

# Report For Diffusion Model

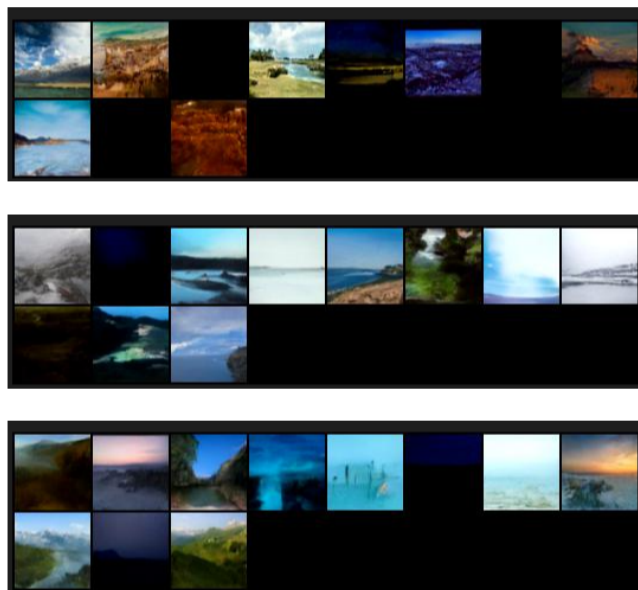
## Introduction

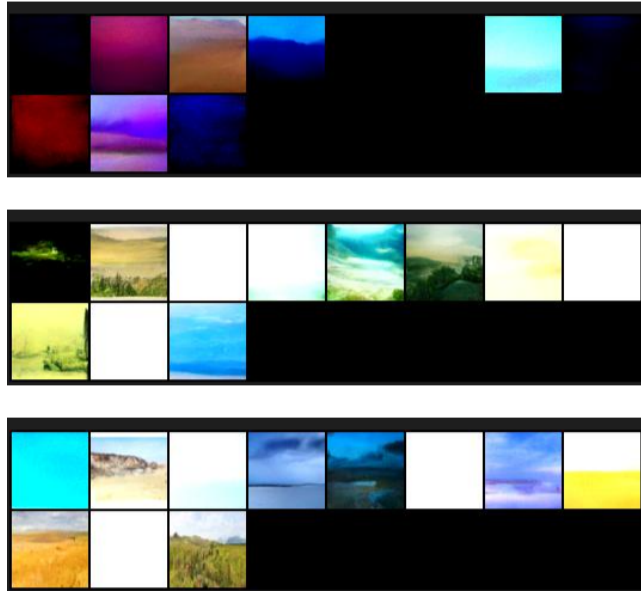
After exploring several generative models like Generative adversarial networks and its flavors, we tried our hands in training diffusion models to generate landscape images which outperformed all the previous models that we worked on.

## Model Comparison

Let's get a quick comparison among images generated by these two models.

### Images generated by Diffusion Model





## Images generated by GAN Model

### Observations

- Diffusion model was able to generate high-quality images in comparison to GAN models.
- Training diffusion model was computationally expensive compared to GAN models.
- Slight changes in training data caused the GAN model to generate a completely different image every time, making it sensitive to training data.
- We stuck to parameters provided by the diffusion model paper and got stable output.
- On changing the hyperparameters, the training became unstable for the Diffusion model, which wasn't the case for GAN models.

## Theoretical Learnings

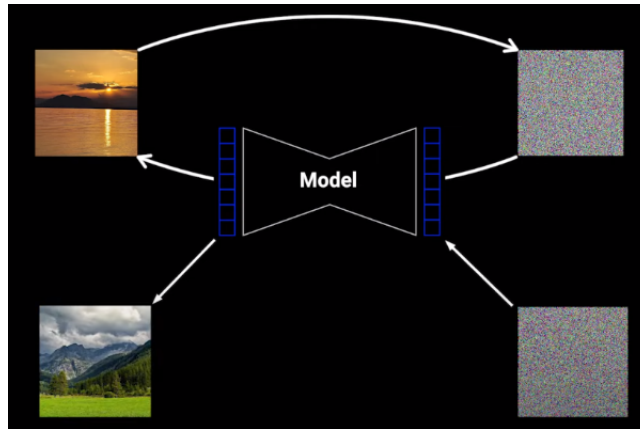
While exploring the theoretical aspects of Diffusion Models, we went through a few Research Papers, with the primary 4 (in chronological format) being:

1. Deep Unsupervised Learning using Nonequilibrium Thermodynamics
2. Denoising Diffusion Probabilistic Models

### 3. Improved Denoising Diffusion Probabilistic Models

### 4. Diffusion Models Beat GANs on Image Synthesis

The essential idea of diffusion models is to systematically and slowly destroy the data distribution through an iterative forward diffusion process. We then learn a reverse diffusion process that restores structure in data, yielding a highly flexible and tractable generative model of the data.



The DDPM paper from 2020 laid out three things the neural network could predict:

1. Predicting the mean of the noise at each time step
2. Predicting the original image directly
3. Predicting the noise in the image directly

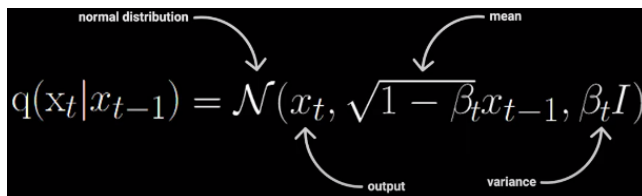
Predicting the original image won't work well, so we can ignore this choice, and the first and the third option are the same just parametrized differently, and all authors decided to go with the third one.

Initially, we were only learning the mean as in the first option, and the variance was fixed. However, in the following papers by OpenAI authors, we decided that learning the variance leads to improvement in the log likelihoods.

The amount of noise added to the image is regulated by a schedule. A common schedule was the linear schedule, which was too rapid, hence some information was getting destroyed, and so modern models use a cosine schedule to make this step more gradual.

The first papers were using a UNet architecture and then the second paper by the openAI authors decided to make a few major changes to this architecture, which heavily improved the overall outcome. The updates that they made were that first of all they increased the Depth of the network and decreased the width then, they included more attention blocks than the original proposal and also

increased the number of attention heads. They also took the residual blocks from big GAN and used this for the upsampling and downsampling blocks. Next they proposed what they call adaptive group normalization which is just a fancy name for the idea of incorporating the time step slightly differently and additionally also the class of the label.



The diagram shows the equation  $q(x_t|x_{t-1}) = \mathcal{N}(x_t, \sqrt{1 - \beta_t}x_{t-1}, \beta_t I)$  on a black background. Four white arrows point to parts of the equation: 'normal distribution' points to the  $\mathcal{N}$  symbol; 'mean' points to the  $\sqrt{1 - \beta_t}x_{t-1}$  term; 'output' points to the  $x_t$  variable; and 'variance' points to the  $\beta_t I$  term.

## Additional Learning

### Server Setup

Due to several obstructions, we weren't able to run our models on a server that was already configured. Thus we had to request a separate server, which we configured from scratch. We were able to create environments necessary for running our models. Although we encountered endless problems while configuring the server, we learned a lot about how everything worked.

### Paper Reading

We implemented all of our models by thoroughly studying academic papers and examining codebases from other developers. Reflecting on our journey, we acknowledge that we began as novices, but through the course of this project, we have elevated our skills beyond that initial stage. This experience has been instrumental in our growth, and we now find ourselves at a level of proficiency that surpasses our initial status as beginners.

## References

1. <https://arxiv.org/abs/1511.06434>
2. [https://pytorch.org/tutorials/beginner/dcgan\\_faces\\_tutorial.html](https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html)
3. <https://github.com/dome272/Diffusion-Models-pytorch>
4. <https://arxiv.org/abs/2006.11239>
5. <https://arxiv.org/abs/2209.00796>
6. <https://www.youtube.com/watch?v=HoKDTa5jHvgt=580s>
7. <https://www.youtube.com/watch?v=fbLgFrITnGU>