Diffusion models

Denoising Diffusion Probabilistic Models

Maciej Żelaszczyk

November 30, 2022

PhD Student in Computer Science
Division of Artificial Intelligence and Computational Methods
Faculty of Mathematics and Information Science

m.zelaszczyk@mini.pw.edu.pl

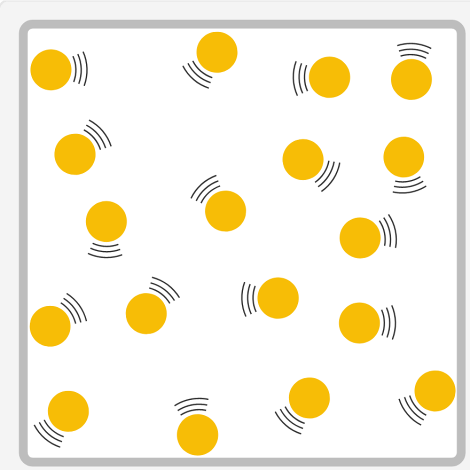
Warsaw University of Technology

Starting points

Some possible starting points for *diffusion models*:

- Physics
- VAEs
- GANs

Physics



Source: IXL

Generative models

Models:

- Discriminative: $P(Y|X = x)$
- Generative. Joint probability distribution: $X \times Y, P(X, Y)$
- No hard demarcation line.

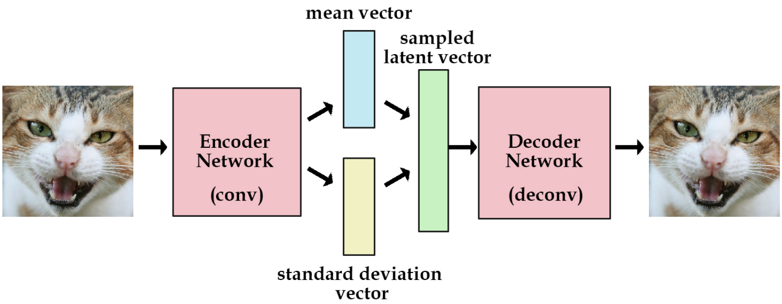
Standard generative models in deep learning:

- Autoencoders.
- Variational autoencoders (VAEs).
- Generative adversarial networks (GANs).

Variational Autoencoders

Introduced in [Kingma and Welling, 2014]:

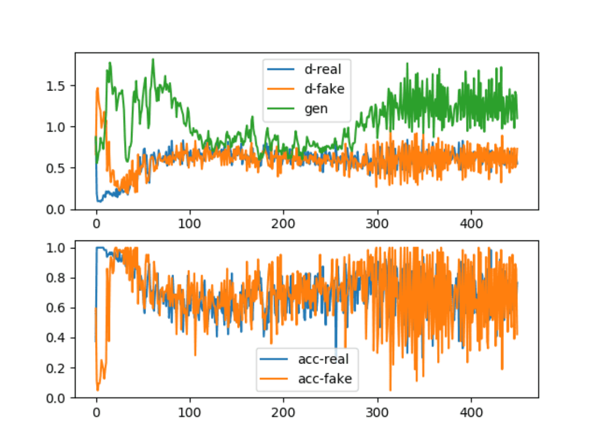
- Latent variable matches unit Gaussian.
- Loss = generation loss + KL divergence.



Source: Frans, K., *Variational Autoencoders Explained*

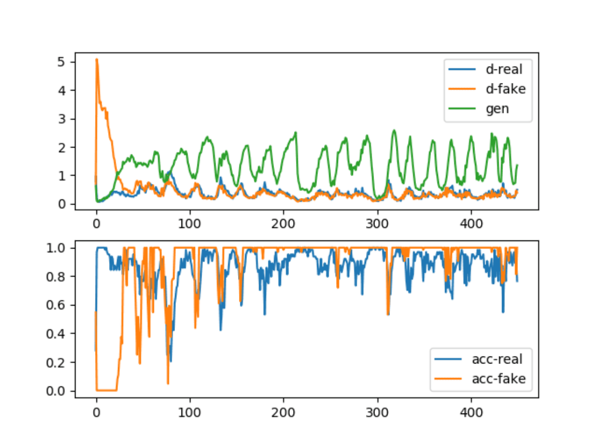
Generative Adversarial Nets

Introduced in [Goodfellow et al., 2014]. Loss functions do not have an immediately intuitive interpretation.



Source: Brownlee, J., *How to Identify and Diagnose GAN Failure Modes*

Generative Adversarial Nets



Source: Brownlee, J., *How to Identify and Diagnose GAN Failure Modes*

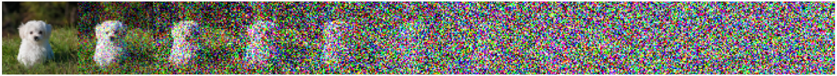
Diffusion as a generative model

Hypothetical steps:

- Successfully apply a forward process to a complex data distribution.
- Arrive at a convenient target distribution.
- Employ a reverse process to move from the target distribution to the initial one.
- Part of architecture: target $\rightarrow$ initial.
- Treat target distribution as a sampling distribution.
- The architecture accounts for the joint distribution – generative model.

Diffusion for vision

How can a forward/reverse process look for images?

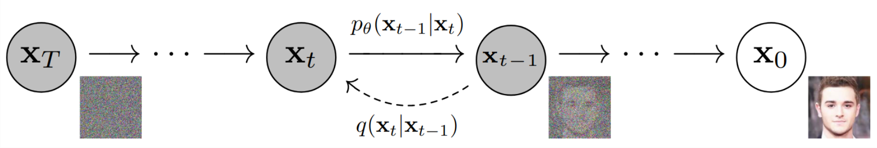


Source: [Nichol and Dhariwal, 2021].

Preliminaries

- Sequential process with steps indexed by t.
- $\mathbf{x}_t$ – image at timestep t.
- $\mathbf{x}_0$ – initial (uncorrupted) image.
- $\mathbf{x}_T$ – final (noise) image.
- $q(\mathbf{x}_t|\mathbf{x}_{t-1})$ – forward process.
- $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ – reverse process.

Forward/reverse process for images



Source: [Ho et al., 2020].

Forward process

- The data (initial) distribution is defined as $q(\mathbf{x}_0)$.
- A step in the forward process follows an isotropic Gaussian:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right) \tag{1}$$

- The variance follows a schedule: $\beta_1, \dots, \beta_T$.
- The actual forward process corresponds to the posterior:

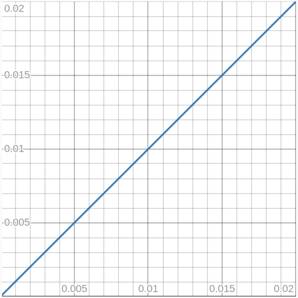
$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{2}$$

- The mean of the forward process posterior $\widetilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0)$ satisfies:

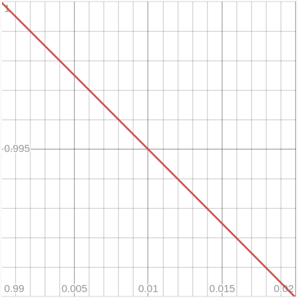
$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_{t-1}; \widetilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0), \widetilde{\beta}_t \mathbf{I}\right) \tag{3}$$

β schedule & image scaling

- One β schedule: $\beta_1 = 0.0001$ grows linearly to $\beta_T = 0.02$ [Ho et al., 2020].
- Using a naive mean $\mathbf{x}_{t-1}$ for the Gaussian process could explode the image.
- Hence, the scaling factor $\sqrt{1-\beta_t}$ is introduced.



β_t



$\sqrt{1-\beta_t}$

Reverse process

- The target (noise) distribution is $p_\theta(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$.
- A step in the reverse process follows a Gaussian:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)) \tag{4}$$

- The actual reverse process corresponds to the posterior:

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \tag{5}$$

- Both q and p_θ have the same functional form when β_t are small [Sohl-Dickstein et al., 2015].

Diffusion model

A diffusion model can be stated as:

$$p_\theta(\mathbf{x}_0) = \int p_\theta(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T} \tag{6}$$

Making it operational

- Given the choice of the distribution and the β schedule, the forward process $q(\mathbf{x}_t|\mathbf{x}_{t-1})$ can be computed for an arbitrary number of steps.
- For the reverse process $p_\theta(\mathbf{x}_{t-1})$ to satisfy the assumptions, choices have to be made for $\mu_\theta(\mathbf{x}_t, t)$ and $\Sigma_\theta(\mathbf{x}_t, t)$.
- A possible choice for the covariance matrix is $\Sigma_\theta(\mathbf{x}_t, t) = \sigma_t^2\mathbf{I}$, where $\sigma_t^2 = \beta_t$ – works for $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.
- For now, we treat $\mu_\theta(\mathbf{x}_t, t)$ as a predictor with learnable parameters θ.
- A training loss can be formulated:

$$L_{t-1} = \mathbb{E}_q \left[\frac{1}{2\sigma_t^2} \|\widetilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \mu_\theta(\mathbf{x}_t, t)\|^2 \right] + C \qquad (7)$$

Sampling from the forward process

- To naively sample from the forward process for an arbitrary step t, sampling from $q(\mathbf{x}_t|\mathbf{x}_{t-1})$ would have to be chained.
- If we denote $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$, we can write:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right) \tag{8}$$

- This admits direct sampling at an arbitrary step t.
- We can use the reparametrization trick [Kingma and Welling, 2014]:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + (1-\bar{\alpha}_t)\epsilon, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{9}$$

Loss function

Final loss for training

$$\mathbb{E}_{\mathbf{x}_0,\epsilon}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\|\epsilon-\epsilon_\theta(\mathbf{x}_t,t)\|^2\right] \tag{10}$$

where ϵ_θ is a model predicting ϵ from $\mathbf{x}_t$.

This uses both direct sampling from the forward process and the reparametrization trick. In reality, the $\|\epsilon-\epsilon_\theta(\mathbf{x}_t,t)\|^2$ term is sufficient for training.

Training/sampling

Algorithm 1 Training

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

Algorithm 2 Sampling

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

Source: [Ho et al., 2020].

Predictors

Various predictor formulations possible:

$$(\mathbf{x}_t, t) \longrightarrow P_\theta \longrightarrow \hat{\epsilon}$$

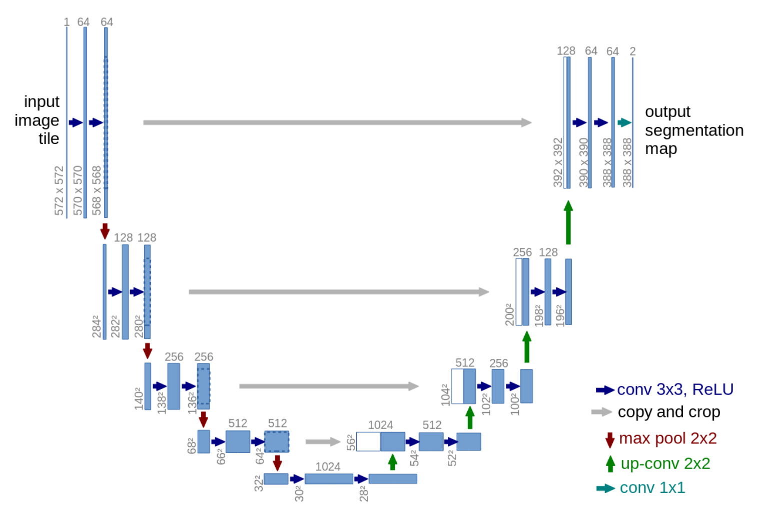
$$(\mathbf{x}_t, t) \longrightarrow P_\theta \longrightarrow \hat{\tilde{\mu}}_t$$

$$(\mathbf{x}_t, t) \longrightarrow P_\theta \longrightarrow \hat{\mathbf{x}}_{t-1}$$

$$(\mathbf{x}_t, t) \longrightarrow P_\theta \longrightarrow \hat{\mathbf{x}}_0$$

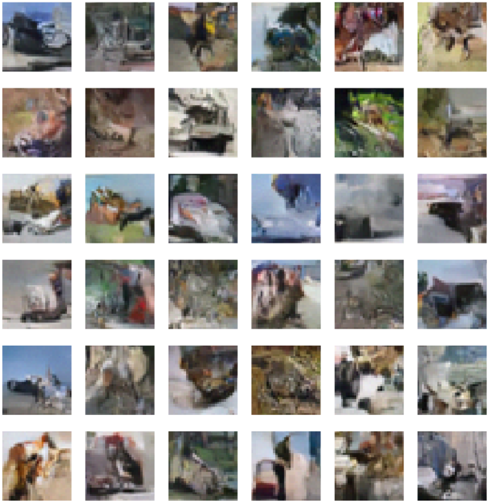
- Predicting the original image is not viable.
- Predicting the noise is a popular choice.

U-Net



Source: [Ronneberger et al., 2015].

Samples



Source: [Sohl-Dickstein et al., 2015].

Samples

Source: [Ho et al., 2020].

Improvements

Some improvements:

- Learning $\Sigma_\theta(\mathbf{x}_t, t)$.
- Cosine noise schedule.

Source: [Nichol and Dhariwal, 2021].

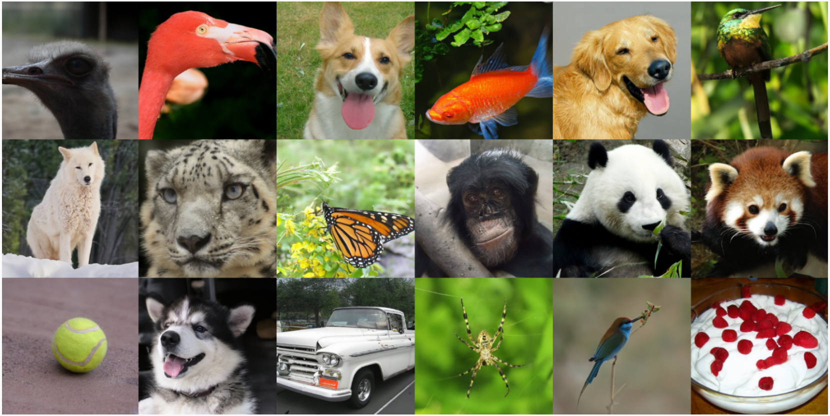
- Gradient noise reduction.
- Sampling speed.

Improvements

Some improvements:

- Architecture size.
- BigGAN blocks [Brock et al., 2018].
- Alternative sampling schemes with fewer steps.

Samples



Source: [Dhariwal and Nichol, 2021].

SOTA

Rank	Model	FID	Inception score	Paper	Code	Result	Year
1	**StyleGAN-XL**	2.3		StyleGAN-XL: Scaling StyleGAN to Large Diverse Datasets			2022
2	**BIGRoC-gt** (Guided-Diffusion)	3.63	260.02	BIGRoC: Boosting Image Generation via a Robust Classifier			2021
3	**BIGRoC-pl** (Guided-Diffusion)	3.69	249.91	BIGRoC: Boosting Image Generation via a Robust Classifier			2021
4	**RQ-Transformer**	3.83	317.1	Autoregressive Image Generation using Residual Quantization			2022
5	**ADM-G, ADM-U**	3.94	215.84	Diffusion Models Beat GANs on Image Synthesis			2021
6	**ADM-G + EDS** (ED-DPM, classifier_scale=0.75)	3.96	217.25	Entropy-driven Sampling and Training Scheme for Conditional Diffusion Generation			2022
7	**MaskGIT** (a=0.05)	4.02	355.6	MaskGIT: Masked Generative Image Transformer			2021
8	**ADM-G + EDS + ECT** (ED-DPM, classifier_scale=1.0)	4.09	221.57	Entropy-driven Sampling and Training Scheme for Conditional Diffusion Generation			2022
9	**ViT-VQGAN**	4.17	175.1				
10	**ADM-G**	4.59	186.7	Diffusion Models Beat GANs on Image Synthesis			2021

Source: PapersWithCode.

Usage

vibrant portrait painting of Salvador Dalí with a robotic half face — a shiba inu wearing a beret and black turtleneck — a close up of a handpalm with leaves growing from it

Source: [Ramesh et al., 2022].

Usage

a dolphin in an astronaut suit on saturn, artstation

a propaganda poster depicting a cat dressed as french emperor napoleon holding a piece of cheese

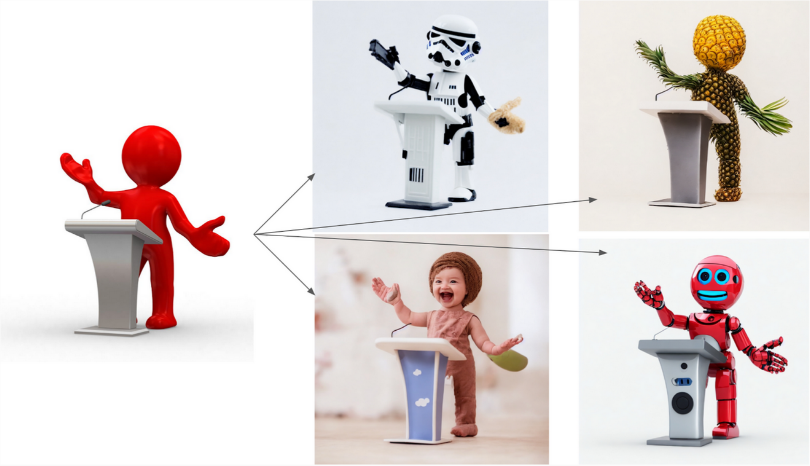
a teddy bear on a skateboard in times square

Source: [Ramesh et al., 2022].

Usage

Source: StableDiffusion 2.0, based on [Rombach et al., 2022].

Usage



Source: StableDiffusion 2.0, based on [Rombach et al., 2022].

Usage

Source: StableDiffusion 2.0, based on [Rombach et al., 2022].

Usage

Source: StableDiffusion 2.0, based on [Rombach et al., 2022].

Usage

Source: StableDiffusion 2.0, based on [Rombach et al., 2022].

Usage

Source: StableDiffusion 2.0, based on [Rombach et al., 2022].

Usage

Source: StableDiffusion 2.0, based on [Rombach et al., 2022].

Brock, A., Donahue, J., and Simonyan, K. (2018).
Large scale gan training for high fidelity natural image synthesis.
arXiv.

Dhariwal, P. and Nichol, A. (2021).
Diffusion models beat gans on image synthesis.
In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 8780–8794. Curran Associates, Inc.

Goodfellow, I. J., Pouget-Abadie, J., et al. (2014).
Generative adversarial networks.
NIPS.

Ho, J., Jain, A., and Abbeel, P. (2020).
Denoising diffusion probabilistic models.

In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc.

Kingma, D. P. and Welling, M. (2014).
Auto-encoding variational bayes.
ICLR.

Nichol, A. Q. and Dhariwal, P. (2021).
Improved denoising diffusion probabilistic models.
In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8162–8171. PMLR.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. (2022).

Hierarchical text-conditional image generation with clip latents.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. (2022).
High-resolution image synthesis with latent diffusion models.
In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695.

Ronneberger, O., Fischer, P., and Brox, T. (2015).
U-net: Convolutional networks for biomedical image segmentation.
In Navab, N., Hornegger, J., Wells, W. M., and Frangi, A. F., editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham. Springer International Publishing.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015).
Deep unsupervised learning using nonequilibrium thermodynamics.
In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France. PMLR.