Reimplementation of Score-based Generative Modeling Through Stochastic Differential Equations

Peng Huang, Jingkun Lin, Wenjun Zhang, Tianhao Yao
{phuang,jkunlin,wjz,yth0922}@bu.edu

1 Task

The core idea of our project is to generate similar pictures with the given picture set. For example, if we have a bunch of portraits of real human individuals, we want to use our algorithm to generate some new images that are similar to real human pictures but they are actually generated by computers instead of photographed. Mathematically, pictures of a specific type, like human portraits can be seen as samples from an unknown complicated high-dimensional distribution. The generation of new pictures is sampling from this distribution without knowing its closed-form probabilistic density function.

To accomplish this, [Song et al., 2020] introduces Score-Based Generative Modeling with Stochastic Differential Equations(SDE). They use SDE to model the process of converting the complicated data distribution of the image data to a simple normal distribution by injecting noise gradually. The SDE modeling this process has a corresponding backward SDE to remove noise to convert it back to data distribution. Then they discretized the domain and used Langevin dynamics to numerically sample from the target distribution of the backward SDE, which resulted in samples from the desired distribution. The output will have the same effect as Fig.1. Our job is to re-implement the process and have a well-trained model.

For the distribution of workload, Jingkun Lin proposed the idea about implementing score-matching SDEs based on [Song et al., 2020][Song et al., 2021], wrote the framework of the article and trained the model. Wenjun Zhang provided codes for training DDPM and integrated with [Song et al., 2021] and wrote about the FID to evaluate the models. Tianhao Yao was mainly reponsible for writing the article and searching related works. Peng Huang modified the code and reviewed the article.

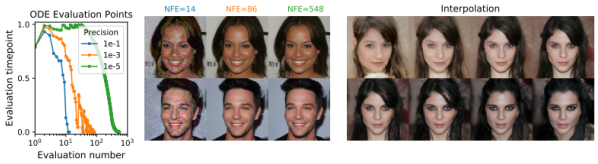

Figure 1: According to [Song et al., 2020], probability flow ODE makes it possible to perform rapid sampling with adaptable stepsizes that vary depending on the numerical precision (depicted on the left), while simultaneously decreasing the number of score function evaluations (NFE) without compromising the quality of the output (depicted in the middle). Additionally, an invertible mapping between latent variables and images facilitates interpolations (depicted on the right).

2 Related Work

The use of generative models in unsupervised learning tasks has become increasingly popular in recent years due to its effectiveness in understanding the implicit distribution of datasets without having to know the closed form of its probability density function. These models can generate high-quality images by learning the distribution information from realistic pictures and generating new samples according to this distribution. Generative models can be categorized into two main groups: cost function-based and energy-based. Variational autoencoders and generative adversarial networks are examples of cost function-based models, while Boltzmann machines and deep belief networks are examples of energy-based models. Generative models have been applied to various fields, including visual recognition, speech [Kong et al., 2021], natural language processing[Li et al., 2022], and robotics.

Recent progress on the likelihood-based method includes variational autoencoder[Kingma and Welling, 2013], recurrent neural network[Graves, 2013],non-linear independent component estimation[Dinh et al., 2014]. In the likelihood-based methods, log-likelihood is used as objective function for training.

Generative adversarial networks, proposed by Goodfellow et al. [2014] in 2014, consists of a discriminator network judging whether a newly sampled image obeys the same distribution as given samples and a generative model adjusting itself with backpropagation to create counterfeits to cheat the discriminator. GANs avoid the drawbacks of the likelihood-based method. However, GANs also have their own disadvantages like lacking stability and it can be difficult to compare different models. Tuning the hyperparameters of GANs is also known to be a tedious task[Brock et al., 2019].

Recently diffusion models has been developed by [Song and Ermon, 2019]. They use several discrete noise scales from very small to very large to perturb the samples. Then they used a score network to match the gradient of transit kernels at different noise levels and used Langevin dynamics to sample.

A similar model called Denoised Diffusion Probabilistic Model was proposed by [Ho et al., 2020]. They generalized autoregressive decoding using tractable probabilistic models and constructed a Markov chain to model the ablation process of images to noises. Then they reversed the Markov chain to get image samples. In their denoising diffusion probabilistic modeling(DDPM), a sequence of probabilistic models is trained to reverse the process of adding noise. With noise added, the information of data is gradually but systematically destroyed while the reverse process is to restore the structure of in given data.

[Song et al., 2020] incorporated these two methods under a unified theoretical framework of stochastic differential equations, which can be seen as a continuous generalization of both Langevin dynamics and Markov Chain. These two methods can be seen as two ways to discretize the same SDE and the SDE can be interpreted as having an infinite number of noise scales whose time steps were continuously sampled. With the result of [Anderson, 1982], the theoretical foundation of why reversing SDE results in our desired distribution is solid. For a given process to add noise to a real image data set, we can represent it as a stochastic process modeled by a specific SDE and with Anderson's theory we can get the form of the SDE of the inverse process which shares the same probability density with the original process at any time. If we can get the parameter of reversed SDE then other numerical methods for solving SDE such as Euler-Maruyama could be used to get samples. They also demonstrated the efficacy of predictor-corrector samplers that blend Hamiltonian Monte Carlo and Langevin Monte Carlo Markov Chain methods. Additionally, deterministic samplers based on the probability flow ordinary differential equation have been shown to provide fast and adaptable sampling, flexible data manipulation, uniquely identifiable encoding, and exact likelihood computation.

In our work, we combined the implementation of [Song et al., 2021] in PyTorch with UNet and tested it on CelebA dataset and compared its FID with the result from DDPM.

3 Approach

3.1 Theory

Score Matching Methods

Score matching was introduced by [Hyvärinen and Dayan, 2005] . They used this method to infer the parameter of probability density models $p(x;\theta)$ with intractable partition function $Z(\theta)$. The probability density model is as following:

$$p(x;\theta) = \frac{1}{Z(\theta)} \exp\big(-E(x;\theta)\big),$$

where E is called energy function. They calculate the gradient of the log density with respect to the data:

$$\psi(x;\theta) = \frac{\partial \log p(x;\theta)}{\partial x} \tag{1}$$

Learning θ is the essential part of score matching. As a result,

$$\psi(x;\theta) = \frac{\partial \log p(x;\theta)}{\partial x} \tag{2}$$

best matches the corresponding score of the true distribution: $\frac{\partial \log q(x)}{\partial x}$. Explicit score matching (ESM) and implicit score matching (ISM) are the two kinds of objectives. The objective function of ESM is the expectation of squared error between these two vectors:

$$J_{ESMq}(\theta) = \mathbb{E}\left[\frac{1}{2}\left\|\psi(x;\theta) - \frac{\partial \log q(x)}{\partial x}\right\|^2\right].$$

Note the following remarkable property:

$$\mathbb{E}\left[\frac{1}{2}\left\|\psi(x;\theta) - \frac{\partial \log q(x)}{\partial x}\right\|^2\right] =$$

$$\mathbb{E}\left[\frac{1}{2}\|\psi(x;\theta)\|^2 + \sum_{i=1}^{d} \frac{\partial \psi_i(x;\theta)}{\partial x_i}\right] + C_1,$$

where C_1 is a constant that does not depend on θ. This yields an implicit SM objective

$$J_{IMSq}(\theta) = \mathbb{E}\left[\frac{1}{2}\|\psi(x;\theta)\|^2 + \sum_{i=1}^{d}\frac{\partial\psi_i(x;\theta)}{\partial x_i}\right]$$

that no longer requires having an explicit score target for q but is nevertheless equivalent to $J_{IMSq}(\theta)$.

Denoising diffusion probabilistic models(DDPM)

As stated in [Sohl-Dickstein et al., 2015], a sequence of noise scales $0 < \beta_1, \beta_2, \ldots, \beta_N < 1$ is used to construct a discrete Markov chain $\{.x_0, x_1, \ldots, x_N\}$ such that

$$p(x_1|x_{i-1}) = \mathcal{N}(x_1; \sqrt{1-\beta_i}x) \tag{3}$$

. The perturbed data distribution is

$$p_\alpha(\tilde{x}) = \int p_{data}(\tilde{x}|x)\mathrm{d}x \tag{4}$$

.

The reversed direction is represented as

$$p_\theta(x_{i-1}|x_i) = \mathcal{N}(x_{i-1}; \frac{1}{\sqrt{1-\beta_i}}(x_1 + \beta_i s_\theta(x_i, i)), \beta_i\mathbf{I}) \tag{5}$$

and can be trained with

$$\theta^* = \arg\min_\theta \sum_{i=1}^{N}(1-\alpha_i)\mathbb{E}_{p_{data}(x)}\mathbb{E}_{p_{\sigma_i}(\tilde{x}|x)} [||s_\theta(\tilde{x}, i) - \nabla_{\tilde{x}}\log p_{\alpha_i}(\tilde{x}|x)||_2^2] \tag{6}$$

The sampling method started from $x_N \sim \mathcal{N}(o, \mathbf{I})$ and the estimated reverse Markov chain is

$$x_{i-1} = \frac{1}{\sqrt{1-\beta_i}}(x_i + \beta_i s_\theta^*(x_i, i)) + \sqrt{\beta_i}z_i. \tag{7}$$

In our project, we proposed an approach consisting of three stages, which are perturbing, score-matching and reverse estimation.

Perturbing data with SDE

In order to produce additional samples from the distribution$p(x)$, we apply the denoising score-matching framework, which serves as an effective theoretical approach to estimate distributions from their samples. But, it necessitates a perturbation process that often calls for heuristic fine-tuning. Song et al. suggested that, as an alternative to perturbing data using fixed-scale Gaussian noise, we can use stochastic differential equations for perturbation.

For our experiment, we apply the Variance Preserving (VP) SDE, which takes the form

$$dx = -\frac{1}{2}\beta(t)dt + \sqrt{\beta(t)}dw$$
$$\beta(t) = \beta_0 + (\beta_T - \beta_0)\frac{t}{T} \tag{8}$$

The benefit of this is that when both drift and diffusion coefficients are affine, the conditional perturbation process $x(t)|x(0)$ becomes a Gaussian process. In fact, it can be determined that for the VP SDE, the conditional distribution is

$$p_{0t}(x(t)|x(0)) = \mathcal{N}(x(t); x(0)e^{-\frac{1}{2}\int_0^t\beta(s)ds}, I - Ie^{-\int_0^t\beta(s)ds}). \tag{9}$$

As a result, the perturbation process does not need numerical integration but can instead rely on Gaussian sampling.

Estimating the perturbed distribution with score matching

In order to effectively use information from all t, Song et al. introduced the following objective for score-matching:

$$\theta^* = \arg\min_\theta \mathbb{E}_t\Big\{\lambda(t)\mathbb{E}_{x(0)}\mathbb{E}_{x(t)|x(0)} \big[\|s_\theta(x(t), t) - \nabla_{x(t)}\log p_{0t}(x(t)|x(0))\|_2^2\big]\Big\} \tag{10}$$

. The influence of different t values is regulated by $\lambda(t)$, which cannot be a constant because, if $x(t) \sim \mathcal{N}(\mu_t, \sigma_t^2)$ and we reparameterize it as $x(t) = \mu_t + \sigma_t\epsilon$, the score transforms to $\nabla\log q = -\frac{\epsilon}{\sigma_t}$. As $t \to 0$, the score's length would explode when $\sigma_t \to 0$.

Nevertheless, Huang et al. noted that the introduction of $\lambda(t)$ results in bias. Therefore, they proposed setting $\lambda(t)$ as a constant, but sampling t not from a uniform distribution, but rather from $t \sim q(t) \propto \frac{1}{\sigma_t^2}$. In doing so, the bias is significantly reduced and the variance does not increase uncontrollably.

Numerical estimation of reverse SDE

It is necessary to simulate the reverse SDE to create samples from the stochastic process,. Song et al. explained that if $x(t)$ is the solution to the SDE

$$dx = f(x,t)dt + g(t)dw \tag{11}$$

, then the reverse process, denoted as $\bar{x}(t)$, is the solution to the SDE

$$dx = [f(x,t) - g(t)^2 \nabla_x \log p_t(x)]dt + g(t)d\bar{w} \tag{12}$$

, where $\bar{w}$ represents the reversed Wiener process from time T to 0. It is important to note that the only additional information required for this SDE is $\nabla_x \log p_t(x)$, which is the score function. Consequently, following score-matching, we possess all the necessary information to generate samples by simulating the reverse SDE.

In our project, we apply three methods: the Euler-Maruyama method, the Predictor-Corrector method and the Ordinary Differential Equation (ODE) sampler. Then we will compare the results.

The Euler-Maruyama method is a straightforward extension of the Euler method for ODEs to SDEs. For an SDE $dx = fdt + gdw$, the Euler-Maruyama method updates recursively through

$$x_{n+1} = x_n + f \cdot (t_{n+1} - t_n) + g \cdot (w_{n+1} - w_n) \tag{13}$$

. Similar to the Euler method, the Euler-Maruyama method is not highly stable in terms of simulation.

Predictor-Corrector (PC) methods incorporate a corrector step into the recursion. After defining

$$\tilde{x}n + 1 = x_n + f \cdot (tn + 1 - t_n) + g \cdot (w_{n+1} - w_n) \tag{14}$$

, the corrector step involves a single step of Langevin dynamic simulation,

$$x_{n+1} = \tilde{x}n + 1 + \epsilon s\theta(\tilde{x}n + 1, tn + 1) + \sqrt{2\epsilon}z \tag{15}$$

, where z is standard Gaussian noise. Empirically, PC methods produce better samples, but theoretically, they deviate from the SDE simulation.

The Ordinary Differential Equation (ODE)sampler transforms the problem of sampling from a target distribution to solving a system of ordinary differential equations. This transformation is achieved through the introduction of an ODE system that evolves in a continuous-time manner, preserving the properties of the target distribution, such as its mean and covariance structure. Given the trained score-matching network we can express the deterministic process in the following ordinary differential equations:

$$\mathrm{d}x = \left[\mathbf{f}(\mathbf{x}, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] \tag{16}$$

This ODE is named probability flow in Song et al's work [Song et al., 2020]. We can sample first from the standard normal $\mathbf{x}_0 \sim p_0(\mathbf{x})$ and use it as a boundary condition and use any ODE solver the get an exact solution with the given boundary and get a specific number of $\mathbf{x}_T$. The ODE sampler can help us trade-off between accuracy and efficiency using different step sizes.

ODE samplers are able to explore the target distribution more efficiently compared to other sampling methods, however, tuning various hyperparameters such as step sizes and integration times are required to ensure accurate simulations.

3.2 Architecture

We estimated the score function using the score-matching model with UNet. The UNet model with skip connections is a popular model used for image generation or segmentation.

The Unet consists of a down sampling part to reduce the resolution of images and increase the number of channels using pooling and convolution,an up sampling part to increase the size of images and blocks with convolution and nonlinearity in between. The size of images inside the net will shrink first and then expand so it has a shape like "U".

In our homework we have used Unet to train the network used in DDPM. To get a fair comparison with score-matching method, we used the same net architecture to get the score in both cases.

However, since there is a mismatch of dimensions, we chose to eliminate a normalization part from the original score-matching version given by [Song et al., 2021]. We suspect that this expedient part may somehow harm the accuracy of our samples since the training could be relatively more unstable without normalization.

4 Dataset and Metric

4.1 Dataset

The dataset based on which we're going to train our model is CelebA. It is a face attribute dataset consisting of more than 200,000 images of 10,177 celebrity identities. It is widely used in computer vision and machine learning research for tasks such as face recognition, face detection, and facial attribute analysis. Each image in the dataset is annotated with 40 binary attributes, including gender, age, hair color, facial hair, and eye color. These annotations make the dataset a valuable resource for studying facial attributes and their correlations.

4.2 Evaluation Metrics

The metric we plan to use is Fréchet inception distance (FID) [Heusel et al., 2017]. FID is commonly used to evaluate the performance of generative models in image synthesis tasks. The FID metric is calculated based on the feature representations of the generated and real images. Specifically, the metric calculates the distance between the multivariate Gaussian distributions of the feature representations of the generated and real images. The feature representations are obtained by passing the images through a pre-trained Inception-v3 network, and the multivariate Gaussian distributions are estimated using sample mean and covariance matrices.

The FID metric is calculated as follows:

$$FID = ||\mu_1 - \mu_2||^2 + \mathrm{Tr}(\Sigma_1 + \Sigma_2 - 2(\Sigma_1\Sigma_2)^{1/2})$$

where μ_1 and μ_2 are the mean vectors of the feature representations of the generated and real images, and Σ_1 and Σ_2 are the covariance matrices of the feature representations of the generated and real images, respectively. The Tr function denotes the trace of a matrix, and $||\cdot||$ denotes the Euclidean norm.

We hope to show that our improvement can decrease FID on the test set compared to the method in the original paper.

5 Results

After training the score-matching network using UNet we sampled with different discretization methods of stochastic differential equations. We sampled using Euler-Maruyama, Predictor-Corrector and ODE solvers to get samples. We then compared the FID between the score-matching method with Unet replacing the original ScoreNet used by Song, and the DDPM network with Unet.

Table 1 displays our result evaluated by FID. Compared to the DDPM baseline, our scored-based model with SDE didn't produce better performance as we expected. The FID for DDPM is 100.7 while the score-matching model through SDEs is 159.3. Meanwhile, DDPM took 10 epoch to get the result while score-matching through SDEs took 50.

From the concrete samples from appendix we can see what the real pictures look like. When epoch is 10, the quality of generated images is hard for human eyes to recognize the face no matter what sampler we use243. When we increase the epoch to around 50, we can clearly see that Euler-Maruyama sampler began to generate pictures which is more similar to real human faces5. However the PC-sampler went blank and ODE samplers produced images with quality inferior to Euler-Maruyama's67.

	DDPM	Score-based SDE
FID	100.7	159.3

Table 1: Quantitative results

For the reason of this result, we thought of several possible extensions we could do in the future. Firstly we should try to solve the dimension mismatch with the normalization added back to make training stable. Secondly there are still some hyperparameters within the samplers which we may able to optimize, like the stepsize of ODE solver. Third, since Song's original implementation used a different architecture called ScoreNet which could be more specific for score-matching through SDE's models, maybe Unet is not the best choice for this model. Other hyperparameters like numebr of layers in the Unet and the random distribution from which we get the time step samples still remain to be optimized.

6 Conclusion

In this project, we implemented scored-based generative methods through stochastic differential equations. We used Fréchet inception distance metrics to evaluate the generated samples and compared them with DDPM using same Unet architecture.

We used Unet architecture to train the neural network to estimate the score and compare the DDPM and score-matching through SDEs. It turned out that SDEs took more time and struggle with getting images with safisfiable qualities. We thought about several possibilities that may improve the performances of SDE based model.

Score-matching through SDEs provided an unified theoretical framework for both DDPM and Score matching Langevin dynamics mentioned in [Song and Ermon, 2019]. It generalized them from discrete stochastic process to a continuous one with solid theoretical foundation and various method and variations to explore. It's definitely worth doing further exploration and optimization within this framework.

A Code repository

You can see our code repository on GitHub.

References

B. D. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12 (3):313–326, 1982.

A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. *ArXiv*, abs/1809.11096, 2019.

L. Dinh, D. Krueger, and Y. Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

A. Graves. Generating sequences with recurrent neural networks. *ArXiv*, abs/1308.0850, 2013.

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. 2017. doi: 10.48550/ARXIV.1706.08500. URL https://arxiv.org/abs/1706.08500.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

C.-W. Huang, J. H. Lim, and A. C. Courville. A variational perspective on diffusion-based generative models and score matching. *Advances in Neural Information Processing Systems*, 34:22863–22876, 2021.

A. Hyvärinen and P. Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *ArXiv*, abs/2009.09761, 2021.

X. L. Li, J. Thickstun, I. Gulrajani, P. Liang, and T. Hashimoto. Diffusion-lm improves controllable text generation. *ArXiv*, abs/2205.14217, 2022.

J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

B Sample Images

Figure 2: Sampled images using Score-based generative modeling through SDEs, epoch = 10

Figure 3: Sampled images using Score-based generative modeling through SDEs, epoch = 10

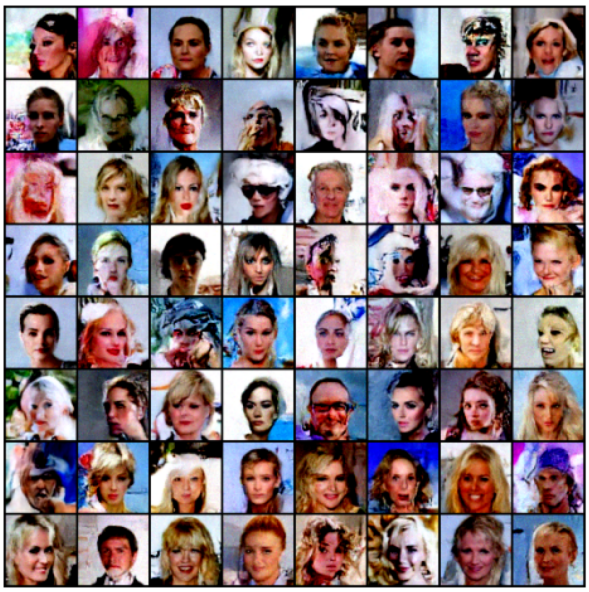

Figure 5: Sampled images using Score-based generative modeling through SDEs, epoch = 50

Figure 4: Sampled images using Score-based generative modeling through SDEs, epoch = 10

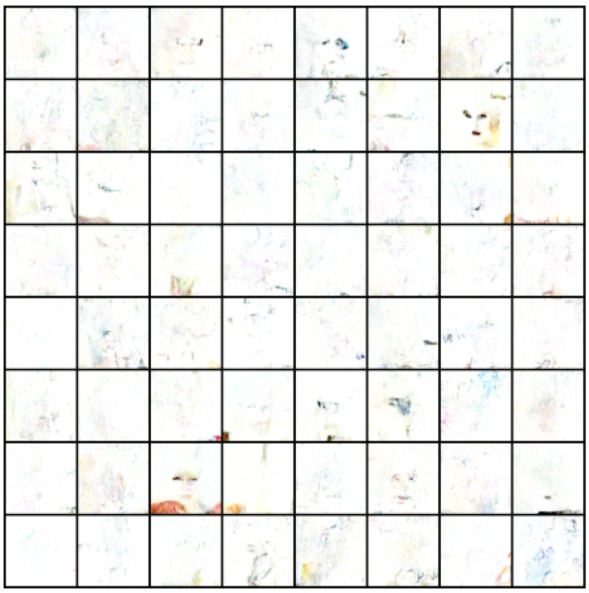

Figure 6: Sampled images using Score-based generative modeling through SDEs, epoch = 50

Figure 7: Sampled images using Score-based generative modeling through SDEs, epoch = 50

Figure 8: Sampled images using DDPM, epoch = 10