Single Image Super-Resolution via Iterative Refinement MVA Project

Mehdi Zemni
Centralesupélec-MVA
mehdi.zemni@student-cs.fr

Chady Raach
Centralesupélec-MVA
chady.raach@student-cs.fr

Abstract

Single-image super-resolution (or zoom) is a crucial problem in image restoration. The goal is to recover the high frequency information that has been lost through image downsampling and compression. Deep learning methods are now producing very impressive solutions to this problem. Numerous super-resolution methods have been proposed in the computer vision community. Much of the early work on super-resolution is regression based and trained with an MSE loss. As such, they effectively estimate the posterior mean, yielding blurry images when the posterior is multi-modal.

In this project, we will investigate a family of models called "denoising diffusion probabilistic models" (DDPM) which are nowadays of great interest for image generation. The goal of this project is to get familiar with this type of method and to understand how they are applied to super-resolution in this paper (SR3).

1. Introduction

Recent works on image generation task have led to enriching the state of the art method with several deep models such as Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), etc. Applied to conditional generation tasks such as image super-resolution, these models have shown convincing results achieving 31% of fool rate with FSRGAN [1] and PULSE [6] on $8\times$ face super-resolution task.

However, these models suffer from some limitations in terms of tractability and flexibility. For example, generative adversarial networks require special attention in regularisation terms and optimisation methods to avoid getting stuck in a small subspace of the complex real data, commonly referred to as "mode collapse". Tractability problems are often faced in likelihood based models such as variational autoencoders. For these models, a normalisation constant dependant on the model parameters is required to be computed and taken into account in backpropagation. Unfortunately, this constant is untractable and its approximation is computationally expensive.

Hence, these models, when applied to conditional generative tasks, were outperformed with more recent deep models inspired from Langevian dynamics such as score based generative models [10] and denoising diffusion probabilistic models [8].

In this project, the method of interest is the generation of high resolution images conditioned by low resolution image, using a model inspired from DDPM [3] and score matching models [10]. The paper [8] in particular is studied through this academic project. It proposes what called SR3 (Super-Resolution via Repeated Refinement) model which learns to transform a standard normal distribution into a a high resolution version of a given low resolution image, through a sequence of refinement steps widely inspired from Langevian dynamics. The proposed architecture is basically U-Net with minor modifications. Results are convincing since SR3 achieves a human fool rate close to 50% on standard $8\times$ face super-resolution task and proves effective on other tasks such natural images super resolution.

Section 2 is dedicated to showing how SR3 model is inspired from diffusion models and at which extend it shares theoritical basis with denoising score matching models. In section 3, implementation details are briefly presented. Finally, in section 4, the main result of the numerical experiment is presented and commented.

2. Conditional diffusion model

We are interested in building a model able to perform a stochastic iterative refinement process that maps a source image of low resolution x to a super resolution image y. We need therefore to learn a parametric approximation of the distribution $\mathbf{p}(y|x)$. This problem can be approached by learning to generate iteratively a sequence of vectors $(y_{T-1}, y_{T-2}, \ \dots \ , y_0)$ from a pure noise image $y_T \sim \mathcal{N}(0, \mathbf{I})$ according to learned conditional distributions $\mathbf{p}_\theta(y_{t-1}|y_t, x)$ in order to reach $y_0 \sim \mathbf{p}(y|x)$. This is

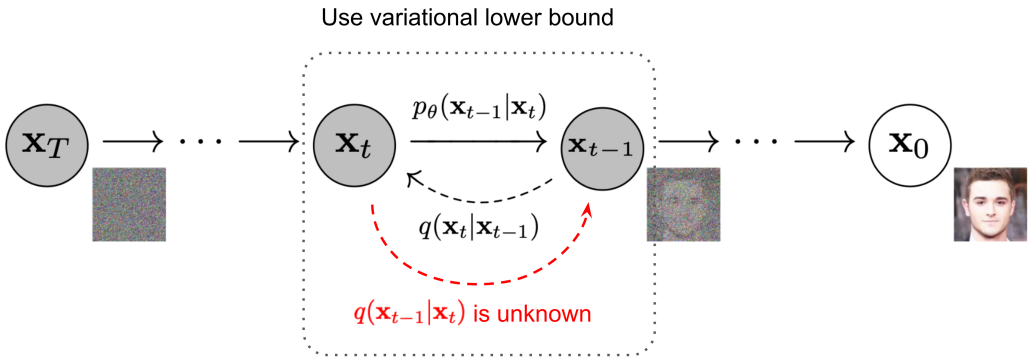


Figure 1: Black arrow: the Markov chain of forward diffusion process of generating a sample by adding gaussian noise. Red arrow: the Markov chain of reverse diffusion process of generating a sample by removing noise. (Source: [3])

done by adapting denoising diffusion probabilistic (DDPM) model to conditional generation. In what follows, the conditioning with respect to x is omitted for simplification matters. Nonetheless, it does not affect any of the statements as one can replace each probability by the conditional probability while keeping the same properties.

2.1. Forward diffusion process

In the forward diffusion process, which is performed during the training phase of the model, we add a small amount of Gaussian noise to a data point sampled from the real data distribution of high resolution images $y_0 \sim q(y)$ in T steps, producing a sequence of noisy samples $y_1, y_2, ..., y_T$. The step sizes are controlled by a variance schedule $\{\beta_t = 1 - \alpha_t \in (0,1)\}_{t=1}^{t=T}$

$$q(y_{1:T}|y_0) = \prod_{t=1}^{T} q(y_t|y_{t-1}) \tag{1}$$

$$q(y_t|y_{t-1}) = \mathcal{N}(y_t, \sqrt{\alpha_t}y_{t-1}, (1-\alpha_t)\mathbf{I}) \tag{2}$$

The larger the step t becomes the more the sample y_0 loses its distinguishable features. Particularly, when $T \to \infty$, $y_T \sim \mathcal{N}(0, \mathbf{I})$. Figure 1 shows how this process is conducted.

In practice, we don't need all intermediate steps to sample y_t at a time step t. The distribution of y_t given y_0 can be expressed as:

$$q(y_t|y_0) = \mathcal{N}(y_t, \sqrt{\gamma_t}y_0, (1-\gamma_t)\mathbf{I}) \tag{3}$$

where $\gamma_t = \prod_{i=1}^{t} \alpha_i$

2.2. Reverse diffusion process

The goal is to reverse the gaussian diffusion process by learning a neural network model p_θ that takes the noisy image y_t as input and estimates y_{t-1} (black arrow in figure 1). This will allow to build y_0 from a random vector $y_T \sim \mathcal{N}(0, \mathbf{I})$. For image super-resolution, the network will be conditioned on a second input which is the low resolution source image and will estimate the noise instead, by reparameterizing the training.

We will prove in this section that we can learn such a model.

First, we note for each t, $\beta_t = 1 - \alpha_t$, and define $z_t \sim \mathcal{N}(0, \mathbf{I})$ such that

$$\begin{aligned} y_t &= \sqrt{\alpha_t}y_{t-1} + \sqrt{1-\alpha_t}z_{t-1} && (4)\\ &= \sqrt{\alpha_t\alpha_{t-1}}y_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\overline{z}_{t-2} && (5)\\ &= ... \\ &= \sqrt{\gamma_t}y_0 + \sqrt{1-\gamma_t}z && (6) \end{aligned}$$

.

Equation 5 is given by the fact that the sum of two gaussians $\mathcal{N}(0, \sigma_1^2\mathbf{I})$ and $\mathcal{N}(0, \sigma_2^2\mathbf{I})$ is a gaussian $\mathcal{N}(0, (\sigma_1^2 + \sigma_2^2)\mathbf{I})$.

On the other hand, it is useful to condition the probability with respect to y_0:

$$q(y_{t-1}|y_t, y_0) = \mathcal{N}(y_{t-1}; \mu(y_t, y_0), \Sigma)$$

and using Bayes' rule:

$$\begin{aligned} q(y_{t-1}|y_t, y_0) &= q(y_t|y_{t-1}, y_0)\frac{q(y_{t-1}|y_0)}{q(y_t|y_0)} && (7)\\ &\propto \exp(-\frac{1}{2}\frac{(y_t - \sqrt{\alpha_t}y_{t-1})^2}{\beta_t}) \\ &\times \exp(-\frac{1}{2}\frac{(y_{t-1} - \sqrt{\gamma_{t-1}}y_0)^2}{1-\gamma_{t-1}}) \\ &\times \exp(\frac{1}{2}\frac{(y_t - \sqrt{\gamma_t}y_0)^2}{1-\gamma_t}). && (8) \end{aligned}$$

By developing equation 8 in y_{t-1}, we come to show that:

$$\mu(y_t, y_0) = \frac{\sqrt{\alpha_t}(1-\gamma_t)}{1-\gamma_t} y_t + \frac{\sqrt{\gamma_t}\beta_t}{1-\gamma_t} y_0 \tag{9}$$

$$\Sigma = \frac{1-\gamma_{t-1}}{1-\gamma_t}\beta_t \tag{10}$$

Finally, combining equations 6 and 9 gives:

$$\mu(y_t, t) = \frac{1}{\sqrt{\alpha_t}}(y_t - \frac{\beta_t}{\sqrt{1-\gamma_t}} z_t) \tag{11}$$

Thus, the learnable parameter is $\mu_\theta(y_t, t)$ and secondarily Σ_θ in case we want to parameterize the variance matrix to have more stable training as shown by [7].

2.3. Reparameterizing the learnable function

Since y_t is available as input at training time, we can reparameterize the problem to make the model predict z_t instead from the input y_t and step t:

$$\mu_\theta(y_t, t) = \frac{1}{\sqrt{\alpha_t}}(y_t - \frac{\beta_t}{\sqrt{1-\gamma_t}} z_\theta(y_t, t)) \tag{12}$$

In the case of conditional diffusion models, $_\theta$ will take another argument which is the low resolution image x. In practice, instead of conditioning on t, authors suggest conditioning on a stochastic version of γ_t ($:= \gamma \sim p(\gamma)$). The proposed objective function for training μ_θ is:

$$\mathbb{E}_{(x,y)}\mathbb{E}_{\epsilon,\gamma}\left\| f_\theta(x, \sqrt{\gamma} y_0 + \sqrt{1-\gamma} z, \gamma) - z \right\|_p^p \tag{13}$$

where $z \sim \mathcal{N}(0, \mathbf{I})$, (x,y) pair of inputs sampled from the dataset (LR and HR pairs), $p \in \{1, 2\}$. For the noise scheduling, we first uniformly sample a time step $t \sim \{0, 1, ..., T\}$ and then sample $\gamma \sim U(\gamma_{t-1}, \gamma_t)$. By using this technique of conditioning on γ instead of t, we have more flexibility in choosing number of steps T and noise schedule during inference ($\gamma_t = f(t)$) (the only remaining constraint is to use the same values for β_1 and β_T).

The above objective function is actually derived from the negative variational lower bound up to a constant weighting of each term for each time step and it was shown that in practice, training the diffusion model works better with simplifying the weighting term [3]

2.4. Origin of the loss function

The loss we introduced in the previous section comes from the variational lower bound which can be used to optimize the negative log-likelihood. In the unconditional case, this can be written as:

$$-\log p_\theta(\mathbf{y}_0) \leq -\log p_\theta(\mathbf{y}_0) + D_{\mathrm{KL}}(q(\mathbf{y}_{1:T}|\mathbf{y}_0) \| p_\theta(\mathbf{y}_{1:T}|\mathbf{y}_0)) \tag{14}$$

$$= -\log p_\theta(\mathbf{y}_0) \tag{15}$$

$$+ \mathbb{E}_{\mathbf{y}_{1:T}\sim q(\mathbf{y}_{1:T}|\mathbf{y}_0)}\Big[\log \frac{q(\mathbf{y}_{1:T}|\mathbf{y}_0)}{p_\theta(\mathbf{y}_{0:T})/p_\theta(\mathbf{y}_0)}\Big] \tag{16}$$

$$= -\log p_\theta(\mathbf{y}_0) + \mathbb{E}_q\Big[\log \frac{q(\mathbf{y}_{1:T}|\mathbf{y}_0)}{p_\theta(\mathbf{y}_{0:T})} + \log p_\theta(\mathbf{y}_0)\Big] \tag{17}$$

$$= \mathbb{E}_q\Big[\log \frac{q(\mathbf{y}_{1:T}|\mathbf{y}_0)}{p_\theta(\mathbf{y}_{0:T})}\Big] \tag{18}$$

$$\text{Let } L_{\mathrm{VLB}} = \mathbb{E}_{q(\mathbf{y}_{0:T})}\Big[\log \frac{q(\mathbf{y}_{1:T}|\mathbf{y}_0)}{p_\theta(\mathbf{y}_{0:T})}\Big] \geq -\mathbb{E}_{q(\mathbf{y}_0)} \log p_\theta(\mathbf{y}_0) \tag{19}$$

Following [9], we can convert each term in the equation to be analytically computable:

$$L_{\mathrm{VLB}} = \mathbb{E}_{q(\mathbf{y}_{0:T})}\Big[\log \frac{q(\mathbf{y}_{1:T}|\mathbf{y}_0)}{p_\theta(\mathbf{y}_{0:T})}\Big] \tag{20}$$

$$= \mathbb{E}_q\Big[\log \frac{q(\mathbf{y}_T|\mathbf{y}_0)}{p_\theta(\mathbf{y}_T)} + \sum_{t=2}^{T} \log \frac{q(\mathbf{y}_{t-1}|\mathbf{y}_t, \mathbf{y}_0)}{p_\theta(\mathbf{y}_{t-1}|\mathbf{y}_t)} \tag{21}$$

$$- \log p_\theta(\mathbf{y}_0|\mathbf{y}_1)\Big] \tag{22}$$

$$= \mathbb{E}_q[\underbrace{D_{\mathrm{KL}}(q(\mathbf{y}_T|\mathbf{y}_0) \parallel p_\theta(\mathbf{y}_T))}_{L_T} \tag{23}$$

$$+ \sum_{t=2}^{T} \underbrace{D_{\mathrm{KL}}(q(\mathbf{y}_{t-1}|\mathbf{y}_t, \mathbf{y}_0) \parallel p_\theta(\mathbf{y}_{t-1}|\mathbf{y}_t))}_{L_{t-1}} \tag{24}$$

$$\underbrace{-\log p_\theta(\mathbf{y}_0|\mathbf{y}_1)}_{L_0}] \tag{25}$$

The above equation uses KL divergence to directly compare $p_\theta(y_{t1}|y_t)$ against forward process posteriors, which are tractable when conditioned on y_0.

Every KL term in L_{VLB} (except for L_0) compares two Gaussian distributions and therefore they can be computed in closed form. L_T is constant and can be ignored during training because q has no learnable parameters and x_T is a Gaussian noise.

To present the mean $\mu_\theta(x_t, t)$, [3] propose the following parameterization of the loss L_t. With $p_\theta(\mathbf{y}_{t-1}|\mathbf{y}_t) = \mathcal{N}(\mathbf{y}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{y}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{y}_t, t))$, we can write:

$$L_t = \mathbb{E}_{\mathbf{y}_0,\mathbf{z}}\Big[\frac{1}{2\|\boldsymbol{\Sigma}_\theta(\mathbf{y}_t,t)\|_2^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{y}_t,\mathbf{y}_0)-\boldsymbol{\mu}_\theta(\mathbf{y}_t,t)\|^2\Big]$$
$$= \mathbb{E}_{\mathbf{y}_0,\mathbf{z}}\Big[\frac{\beta_t^2}{2\alpha_t(1-\gamma_t)\|\boldsymbol{\Sigma}_\theta\|_2^2}\|\mathbf{z}_t-\mathbf{z}_\theta(\mathbf{y}_t,t)\|^2\Big]$$
$$= \mathbb{E}_{\mathbf{y}_0,\mathbf{z}}\Big[K_{\theta,t}\|\mathbf{z}_t-\mathbf{z}_\theta(\sqrt{\gamma_t}\mathbf{y}_0+\sqrt{1-\gamma_t}\mathbf{z}_t,t)\|^2\Big]$$

with $K_{\theta,t} = \frac{\beta_t^2}{2\alpha_t(1-\gamma_t)\|\boldsymbol{\Sigma}_\theta\|_2^2}$ Empirically, [3] found that training the diffusion model works better with a simplified objective that ignores the weighting term $K_{\theta,t}$.

To summarize, we can train the reverse process mean function approximator μ_θ to predict $\tilde{\boldsymbol{\mu}}_t$, or by modifying its parameterization, we can train it to predict z_t.

2.5. Connection with score based generative models

Both noise-conditioned score networks (NCSN) and denoising diffusion probabilistic models (DDPM) bypass the problem of untractable normalisation constant. On one hand, NCSN learn $\nabla_y \mathbf{p}(y)$ which is independent from the normalisation term $Z_\theta = \int \mathbf{p}_\theta(y)dy$. This solves the problem of tractability faced by likelihood based models in image generation task. On the other hand, DDPM map a distribution of noisy data to a higher quality data by only learning the mean and the variance of the distribution as shown in section 2.2.

The two families of models are used to data generation and particularly conditional image generation such as super resolution. Both are based on an iterative decoding process. While DDPM iteratively reverse a diffusion process starting from a random gaussian vector in order to obtained a high quality data point, NCSN are based on an iterative process explicitly related to Langevian dynamics. This family of models learn $\mathbf{s}_\theta(y)$ to approximate the gradient of the density function $\mathbf{p}(y)$ (or $\mathbf{p}(y|x)$ in case of conditional generation). The repetitive process is given by equation 26

$$y_t = y_{t-1} + \epsilon\nabla_y \mathbf{p}(y) + \sqrt{2\epsilon}z \tag{26}$$

where y_0 is sampled from an arbitrary prior $\pi(y)$ and $z \sim \mathcal{N}(0,\mathbf{I})$. When $T \to \infty$, y_T and $\epsilon \to 0$, y_T converges to a sample drawn from $\mathbf{p}(y)$.

Finally, these models are trained with multiple noise perturbation which can be viewed as a discretisation of the stochastic differential equations that rules the chained process. This allows to populate low density regions of the training data in the case of NCSN as shown in Figure 2 and leads to better results, especially to avoid the generation of unrealistic samples. For diffusion models, training a multiple noise level decoder allows to cascade the same trained network at different levels of the reverse chain, with a decreasing noise level. Consequently, a deep model for conditioned generation (as it is the case of super resolution task) is defined as $\mathbf{f}_\theta(y_t, x, \sigma_t)$ where y_t is the input noisy vector at step t, x is the input vector that conditions the probability model and σ_t is the estimated noise level to be removed from y_t. The output is nothing other then y_{t-1} a cleaner version of y_t, knowing x. $\mathbf{f}_\theta$ is applied multiple times until a final result is obtained in a finite number of iterations.

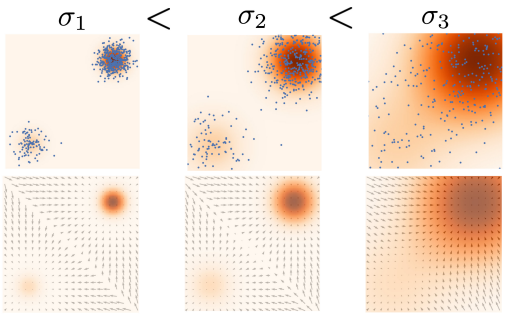


Figure 2: Score space of a trained score base network with different noise scales.

3. Implementation details

The architecture used in SR3 is a modified U-net architecture based on a wide ResNet which is the same architecture used in DDPM [3] with some minor modifications. Self-attention blocks are used between convolutional blocks in low resolution features (16x16). To condition the model on the input x, we up-sample the low-resolution image to the target resolution using bicubic interpolation. The result is concatenated with y_t along the channel dimension (Figure 3). And to condition on γ, the unofficial implementations we used for this project, uses FiLM :Feature-wise Linear Modulation layer for embedding wheras DDPM [3] uses the Transformer sinusoidal position embedding into each residual block (cte attention is all you need).

We used an unofficial implementation of SR3 `https://github.com/Janspiry/Image-Super-Resolution-via-Iterative-Refinement` which we modified a little by adding a new baseline model based on the same architecture of SR3 (UNet). This baseline is a one-step regression model that is trained with a reconstruction loss (l_1 or MSE). Usually, we don't train such models with a MSE loss only as it yields poor results because these models will effectively estimate the posterior mean, yielding blurry images when the posterior is multi-modal. Other combination of losses are usually used in this kind of scenarios such the perceptual loss [4]. But we will stick with the MSE loss since it is only a baseline.

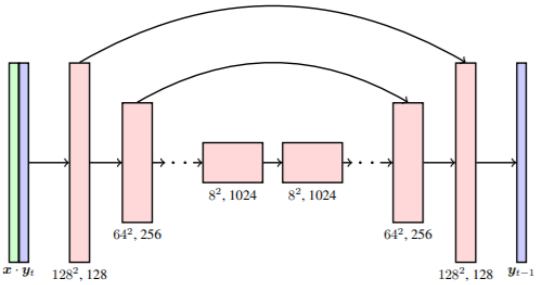


Figure 3: Description of the U-Net architecture. The low resolution input image x is interpolated to the target high resolution, and concatenated with the noisy high resolution image y_t.

Algorithm 1 Training

repeat
- $(x, y_0) \sim p(x, y)$
- $t \sim Uniform(1, .., T)$
- $\gamma \sim Uniform([\gamma_{t-1}, \gamma_t])$
- $z \sim \mathcal{N}(0, \mathbf{I})$
- Take a gradient step
- $\nabla_\theta \|\mu_\theta(x, \sqrt{\gamma} y_0 + \sqrt{1-\gamma} z, \gamma) - z\|_p^p$

Algorithm 2 Inference in T refinement steps, given LR image x

$y_T \sim \mathcal{N}(0, \mathbf{I})$
for t = 1, .., T **do**
- $z \sim \mathcal{N}(0, \mathbf{I})$
- $y_{t-1} = \frac{1}{\sqrt{\alpha}}(y_t - \frac{1-\alpha_t}{\sqrt{1-\gamma_t}} \mu_\theta(x, y_t, \gamma_t) + \sqrt{1-\alpha_t} z$

4. Experiments

In this section we will provide some experimental results based on our trainings and inferences of the SR3 model.

4.1. Experiment on the model trained by the authors

We started our experiments with few inference runs on the model trained by the authors on the FacesHQ dataset. We identified lack of generalization problems in the trained model that were seen through:

- Overall, the model generates the same outcome when conditioned on the same low resolution image. Even though, this ensures about the consistency of the model, we expected that the upscaling of an image by two independent experiences would give different outcomes, since the process is stochastic and dependent on a random realisation at the beginning of the reverse diffusion process.

Reference Bicubic SR3

Figure 4: Generalisation issue of SR3 and bias introduced by the conditioning

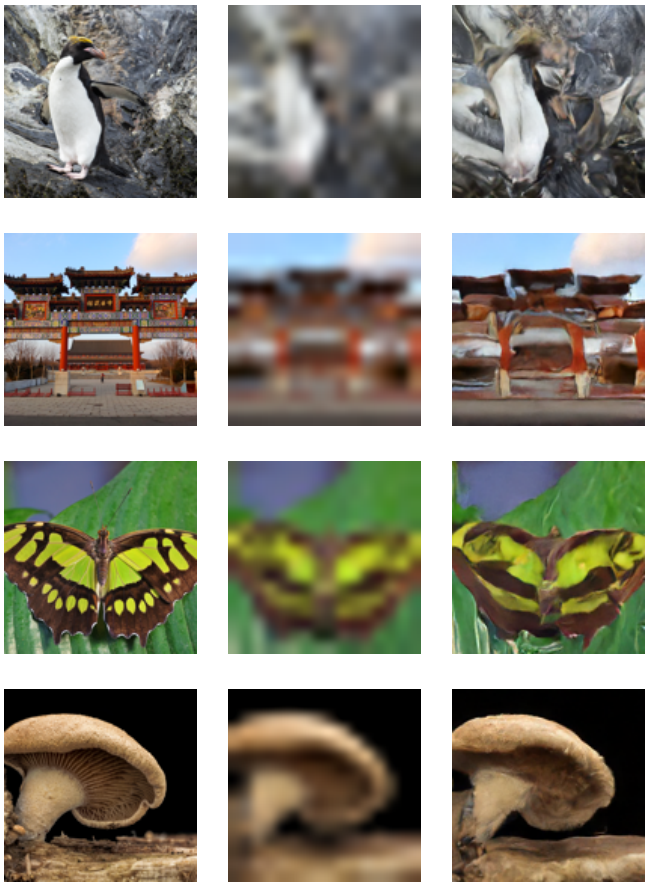
- The model is sensitive to small noises such as borders, position of the face in the image, and saturation of the image in some colors. Data augmentations (e.g. flipping) were introduced by the authors, but they seem insufficient to generalise better.

To illustrate the bias induced by the saturation of some colors, Figure 4 shows that in the first experiment the presence of shades of blue around the eyes of the LR image led to a blueish reconstructed image and a poor recovering of the green eye of the character. In a second experiment, we added a black mark near the area responsible for the bias. The tone of colors in the reconstruction becomes more usual and closer to the reference.

4.2. Training a new model on DIV2K dataset

First, we tried to train a model on the DIV2K dataset which is a common dataset used to train super-resolution networks. This dataset consists in images with a large diversity of classes. On this dataset we didn't manage to get interesting results comparable to the results obtained on faces. We think that conditional generative models for super-resolution like SR3 which are not trained directly with a reconstruction loss on images fail on this kind of tasks since it is harder for them to learn the distribution of a small dataset (800 training images only) containing large diversity of classes.

The trained model on the train set generalized poorly to the test set and results of PSNR (Peak signal-to-noise ratio) and SSIM (structural similarity index measure) were relatively low when compared to state of the art results or even the results the author obtained on the CelebA-HQ dataset [5].



Reference Bicubic SR3

Figure 5: SR3 applied to 4 test images from DIV2K dataset

In Figure 5, we show some results of super-resolution (16x16 $\rightarrow$ 128x128) on the test set.

We also measured the mean PSNR and mean SSIM on 50 images from test set and compared the results with the bicubic interpolation. Values of PSNR and SSIM are written in the table below:

	PSNR	SSIM
SR3	17,67	0.36
Bicubic	21.7	0.59

4.3. Training a new model on (AFHQ)-dog dataset

In our second attempt, we tried the Animal FacesHQ (AFHQ)-dog dataset which contains around 5k high-quality images at 512 × 512 resolution. Because of our limited computation capacity, we opted for a small architecture ($16 * 16 \rightarrow 128 * 128$).

We used these hyperparameters for the training:

L_1 loss, T=2000 iteration, linear noise scheduler, with parameters $\beta_1 = 10^{-6}, \beta_T = 10^{-2}$, Optimizer: Adam and a constant learning rate of 10^{-5}.

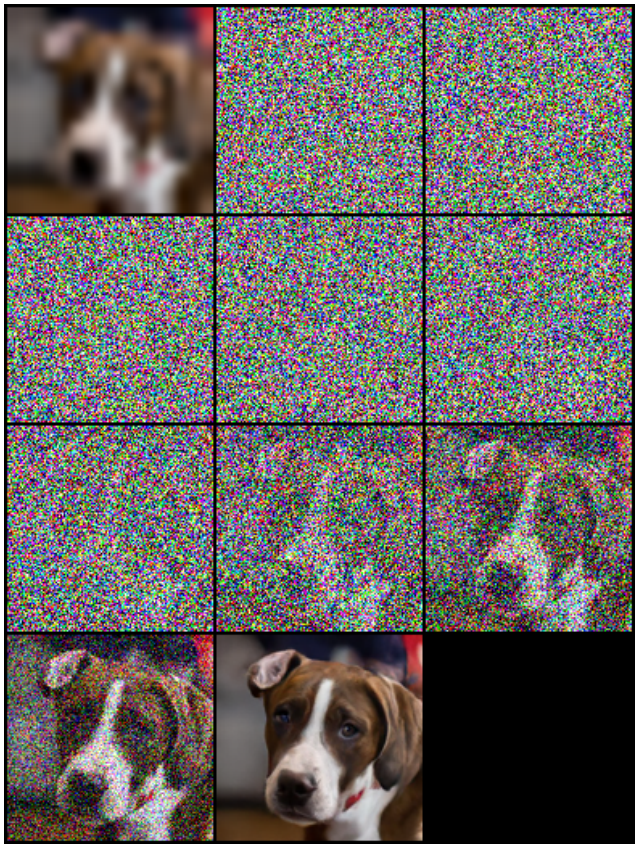


Figure 6: Sampling.

In Figure 6, we detail the iterative process of upscaling and image from the test set. In Figure 7, we show the upscaling results of several image from the test set, starting from a bicubic interpolated images.

We measure the mean PSNR and SSIM of 50 superresolution images from both train and test set. These values are reported in the table below:

	PSNR	SSIM
Bicubic	22.08	0.57
Resgression	**24.26**	**0.68**
SR3	23,05	0.62

These quality scores (PSNR and SSIM) often penalize synthetic high-frequency details, such as hair texture, because synthetic details do not perfectly align with the reference details. This can be observed in Figure 8 where we see that images obtained with the regression based model are blurry but images with the SR3 model have more high frequency details and look more natural and perceptually closer to the reference images even though the PSNR is lower on images generates by the SR3. Therefore, we

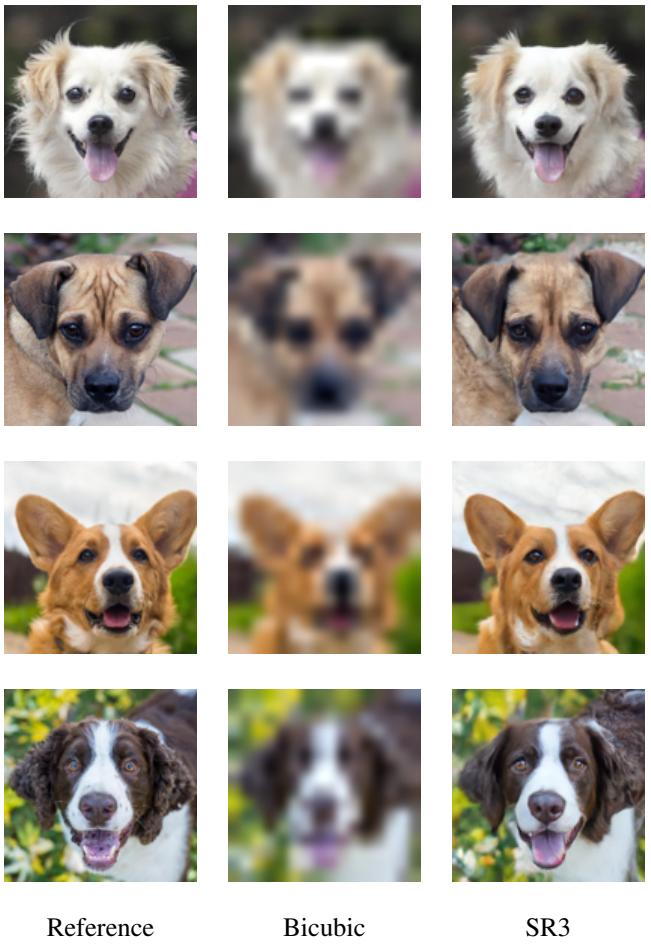


Figure 7: SR3 applied to 4 test images from AFQH-dog dataset

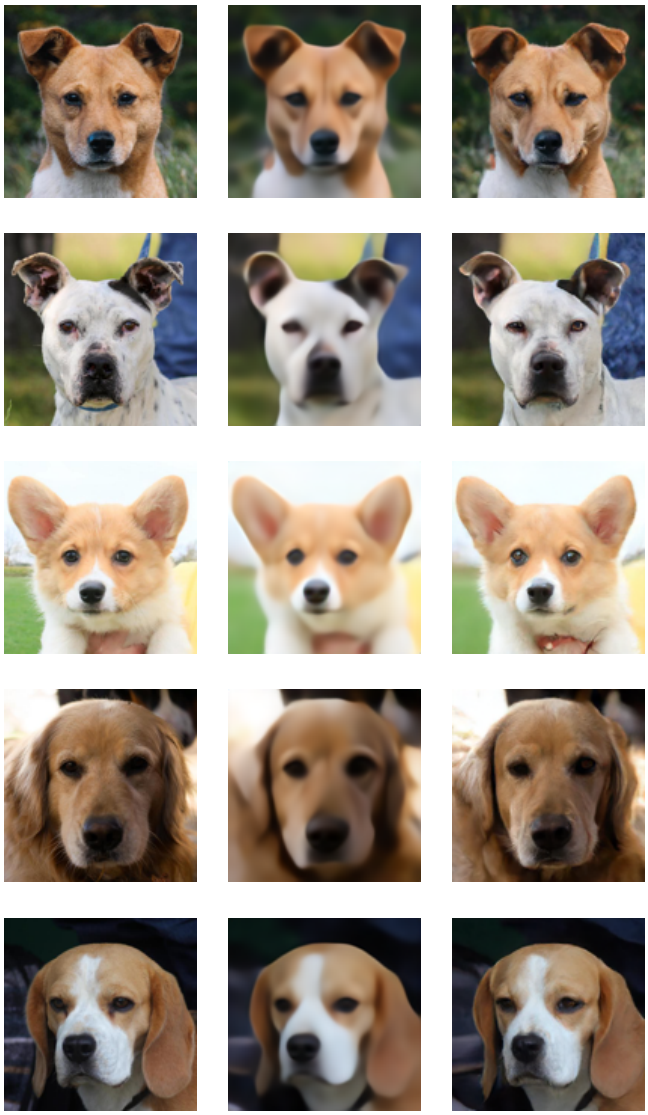


Figure 8: Comparison between SR3 and baseline model (regression based) on test images from AFQH-dog dataset

should rely on other metrics to evaluate perceptual quality eg. FID (Fréchet Inception Distance) [2] which captures the similarity of generated images to real ones. Unfortunately, this metric requires large number of samples (∼50000 images) which is very time consuming in the case of diffusion models.

4.4. Noise scheduling

In this paragraph we will investigate some interesting advantages of the noise scheduling proposed by authors.

As explained in sec. 2.3, SR3 conditions on γ directly (vs t as in [3]), which allows more flexibility in choosing number of diffusion steps, and the noise schedule during inference since the training was not done on a fixed discretized set of γ_t but randomly sampled ones.

After training the SR3 model on AFQH-dog dataset with a linear scheduling of $T = 2000$ iterations, we tested the inference on some test images using a lower number of iterations while keeping the same values of variance β_0 and β_T. By doing so we are discretizing the interval $[\beta_0, \beta_T]$ with a larger step size.

Figure 9 shows results of inference unsing different number of iterations (2000 (same as training), 1000, 2000 and 50). Interestingly, we found that with half the number of iterations used for training, we achieved the same quality results in terms of PSNR and SSIM as shown in table 4.4.

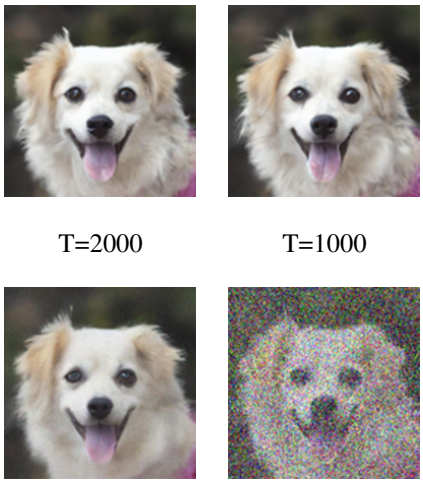


Figure 9: Comparison different number of iterations during inference

Number of sampling iterations	PSNR	SSIM
2000 (same as training)	23.05	0.62
1000	23.08	0.62
200	22.08	0.58
50	13.39	0.11

5. Conclusion

Through our theoretical study of SR3 model, we realised the flexibility and tractability advantages offered by diffusion models compared to other generative models such as GANs and Variational AutoEncoders. We noticed the convergence that may happen in future works between probabilistic diffusion models and score based models.

Our trainings have shown that the super resolution task is succeeded by SR3 model better than state-of-the-art models. This performance is not always concretized by metrics such as PSNR and SSIM, which confirms that those metrics are not adapted to evaluate models on such a task. However, human based metrics such as visual evaluation and human fool rate and FID show that results achieved by SR3 are remarkably satisfying.

References

[1] Y. Chen, Y. Tai, X. Liu, C. Shen, and J. Yang. Fsrnet: End-to-end learning face super-resolution with facial priors. 2018.

[2] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, G. Klambauer, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017.

[3] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. 2020.

[4] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. *CoRR*, abs/1603.08155, 2016.

[5] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[6] S. Menon, A. Damian, S. Hu, N. Ravi, and C. Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. 2020.

[7] A. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. 2021.

[8] C. Saharia, J. Ho, W. Chan, T. Salimans, D. J. Fleet, and M. Norouzi. Image super-resolution via iterative refinement. 2021.

[9] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *CoRR*, abs/1503.03585, 2015.

[10] Y. Song. Score-based generative modeling through stochastic differential equations. 2020.