Diffusion Models for Image Generation

Anonymous author

Abstract

This paper explains diffusion models and their usage in image generation. I started with a simple implementation, which I improved in order to produce reasonable results. The improvements involved the model's architecture, the scheduling of the diffusion, implementing conditional guided diffusion, exponential moving average and dynamic thresholding. In order to train the model CIFAR-10 and STL-10 datasets were utilised. Some results are exhibited and the limitations of this model are discussed.

1 Methodology

1.1 Introduction to Diffusion Models

Over the last few years generative models have experienced tremendous growth [18] with Generative Adversarial Networks (GANs) [10] having the best results in terms of image quality [3]. Recently, diffusion models have become highly successful [2, 6, 15], as they have been shown to achieve superior image quality than the one of GANs [7] and offer great sample diversity [31]. An application of these genre of models is image generation from text prompts like [27], [24] and [25] that have produced state-of-the-art images.

The first time diffusion models were introduced was with [29], but were not particularly popular until [12] showed major improvements in performance.

Following the derivations of [12], diffusion models involve two processes. First, we have the forward diffusion, which is an iterative process with T steps It gradually adds noise in an image x_0 from the data distribution $q(.)$ [6], according to a variance schedule $\beta_1, \dots, \beta_T$, until it becomes pure noise. We choose the cosine schedule as a simple linear schedule is not optimal for images of resolution 32x32 [19]. This diffusion process, is Markovian and follows the equation:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) \coloneqq \mathcal{N}(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}) \tag{1}$$

Further, we can define $\alpha \coloneqq 1 - \beta_t$ and $\bar{\alpha}_t \coloneqq \prod_{s=1}^{t} \alpha_s$ and write the marginal:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\boldsymbol{I}) \tag{2}$$

Intuitively, we want α_t for small t to be close to 1, as x_t should be similar to x_0. Conversely, for large t, α_t should be close to 0, so that x_t resembles a realisation from an isotropic Gaussian distribution.

On the other hand, we have the reverse diffusion process which begins with some random noise x_T from an isotropic normal distribution $\mathcal{N}(0, 1^2)$ and attempts to reconstruct an image by estimating some noise in each step, using a deep neural network [28]. The problem is that $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ is intractable as it depends on the entire data distribution [19]. However, if we further condition on $\boldsymbol{x}_0$ we can write q in closed form [22]:

$$q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0), \tilde{\beta}_t \boldsymbol{I}) \tag{3}$$

where $\tilde{\boldsymbol{\mu}}_t$ can be expressed in the form

$$\tilde{\boldsymbol{\mu}}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) = \frac{1}{\sqrt{\alpha_t}} \left(\boldsymbol{x}_t(\boldsymbol{x}_0, \boldsymbol{\epsilon}) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon} \right) \tag{4}$$

We can approximate $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ using a deep neural network with the following equation:

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

We set $\boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t)$ to $\beta_t \boldsymbol{I}$ as in [12], but we note other approaches like [19], where they learn the variance as an interpolation between β_t and $\tilde{\beta}_t := \frac{1-\tilde{\alpha}_{t-1}}{1-\tilde{\alpha}}\beta_t$. We choose to train the model using the variational lower bound:

$$L_{vlb} := -\log(p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)) + \sum_{t=2}^{T-1} D_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)\|p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)) + D_{\mathrm{KL}}(q(\boldsymbol{x}_T|\boldsymbol{x}_0)\|p(\boldsymbol{x}_t)) \tag{6}$$

which if we choose the parameterization $\boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)\right)$ and follow some simplifications that produce better images, in practise [12], we arrive to:

$$L_{simple} := \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{\epsilon}}[||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)||^2] \tag{7}$$

where $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$ is an approximator to predict $\boldsymbol{\epsilon}$ using $\boldsymbol{x}_t$, that we train using a deep neural network.

Finally, the training and sampling algorithms are [12]:

Algorithm 1 Training

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

Algorithm 2 Sampling

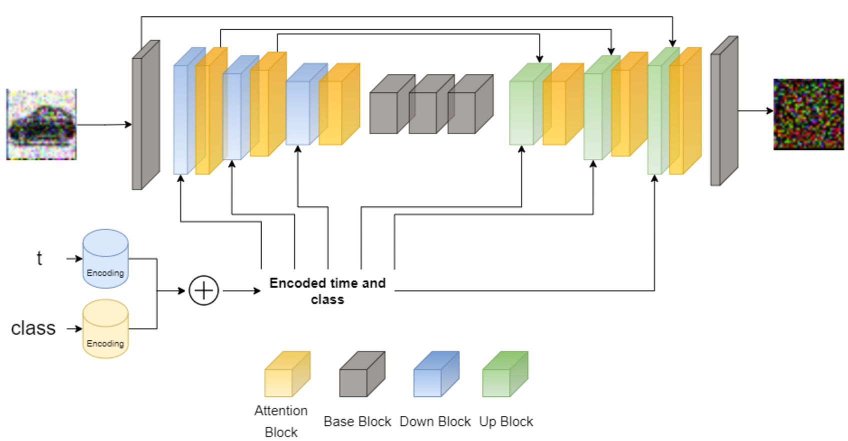
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\quad \mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,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

1.2 Improvements

1.2.1 Architecture

The neural network we use to predict the noise is a U-Net [26]. Recently, there have been many improvements to the basic U-Net architecture like implementing attention blocks [32, 9, 21], recurrent residual connections [1] a mixture of both [5, 8] or even more complicated architectures [4].

In our implementation we followed closely the architecture of [23] and [7]. We implemented multiple Self-Attention blocks, dropout layers, tried rescaling residual connections with $\frac{1}{\sqrt{2}}$ (which produced worse images despite the results in [7]) and more. We also attempted different depths for the network, but they don't imporove its performance. For some detailed diagrams view the GitHub repository of this project[1]. The following diagram [17] shows an outline of the architecture of our U-Net :



[1] https://github.com/zedt123/MyCode/blob/main/deeplearning

Table 1: Parameter fitting

PARAMETER	FID/LPIPS
UNet depth 3 vs 4	12.5/0.0583 vs 19.2/0.0655
Dropout 0.0 vs 0.1	18.8/0.0628 vs 12.5/0.0583
Learning rate 1e-4 vs 3e-4	13.2/0.0514 vs 12.5/0.0583

1.2.2 Conditional Guided Diffusion

Conditional-free diffusion guidance (CFG) was first introduced with [13]. Before this, the state-of-the-art approach was to train an extra classifier [7].

CFG is a technique that jointly trains the model on both conditional (using the labels of the images) and unconditional objectives [27]. We drop with 10% probability the label of the image as in [27, 23]. During sampling we replace the predicted noise of the basic model with:

$$\tilde{\boldsymbol{\epsilon}}_\theta := w\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, c) + (1 - w)\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t) \tag{8}$$

where w is the guidance weight and $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t, c)$ is the $\boldsymbol{\epsilon}$ prediction using the label of the image.

1.2.3 Dynamic Thresholding

A downside of CFG when using a high guidance weight is that it produces saturated images. According to [27] this is due to a test-train mismatch; the training images are scaled initially in the range $[-1, 1]$, but high values for w, cause the output to exceed this bound by a large margin [20]. A solution to this is dynamic thresholding, which was introduced in [27].

This method rescales the images in each sampling step (if pixels are outside the desired range), by dividing by a percentile of of the largest absolute pixel value.

1.2.4 Exponential Moving Average

Exponential moving average (EMA) is a way of enforcing a smoother training [23]. Its effectiveness in image generation has been proved by [34]. EMA creates a deep copy of the model's parameters and updates the copy using the moving average of the main model using the following formula:

$$\theta_{EMA}^{(t)} = \rho\theta_{EMA}^{(t-1)} + (1 - \rho)\theta^{(t)} \tag{9}$$

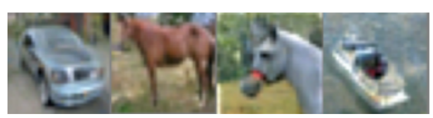
where $\theta_{EMA}^{(t)}$ are the parameters of the deep copied model, $\theta^{(t)}$ are the parameters of the main model and ρ is the EMA rate. We also choose to update the EMA parameters after 2000 iterations to give the main model a quick warm-up [23], i.e. $\theta_{EMA}^{(2000)} = \theta^{(2000)}$

1.2.5 Parameter Fitting

We fit the parameters by running the model on a small amount of epochs (100) and assuming that it generalises to an arbitrary number of iterations, as we have poor computational capabilities. Table 1 indicates some of the experiments we run. We then choose the best parameters and train for 400 epochs [11].

2 Results

We present our results. The following images show a random batch of non-cherry picked samples, images generated by interpolating between points in the latent space and some cherry-picked samples that show the best outputs the model has generated.



3 Limitations

Diffusion models in general have some limitations. They are slow to sample from [2, 33] and as they depend on a long Markov chain they are computationally more expensive in comparison to other models [28]. Our implementation can be further enhanced in many ways (as we can see the interpolations are poor). One can further improve the architecture, by using cascaded diffusion [14] or speed up the sampling process [30, 19]. The model sometimes fails to produce images, due to the small training time (as we didn't have large GPU capacities). Finally, some other improved metrics [16] could of been implemented to better assess the model.

Bonuses

This submission has no bonuses, as we trained with CIFAR-10 and STL-10 resized to 32x32 pixels.

References

[1] Md. Zahangir Alom et al. "Recurrent Residual Convolutional Neural Network based on U-Net (R2U-Net) for Medical Image Segmentation". In: *CoRR* abs/1802.06955 (2018). arXiv: 1802.06955. URL: http://arxiv.org/abs/1802.06955.

[2] András Béres. *Denoising Diffusion Implicit Models*. 2022. URL: https://keras.io/examples/generative/ddim/.

[3] Sam Bond-Taylor et al. "Deep Generative Modelling: A Comparative Review of VAEs, GANs, Normalizing Flows, Energy-Based and Autoregressive Models". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.11 (Nov. 2022), pp. 7327–7347. DOI: 10.1109/tpami.2021.3116668. URL: https://doi.org/10.1109%2Ftpami.2021.3116668.

[4] Jieneng Chen et al. "TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation". In: *CoRR* abs/2102.04306 (2021). arXiv: 2102.04306. URL: https://arxiv.org/abs/2102.04306.

[5] Xiaocong Chen, Lina Yao, and Yu Zhang. *Residual Attention U-Net for Automated Multi-Class Segmentation of COVID-19 Chest CT Images*. 2020. DOI: 10.48550/ARXIV.2004.05645. URL: https://arxiv.org/abs/2004.05645.

[6] Kirill Demochkin. *75: Guided Diffusion*. 2021. URL: https://www.casualganpapers.com/guided_diffusion_langevin_dynamics_classifier_guidance/Guided-Diffusion-explained.html.

[7] Prafulla Dhariwal and Alex Nichol. "Diffusion Models Beat GANs on Image Synthesis". In: *CoRR* abs/2105.05233 (2021). arXiv: 2105.05233. URL: https://arxiv.org/abs/2105.05233.

[8] DigitalSreeni. *226 - U-Net vs Attention U-Net vs Attention Residual U- Net - should you care?* 2021. URL: https://www.youtube.com/watch?v=L5iV5BHkMzM.

[9] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *CoRR* abs/2010.11929 (2020). arXiv: 2010.11929. URL: https://arxiv.org/abs/2010.11929.

[10] Ian J. Goodfellow et al. *Generative Adversarial Networks*. 2014. DOI: 10.48550/ARXIV.1406.2661. URL: https://arxiv.org/abs/1406.2661.

[11] Martin Heusel et al. "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models". In: *CoRR* abs/2006.11239 (2020). arXiv: 2006.11239. URL: https://arxiv.org/abs/2006.11239.

[13] Jonathan Ho and Tim Salimans. *Classifier-Free Diffusion Guidance*. 2022. DOI: 10.48550/ARXIV.2207.12598. URL: https://arxiv.org/abs/2207.12598.

[14] Jonathan Ho et al. "Cascaded Diffusion Models for High Fidelity Image Generation". In: *CoRR* abs/2106.15282 (2021). arXiv: 2106.15282. URL: https://arxiv.org/abs/2106.15282.

[15] Yannic Kilcher. *DDPM - Diffusion Models Beat GANs on Image Synthesis (Machine Learning Research Paper Explained)*. 2021. URL: https://www.youtube.com/watch?v=W-O7AZNzbzQ.

[16] Tuomas Kynkäänniemi et al. "Improved precision and recall metric for assessing generative models". In: (2019). URL: https://https://arxiv.org/abs/1904.06991.

[17] Kenneth Leung. *How to Easily Draw Neural Network Architecture Diagrams*. 2021. URL: https://towardsdatascience.com/how-to-easily-draw-neural-network-architecture-diagrams-a6b6138ed875.

[18] A K Nain. *Denoising Diffusion Probabilistic Models*. 2022. URL: https://keras.io/examples/generative/ddpm/.

[19] Alex Nichol and Prafulla Dhariwal. "Improved Denoising Diffusion Probabilistic Models". In: *CoRR* abs/2102.09672 (2021). arXiv: 2102.09672. URL: https://arxiv.org/abs/2102.09672.

[20] Ryan O'Connor. *How Imagen Actually Works*. 2022. URL: https://www.assemblyai.com/blog/how-imagen-actually-works/.

[21] Ozan Oktay et al. "Attention U-Net: Learning Where to Look for the Pancreas". In: *CoRR* abs/1804.03999 (2018). arXiv: 1804.03999. URL: http://arxiv.org/abs/1804.03999.

[22] Outlier. *Diffusion Models — Paper Explanation — Math Explained*. 2022. URL: https://www.youtube.com/watch?v=HoKDTa5jHvg&t=3s.

[23] Outlier. *Diffusion Models — PyTorch Implementation*. 2022. URL: https://www.youtube.com/watch?v=TBCRlnwJtZU.

[24] Aditya Ramesh et al. *Hierarchical Text-Conditional Image Generation with CLIP Latents*. 2022. DOI: 10.48550/ARXIV.2204.06125. URL: https://arxiv.org/abs/2204.06125.

[25] Robin Rombach et al. "High-Resolution Image Synthesis with Latent Diffusion Models". In: *CoRR* abs/2112.10752 (2021). arXiv: 2112.10752. URL: https://arxiv.org/abs/2112.10752.

[26] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597 (2015). arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

[27] Chitwan Saharia et al. *Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding*. 2022. DOI: 10.48550/ARXIV.2205.11487. URL: https://arxiv.org/abs/2205.11487.

[28] J. Rafid Siddiqui. *Diffusion Models Made Easy*. 2021. URL: https://towardsdatascience.com/diffusion-models-made-easy-8414298ce4da.

[29] Jascha Sohl-Dickstein et al. "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". In: *CoRR* abs/1503.03585 (2015). arXiv: 1503.03585. URL: http://arxiv.org/abs/1503.03585.

[30] Jiaming Song, Chenlin Meng, and Stefano Ermon. "Denoising Diffusion Implicit Models". In: *CoRR* abs/2010.02502 (2020). arXiv: 2010.02502. URL: https://arxiv.org/abs/2010.02502.

[31] Arash Vahdat, Karsten Kreis, and Ruiqi Gao. *Tutorial on Denoising Diffusion-based Generative Modeling: Foundations and Applications*. 2022. URL: https://www.youtube.com/watch?v=cS6JQpEY9cs.

[32] Ashish Vaswani et al. "Attention Is All You Need". In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

[33] Lilian Weng. "What are diffusion models?" In: *lilianweng.github.io* (July 2021). URL: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/.

[34] Yasin Yazıcı et al. "The Unusual Effectiveness of Averaging in GAN Training". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=SJgw_sRqFQ.