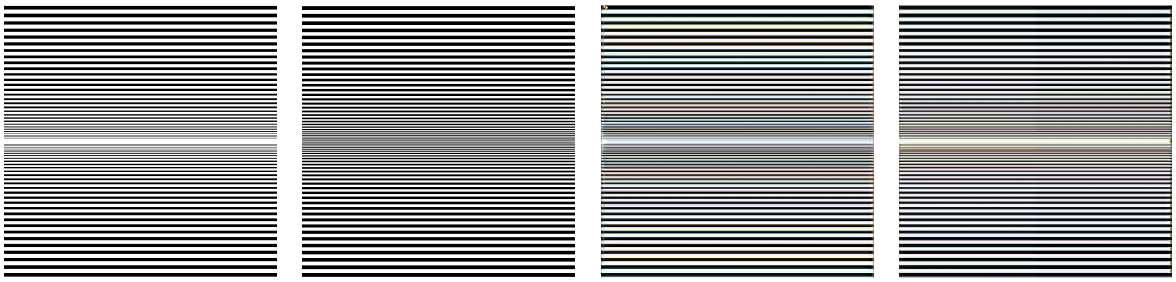
Input | Output | SRGAN | MAD-SRGAN

Input | Output | SRGAN | MAD-SRGAN

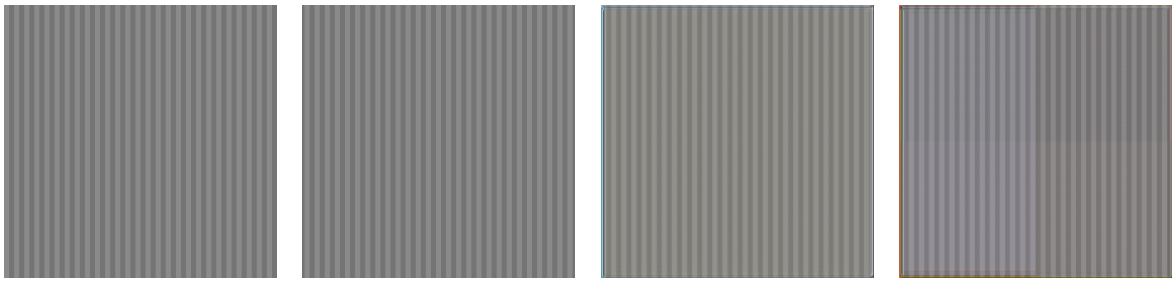
Input | Output | SRGAN | MAD-SRGAN

Input | Output | SRGAN | MAD-SRGAN

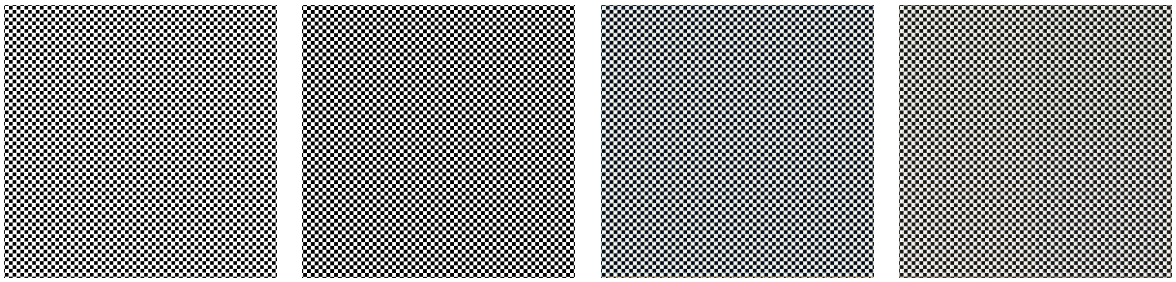
Aliasing Patterns



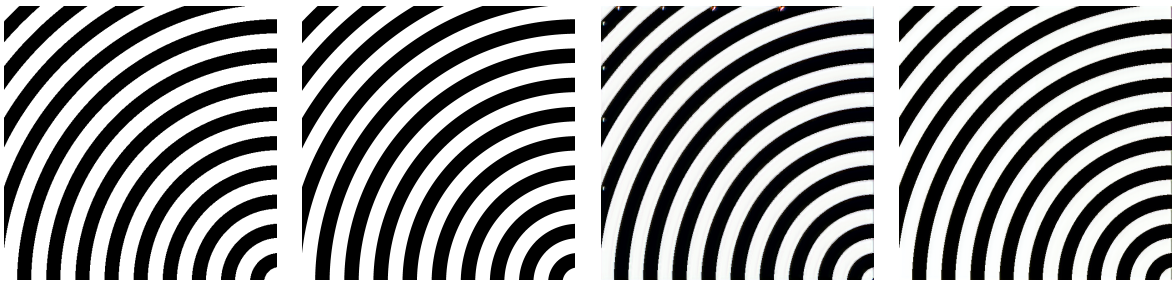
Input Output SRGAN MAD-SRGAN



Input Output SRGAN MAD-SRGAN



Input Output SRGAN MAD-SRGAN



Input Output SRGAN MAD-SRGAN