Study of SinGAN: Learning a Generative Model from a Single Natural Image

Khaoula Belahsen
ENS Paris-Saclay
ENSTA Paris
khaoula.belahsen@ensta-paris.fr

Aicha Boujandar
ENS Paris-Saclay
ENSTA Paris
aicha.boujandar@ensta-paris.fr

Abstract

*For this project, we have reviewed the reference article **SinGAN: Learning a Generative Model from a Single Natural Image** [3]. We will start by presenting the SinGAN approach presented in the reference article, and its specifities. We will then move to sanity check by comparing the results of the super resolution done by SinGAN with its results when done with two other state of the art algorithms : SRGan ([1] and [2]) and DeepPrior ([4]). Finally, we will present the results of our implementation of the denoising task.*

1. Generative Adversial Networks

Generative Adversial Networks is a very popular method for training a generative model. It is based on an adversial training between a discriminator and a generator which aims at fooling the discriminator and generate samples that cannot be detected as fake.

2. The SinGan model :

The SinGan model was developed by Google Brain team in TelAviv, and published in the paper [3] in September 2019. It aims at learning the patch distribution among different scales of the image in order to be able to generate new image samples that preserve the original patch distribution while creating new object configurations and structures. Its strength resides in the facts that:

- It is a single image training model : the authors of [3] assume that "single natural image typically carries enough information for learning a powerful generative model".
- It is an unconditional generative model : it does not require training on a data set to be applied to similar images.
- It can perform different tasks on images with the same architecture : Super resolution, editing, harmonization,...

2.1. Architecture

The architecture of SinGan consists of a pyramid of GANs : pairs of generators and discriminators. During the training, and a each scale of the pyramid, the generator takes as inputs a noise and the output of the previous generator, produces an image and tries to fool the discriminator which compares the fake image to a downsampled version of the real image. The first scale is purely generative since it takes only one input : a noise.

2.2. Loss function

The multi-scale architecture of SinGan is trained sequentially, from the coarsest scale to the finest one. Once each GAN is trained, it is kept fixed. The training loss for the nth GAN is comprised of an adversarial term and a reconstruction term :

$$min_{G_n} max_{D_n} \mathcal{L}_{adv}(G_n, D_n) + \alpha \mathcal{L}_{rec}$$

The adversarial loss $\mathcal{L}_{adv}$ penalizes for the distance between the distribution of patches in x_n and the distribution of patches in generated samples $\tilde{x}_n$. The reconstruction loss $\mathcal{L}_{rec}$ insures the existence of a specific set of noise that can produce x_n, which is an important feature for image manipulation [3].

3. Sanity check : Super Resolution

3.1. SinGan approach

Super resolution (SR) is the task of estimating a high resolution image (HR) from a low resolution image (LR). In [3], the authors present how the SinGan model can be used in the super resolution task. The idea consists on training the model on the LR image. Then, and at test time, upsample the LR image with a factor r, inject it (together with noise) in the the last generator G_0, and repeat this k times to get the HR image. We check the performance of the SinGan

approach by comparing it to two state of the art algorithms : SRGan and DeepPrior.

We have used and trained the sinGAN algorithm on the BSD100 data set LR images and then created the HR fake images with the existent code for super resolution on SinGan's github.

3.2. SRGan

The SRGan model was first presented in [1]. In order to apply it, we use the implementation presented in [2]. The architecture of SRGan relies on a generator G and a discriminator D. The idea is that the generator must produce a HR image from the LR image ($G(I^{LR})$) with an upscaling factor $\times 4$, and try to fool the discriminator that compares it to the real image (I^{HR}) during the training step. Thus, the loss function of the model is defined as follows :

$$min_{\theta_G} min_{\theta_D} \mathbb{E}_{I^{HR} \sim p_{train}(I^{HR})}[log(D_{\theta_D}(I^{HR}))] + \mathbb{E}_{I^{LR} \sim p_G(I^{LR})}[log(1 - D_{\theta_D}(G_{\theta_G}(I^{LR})))]$$

In the implementation proposed in [2], the model was trained on the DIV2K dataset. To be able to compare its results with SinGan, we have trained the model on the BSD100 dataset, with the parameter number of epochs = 400.

3.3. DeepPrior

In image restoration problems, the goal is to recover original image x from a corrupted picture x_0. These problems are formulated as follows :

$$min_x E(x; x_0) + R(x)$$

Where E is a data term and R(x) an image prior. The data term is different depending on the task (denoising, super-resolution...). The prior is captured by a Neural Network and more precisely a ConvNet.

Instead of searching for the answer in the image space, we search for it in the space of neural network's parameters. There is no use of a pretrained network or an image database. Only corrupted image x_0 is used in the restoration process.

3.4. Quantitative Results

We used 2 metrics to compare the 3 algorithms for the super resolution task : PSNR and SSID.

- PSNR : Peak to Noise Ratio, higher PSNR generally indicates that the reconstruction is of higher quality, in some cases it may not.

Metrics	Deep Prior	SrGAN	SinGAN
PSNR	23.28	21.83	21.85
SSIM	0.68	0.59	0.59

Table 1: Quantitative evaluation for the SR task

- SSIM : computes the structural similarity between two images, assumes that the clear image is identifiable. It better represents human visual perception than does PSNR. SSIM is more complex, however, and takes more time to calculate.

3.5. Qualitative Results

The results from these computations and experiments can be shown in the figures 1, 2 down below in the appendix.

4. The denoising task

We describe in tis section the two approaches we used to perform the denoising task using the SinGan model. The difference between them is that, in the first one, we train the model on the noisy image, and in the second we train the model on the real image. After training the model, we inject at the test step the noisy image at different scales (from the coarsest to the finest), we get the denoised images and we compare them quantitatively using the PSNR metric, and qualitatively via personal observation.

We perform the denoising task on images with gaussian noise of mean 10^3 and variance 100 (that we add via a python function) as shown in figure 4, salt and pepper noise (figure 5) and bad illumination conditions obtained using the PIL library in Python (figure 6).

4.1. Case 1 : Train on the noisy image

We train the SinGan model on the noisy image (like in the super resolution task), then we re-inject it in different scales during the test step.

We have started by re-injecting it only once and in one scale each time. The figure 7 shows the results for the gaussian noise. In terms of PSNR (computed with respect to the original image), injecting in the fifth scale gives the best results. Scales 4, 6 and 7 also give good results in terms of PSNR. For us, the injection in scale 5 gives the closer result to the original image of cows. For the salt and pepper noise (results figure 8, the experiment fails to produce a result similar or even close to the original image since the noise was not removed in any of the image. This can be also be noticed in the PSNR values that are small in comparison with their values in the denoising

task with gaussian noise. Same remarks can be done on the results of the experiment in case of bad illumination conditions (figure 9) where we can see that the models fails to reproduce an image with good illumination conditions.

We have also done another experiments by injecting it many times, and in one scale in each scale (like what was done in the super resolution task). The figure 10, shows the results for the denoising of the gaussian noise image. As we can see, the experiment fails to produce a denoised image although it has succeeded in the super resolution task.

This may be explained by the difference between the two tasks, since in SR, we inject an upsampled version of the LR image, here, we have injected the real noisy image in order to respect the sizes of the architecture, so, the problem is not treated in the same way by the model. Same for the salt and pepper noise results shown in 11 and for bad illumination noise shown in 12.

We have also done the experiments of injecting the noisy image in many consecutive scales, and by repeating the injection many times, and we have got the results in figure 13 for the gaussian noise, the ones in figure 14 for the salt and pepper noise, and the ones in figure 15 for the darkness noise. The scale represents the numbers of scales where we have injected the noisy image starting from the last scale. For example, scale=1 means that we have injected only in the last scale, scale=2 means that we have injected in the last scale and the scale before it... . We only represent the results of injection starting from the last scale since the other results are not better. As we can see in the figures, the experiment fails to reproduce a denoised image in the three types of noise.

Finally, we have done the experiments of injecting the noisy image in many consecutive scales, only once. The results shown in figure 16 correspond to the gaussian noise image, those in figure 17 correspond to the salt and pepper noise, and those in figure 18 correspond to the bad illumination noise. For the same reasons as in the previous experiment, we only represent the injection starting from the last scale since the other results are not better. As we can see in the results, the experiment has failed to reproduce a denoised image in all the types of noise.

We can conclude that training on the noisy image do not give good results in the case of salt and pepper noise and in the case of bad illumination conditions, but, we can say that it has succeeded in reproducing an enhanced version of the image in case of gaussian noise in the first experiment.

4.2. Case 2 : Train on the clear image

We have seen in the description of the SinGan algorithm that at training time, it learns how to reproduce patches and features from the input image through the pyramid of GANs architecture.

This was the intuition behind performing denoising by training the model in the clear image. Indeed, once the training done, the finest scales are expected to produce a fake clear image from some noise. Thus taking the patches of the noisy image and injecting them at finer scales must result in a clear image, and thus denoising.

We have done the experiments of injecting the noisy image and image patches in many consecutive scales. The results shown in figure 19 correspond to the gaussian noise image, those in figure 20 correspond to the salt and pepper noise, and those in figure 21 correspond to the bad illumination noise.

5. Conclusion

- Drawbacks :

  - No way to choose the injection scale prior to the results as the best scale is chosen by the user or we can do an optimization posterior to the experiments

  - Case 2 (train with clear image) : not feasible in practice as we don't have access to the clear image

  - Could be used in the case of bad illumination by using a good and bad illumination shots of the same spot

References

[1] Ferenc Huszar Jose Caballero Andrew Cunningham Alejandro Acosta Andrew Aitken Alykhan Tejani Johannes Totz Zehan Wang Wenzhe Shi Christian Ledig, Lucas Theis. Photo-realistic single image super-resolution using a generative adversarial network. (Submitted on 15 Sep 2016 (v1), last revised 25 May 2017 (this version, v5)).

[2] Hao Dong, Akara Supratak, Luo Mai, Fangde Liu, Axel Oehmichen, Simiao Yu, and Yike Guo. TensorLayer: A Versatile Library for Efficient Deep Learning Development. *ACM Multimedia*, 2017.

[3] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. *CoRR*, abs/1905.01164, 2019.

[4] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. *arXiv:1711.10925*, 2017.

6. Appendix : Figures

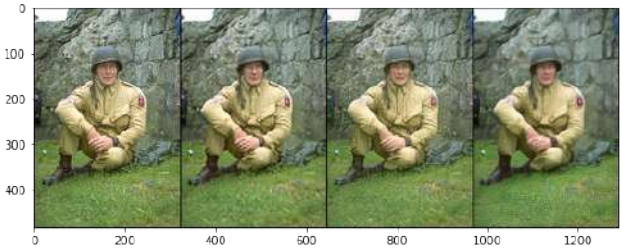

Figure 1: From left to right : Real image, deep prior, SinGAN, SrGan

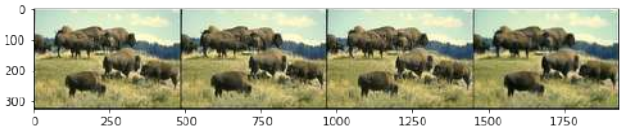

Figure 2: From left to right : Real image, deep prior, SinGAN, SrGan

Figure 3: Real image

Figure 4: Gaussian noise

Figure 5: Salt and pepper noise

Figure 6: Darkness noise

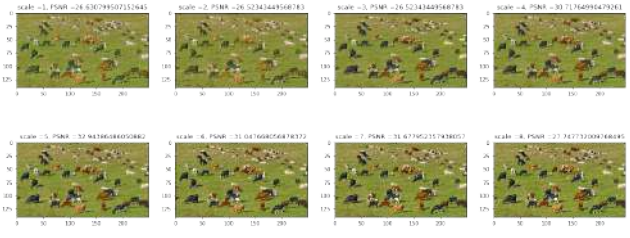

Figure 7: Denoising of the gaussian noise image : Train on the noisy image, different injection scales without loop

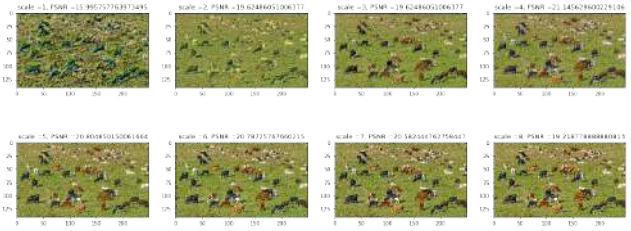

Figure 8: Denoising of the salt and pepper noise image : Train on the noisy image, different injection scales without loop

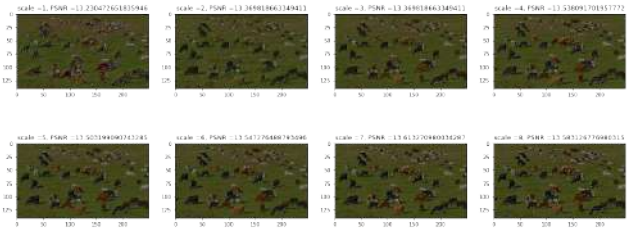

Figure 9: Denoising of the dark noise image : Train on the noisy image, different injection scales without loop

Figure 10: Denoising of the gaussian noise image : Train on the noisy image, different injection scales with loop

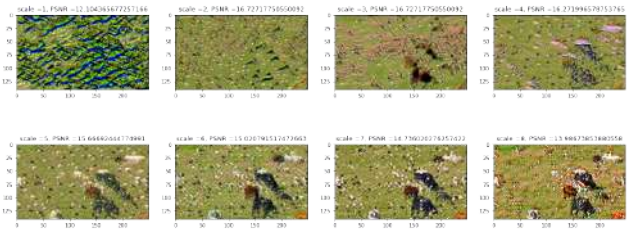

Figure 11: Denoising of the salt and pepper noise image : Train on the noisy image, different injection scales with loop

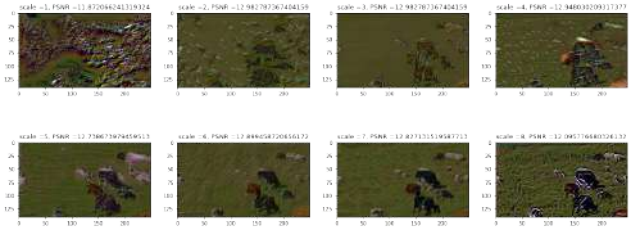

Figure 12: Denoising of the darkness noise image : Train on the noisy image, different injection scales with loop

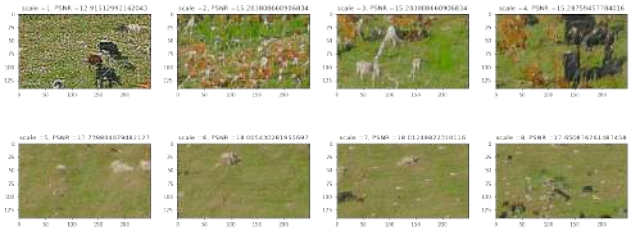

Figure 13: Denoising of the gaussian noise image : Train on the noisy image, injection in many scales with loop

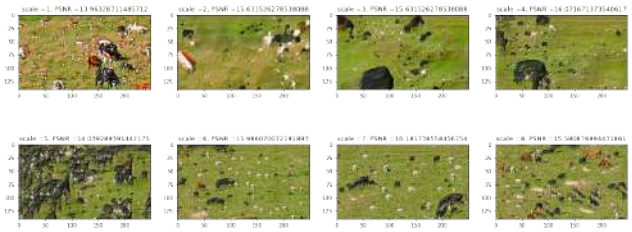

Figure 14: Denoising of the salt and pepper noise image : Train on the noisy image, injection in many scales with loop

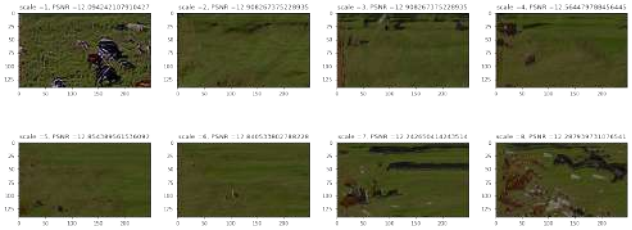

Figure 15: Denoising of the darkness noise image : Train on the noisy image, injection in many scales with loop

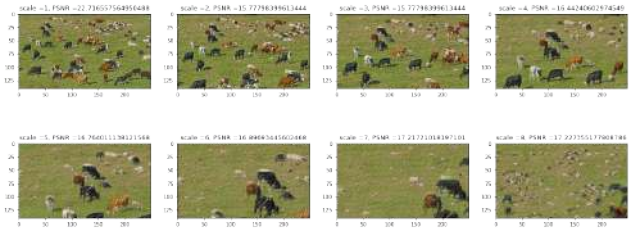

Figure 16: Denoising of the gaussian noise image : Train on the noisy image, injection in many scales without loop

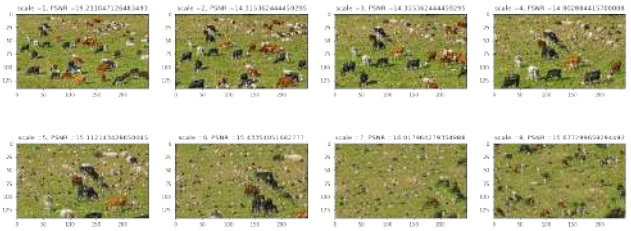

Figure 17: Denoising of the salt and pepper noise image : Train on the noisy image, injection in many scales without loop

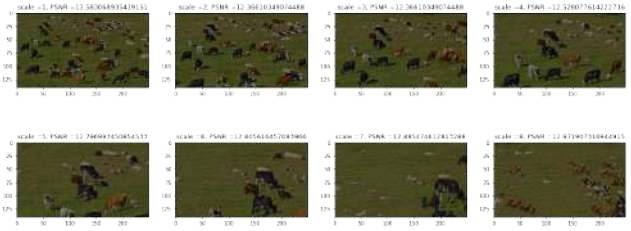

Figure 18: Denoising of the darkness noise image : Train on the noisy image, injection in many scales without loop

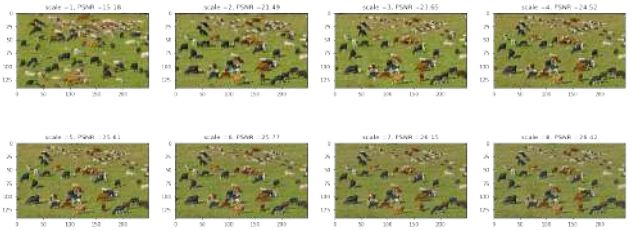

Figure 19: Denoising of the gaussian noise image : Train on the clear image, injection in many scales

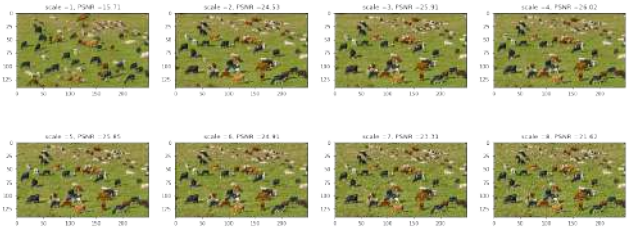

Figure 20: Denoising of the salt and pepper darkness noise image : Train on the clear image, injection in many scales

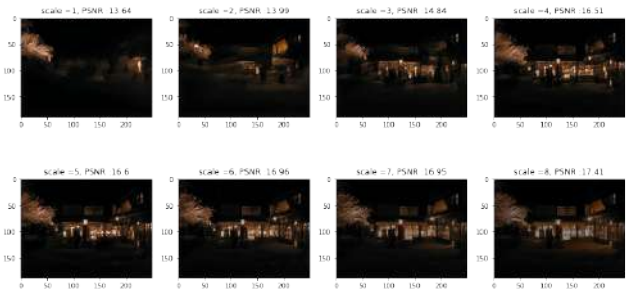

Figure 21: Denoising of the darkness noise image : Train on the clear image, injection in many scales