Image Generation Using a Diffusion Model with Classifier-Free Diffusion Guidance

Anonymous author

Abstract

This paper presents image generation using a denoising diffusion probabilistic model which utilizes classifier-free diffusion guidance. The model is trained on the CIFAR-10 dataset at a resolution of 32x32.

1 Methodology

This project uses a denoising diffusion probabilistic model (DDPM) [1] that utilizes classifier-free diffusion guidance [2] to generate 32x32 resolution images based on the CIFAR-10 dataset. Diffusion models learn by denoising, which consists of two key processes - each of which being a Markov chain [3].

1.1 Forward and Reverse Processes

The first is a predefined forward process that gradually adds Gaussian noise to an image x_0 over T timesteps, according to a cosine [4] variance schedule $\beta_1, ..., \beta_t$.

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \tag{1}$$

$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1}) \tag{2}$$

Using the reparameterization trick, we can sample x_t at any timestep via,

$$\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \tag{3}$$

where

$$\alpha_t = 1 - \beta_t, \bar{\alpha}_t = \prod_{t=1}^{T} \alpha_t \tag{4}$$

The second stage is a learned reverse process in which we attempt to undo the added noise at every timestep, starting from the pure noise distribution.

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)) \tag{5}$$

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \tag{6}$$

Where θ are parameters to train. In order to perform this training, we define an appropriate loss function to optimize for.

1.2 Loss Function

Since the combination of q and p resembles a variational autoencoder (VAE), we can train the model by optimizing the negative log-likelihood of the data. Through a series of calculations, it can be found that instead of predicting the mean of the distribution, the model can predict the noise ϵ at each timestep [1].

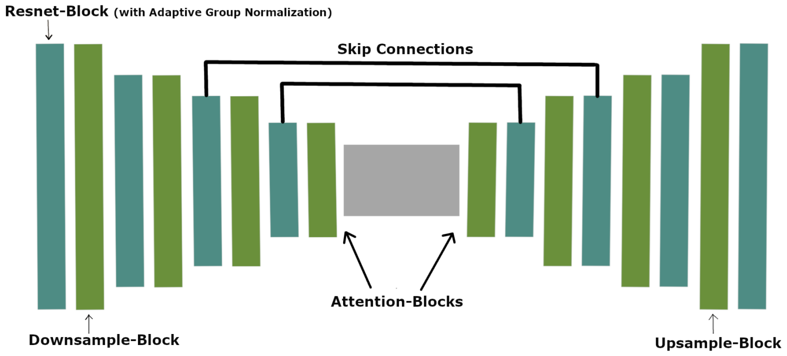
$$L_t = \mathbb{E}_{x_0,t,\epsilon}\left[\frac{\beta_t^2}{2\alpha_t\left(1-\bar{\alpha}\right)\left\|\Sigma_\theta\right\|_2^2}\left\|\epsilon_t - \epsilon_\theta\left(\sqrt{a_t}\mathbf{x}_0 + \sqrt{1-\bar{a}_t}\epsilon, t\right)\right\|^2\right] \tag{7}$$

Some simplifications to this loss term can be applied by ignoring a weighting term. This has been shown to improve the performance of the model [1]. For this reason, the following loss term is used for training,

$$L_t^{simple} = \mathbb{E}_{x_0,t,\epsilon}\left[\left\|\epsilon_t - \epsilon_\theta\left(\sqrt{\bar{a}_t}\mathbf{x}_0 + \sqrt{1-\bar{a}_t}\epsilon, t\right)\right\|\right] \tag{8}$$

1.3 The Neural Network

The neural network needs to accept a noised image at some timestep and return the predicted noise. This project uses a U-Net architecture for this purpose, which is a symmetrical architecture that has identically sized input and output. This architecture also includes skip connections between encoder and decoder blocks of the same dimensions, which improves gradient flow [5]. Additionally, this project utilizes some specific improvements proposed by OpenAI [6].



The model also makes use of classifier-free diffusion guidance, which helps to prevent posterior collapse [2]. This means that the model takes class labels into account, and is therefore able to explicitly differentiate between classes. Since the model already contains timesteps as a condition, learned embeddings of the conditional labels are added to the existing timestep embeddings. The model is trained conditionally for 90% of the total training time, with the remaining 10% used to train unconditionally. This way the model is able to sample in both ways, which can lead to improved results [2].

1.4 Sampling

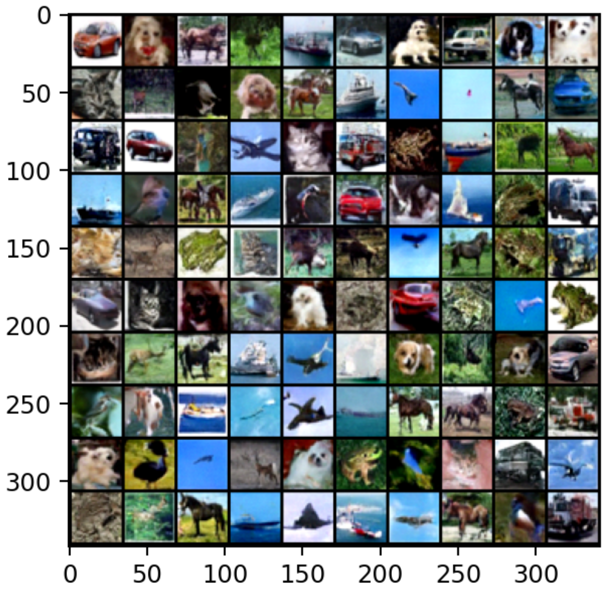
In order to perform image generation, Algorithm 2 from [1] is used, with a modification to utilize classifier-free guidance. This modification essentially means sampling both conditionally and unconditionally, then linearly interpolating from the unconditional sample towards the conditional sample at every iteration.

1.5 Parameters

The method described above was trained on the CIFAR-10 dataset for approximately 1 million iterations, with a batch size of 14. Also, AdamW optimization was used with a learning rate of 3×10^{-4}. These parameter settings were suggested in [6].

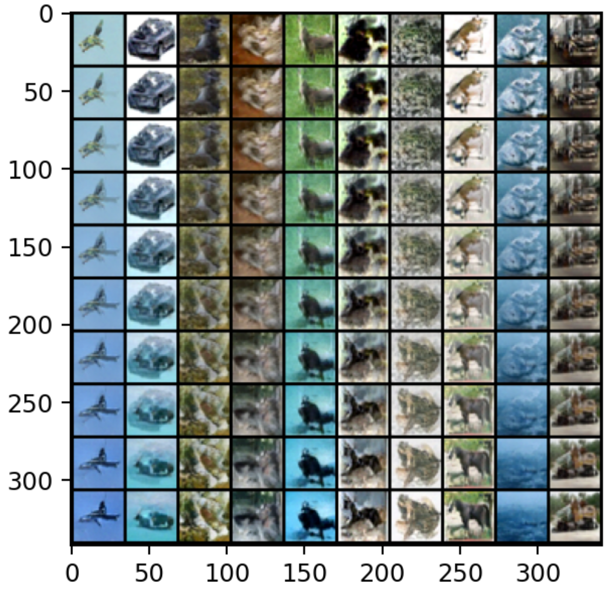
2 Results

The results seem to be quite positive, with a significant number of realistic images being generated. Some classes from the dataset seem to be more successful than others, with cars, horses and frogs being the most consistently realistic. A random batch of non-cherry picked samples looks like this:



Similarities between the training data were measured by comparing sampled images with their nearest neighbours in the training data. Using the lpips metric with the 'alex' net, an average score of 0.13 was achieved across 100 random samples. This indicates a significant difference between the samples and the original training data.

The next results show images generated by interpolating between points in the latent space:



And here are some cherry-picked samples that show the best outputs the model has generated:

bird

horse

truck

boat

3 Limitations

The samples generally look quite realistic, however the resolution is noticeably low. Future work could look at re-configuring the network to accommodate for higher resolutions such as 128x128. However, large amounts of computational resource and training time will be required to process this amount of data. To work around this, an architecture that is more suited to scaling up to these resolutions would need to be implemented. An example of this would be a cascade diffusion model [7].

Bonuses

This submission has a total bonus of -4 marks (a penalty), as it is trained only on CIFAR-10.

References

[1] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising diffusion probabilistic models”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6840–6851.

[2] Jonathan Ho and Tim Salimans. “Classifier-free diffusion guidance”. In: *arXiv preprint arXiv:2207.12598* (2022).

[3] Paul A Gagniuc. *Markov chains: from theory to implementation and experimentation.* John Wiley & Sons, 2017.

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

[5] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

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

[7] Jonathan Ho et al. “Cascaded Diffusion Models for High Fidelity Image Generation.” In: *J. Mach. Learn. Res.* 23.47 (2022), pp. 1–33.