Generating Faces with Latent Diffusion

Anonymous author

Abstract

In this paper, we introduce our approach of using a computationally efficient latent diffusion model (LDM) to generate high-resolution 128x128 faces. We trained the model on the FFHQ dataset and evaluated the generated images for realism, diversity, and uniqueness. Our results show that the model can generate high-quality images with realistic shapes and textures, while maintaining diversity and uniqueness, all with computational efficiency. We also discuss limitations and provide examples of the generated images.

1 Methodology

1.1 Background

1.1.1 Diffusion model

Diffusion models, introduced in [5] aim to convert any complex data distribution into a simple tractable data distribution, and then find a finite step reverse process for it.

They do this by gradually converting the data distribution labeled $q(x_0)$ to noise by a repeated application of a Markov Diffusion Kernel T_π. In DDPM they use the Gaussian distribution with identity covariance. In other words they apply normally distributed Guassian noise where the only variables defining it are mean and variance. The variance is controlled by a variance schedule denoted β_t. This gives us

$$q(x_t \mid x_{t-1}) = T_\pi(x_t \mid x_{t-1}; \beta_t) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}; \beta_t \mathbf{I}) \tag{1}$$

the full forward trajectory is thus

$$q(x_{0:T}) := q(x_0 \prod_{t=1}^{T} q(x_t \mid x_{t-1})) = q(x_0 \prod_{t=1}^{T} \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}; \beta_t \mathbf{I})) \tag{2}$$

If we fix the noise schedule we can then obtain from this a closed form expression for x_t after applying t timesteps of noise given we are starting from x_0

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\alpha_t}x_0, (1-\alpha_t)\mathbf{I}) \tag{3}$$

where $\alpha_t = \prod_{s=1}^{t}(1-\beta_t)$

The reverse process is the same but in reverse. We denote it by $p_\theta(x_0)$ where θ are the parameters used in the prediction model.

In order to learn this process [5] samples random batches of images and applies a random number of time steps of noise to each image using the closed form. The model then predicts the mean and variance of the noise that was added and is optimised using stochastic gradient descent. Diffusion tries to maximize the likelihood of the target distribution $p_\theta(x_0)$ given a set of training samples. We negative-log-likelihood however as this is is intractable, so we approximate it using the usual variational lower bound, specifically ELBO loss. This gives

$$\mathbb{E}\left[-\log p_\theta\left(\mathbf{x}_0\right)\right] \leq \mathbb{E}_q\left[-\log \frac{p_\theta\left(\mathbf{x}_{0:T}\right)}{q\left(\mathbf{x}_{1:T} \mid \mathbf{x}_0\right)}\right] = \mathbb{E}_q\left[-\log p\left(\mathbf{x}_T\right) - \sum_{t \geq 1} \log \frac{p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t\right)}{q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right)}\right] =: L \tag{4}$$

1.1.2 DDPM and DDIM

[2] re-parametrises the problem and predicts the noise matrix that was added to image in the first place (ϵ). We replace the vague definition of the loss from before with a specific definition of the Mean Squared Error between the predicted noise and the true noise added (using the closed form defintion from before) this gives:

$$\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t\left(1-\bar{\alpha}_t\right)}\left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0+\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon},t\right)\right\|^2\right] \tag{5}$$

lastly [2] simplifies the loss function by removing the weighting as find it is unnecessary, this gives them their final algorithm for training as

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1,\dots,T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$
5: Take gradient descent step on
$\nabla_\theta \left\|\boldsymbol{\epsilon}-\boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0+\sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon},t)\right\|^2$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0},\mathbf{I})$
2: **for** $t = T,\dots,1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t)\right) + \sigma_t\mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

The Denoising Diffusion Implicit Model (DDIM) [6] extends DDPM by using an implicit generative model to map the noisy input and the diffusion step to the denoised output. It is learned by minimizing a contrastive loss that encourages similar noise distributions to produce similar outputs. This can be used to produce samples much faster with a higher quality when compared to DDPM used with a low number of samples.

1.2 The need for speed

Diffusion models are notoriously slow to train. For instance, ADM in [1] requires 116 v100 days for LSUN Horse 256x256. Although [3], [6], [1] improve FID scores, they don't reduce training times. LDM in [4] trains models in a compressed latent space, which decreases parameter size and Flops per iteration, leading to improved efficiency without compromising image quality. Additionally, [8] proposes a spectrum-aware distillation method for training a reduced model while achieving the same quality, however the paper was unpublished and the code was unavailable. Therefore, we focused on LDM alone, which provides a speedup in training time to achieve high-quality images.

1.3 LDM first stage

The main contribution of LDM [4] is training a generative model in a compressed latent space rather than in pixel space. An autoencoder architecture is used to compress the large images efficiently before training. The encoder performs down sampling using a convolutional layer with stride 2 followed by two resnet blocks. The decoder is the reverse of the encoder. The perceptual similarity score from "Learned Perceptual Image Patch Similarity"[9] neural network is used as the loss function instead of MSE. The autoencoder is trained with various latent sizes from $8x8x4$ to $64x64x1$, and larger latent sizes are found to train the diffusion model better due to the noise-to-signal ratio being too high for smaller sizes. The autoencoder was trained for 24 hours with an input channel size of 128 and a final reconstruction loss of 0.05. This sets the limit on what the second stage could produce, making it necessary for the autoencoder to perform well. We found using a discriminator at the 50k step mark greatly improved the visual quality even if loss did not decrease.

1.4 LDM second stage

The second stage of LDM, is built on [2] with many elements of the ADM model from [1] mixed in. For example the U-Net architecture is entirely from [1]. The UNet architecture

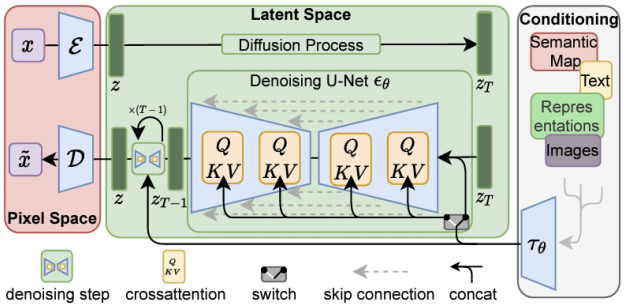


Figure 1: The original LDM architecture. Ours is almost identical except we have removed the conditioning elements (except for timestep conditioning)

used in the paper includes a stack of residual layers and downsampling convolutions, followed by a stack of residual layers with upsampling convolutions. Skip connections connect the layers with the same spatial size. The model uses attention at multiple resolutions, including 32x32, 16x16, and 8x8, in addition to the global attention layer at 16x16. A projection of the timestep embedding is added into each residual block, allowing the model to capture the temporal evolution of the diffusion process. Surprisingly LDM opts for linear beta scheduling for the noise, intentionally passing on recent methods like cosine scheduling [3].

Based on [8] we knew that whilst a reduction in the UNet model channels would reduce complexity and speed up training times, it would certainly come at the cost of quality. However unlike in LDM's paper we were training on smaller images, so intuitively we did not need the full model size. Tobe cautious we reduced the model channels only slightly from 224 to 192. We kept the attention resolutions the same.

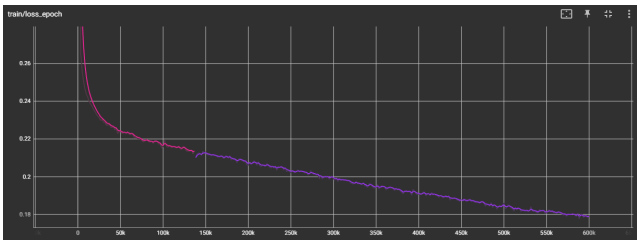
LDM also uses an Exponential Moving Average (EMA) [7] of the models parameters to ensure that when evaluating the model uses a smoothed version of it's parameters and is not as susceptible to noise in the training.

LDM also uses DDIM, as described before inorder to increase the speed of logging images and when faster inference is needed as it is able to produce images 10-15x faster than with ddpm at only a slight loss of quality. This is very helpful when debugging.

The LDM was trained for exactly 600,000 step. This took 72 hours on a 4090 GPU.

2 Results

- **Quality**: The sampled images are of high quality where the best images even appear photo realistic. It is not as good as StyleGan2 but with more training time it would've likely come close. There does not appear to be any mode collapse even when zooming in.
- **Variety**: In the cherry picked group of 16, we included some lower quality but more diverse samples, which demonstrate the am zing variety of faces that the model can generate. Within the non-cherry picked samples we can see that it produces faces of both genders, all ethnicity's, all hair colours and all ages.
- **Uniqueness**: Whilst we could not include a neighbourhood image without running out of space the often strange combinations, like a baby with died blue hair, should convince you that it's images are unique.



3 Limitations

- The training of latent diffusion models can be slow and may not converge in a reasonable amount of time, particularly on lower-end hardware.Inference with LDM is also very slow and makes it difficult to measure certain desirable metrics like FID.
- The performance of latent diffusion models can be sensitive to the choice of hyperparameters and the quality of the training data.
- "Diffusion made slim" and conditional models were not implemented and could be explored in future research.

Bonuses

This submission has a total bonus of 8 marks, as it is trained on FFHQ 128x128, and does not use GANs.

References

[1] Prafulla Dhariwal and Alexander Nichol. "Diffusion models beat gans on image synthesis". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 8780–8794.

[2] Jonathan Ho, Ajay Jain, and P. Abbeel. "Denoising Diffusion Probabilistic Models". In: *ArXiv* abs/2006.11239 (2020).

[3] Alexander Quinn Nichol and Prafulla Dhariwal. "Improved denoising diffusion probabilistic models". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8162–8171.

[4] Robin Rombach et al. "High-Resolution Image Synthesis with Latent Diffusion Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 10674–10685.

[5] Jascha Sohl-Dickstein et al. "Deep unsupervised learning using nonequilibrium thermodynamics". In: *International Conference on Machine Learning*. PMLR. 2015, pp. 2256–2265.

[6] Jiaming Song, Chenlin Meng, and Stefano Ermon. "Denoising diffusion implicit models". In: *arXiv preprint arXiv:2010.02502* (2020).

[7] Antti Tarvainen and Harri Valpola. "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results". In: *NIPS*. 2017.

[8] Xingyi Yang et al. "Diffusion Probabilistic Model Made Slim". In: *ArXiv* abs/2211.17106 (2022).

[9] Richard Zhang et al. "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric". In: *CVPR*. 2018.