Deep Generative Modeling with Backward Stochastic Differential Equations

Xingcheng Xu

Shanghai Artificial Intelligence Laboratory

April 8, 2023

Outline

1. Introduction: Diffusion Models
2. BSDE-based Generative Models
3. Experiments
4. Discussion

Introduction: Diffusion Models

Showcase: DALLE 2

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

an espresso machine that makes coffee from human souls, artstation

panda mad scientist mixing sparkling chemicals, artstation

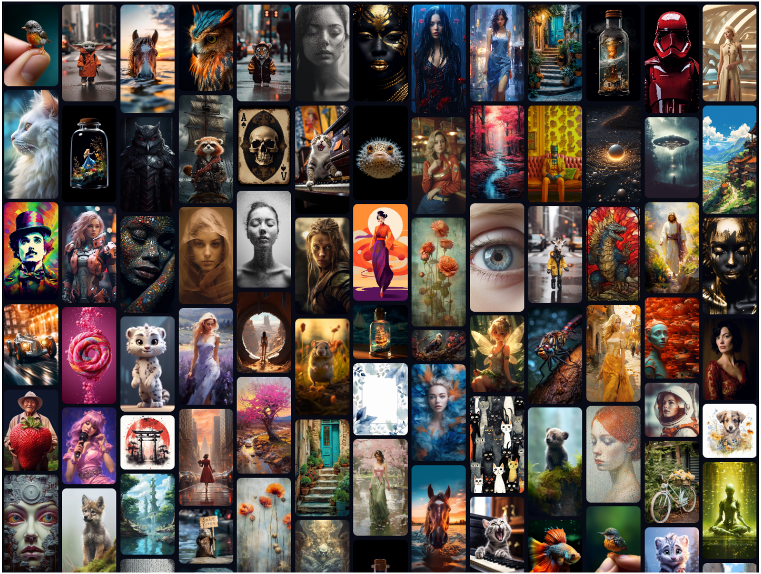
a corgi's head depicted as an explosion of a nebula

a dolphin in an astronaut suit on saturn, artstation

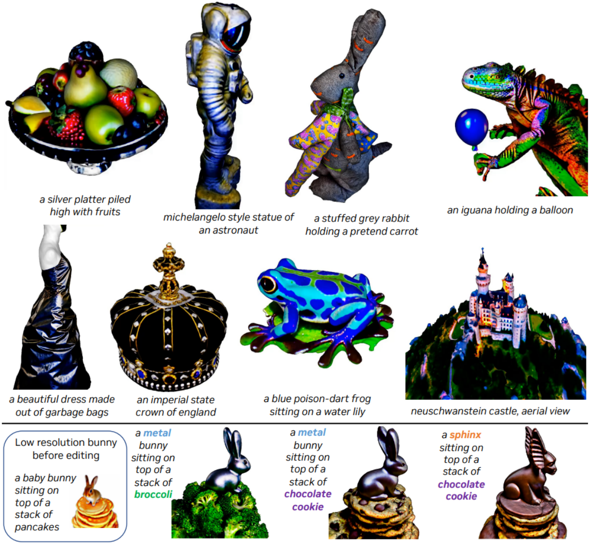
a propaganda poster depicting a cat dressed as french emperor napoleon holding a piece of cheese

a teddy bear on a skateboard in times square

Showcase: Midjourney

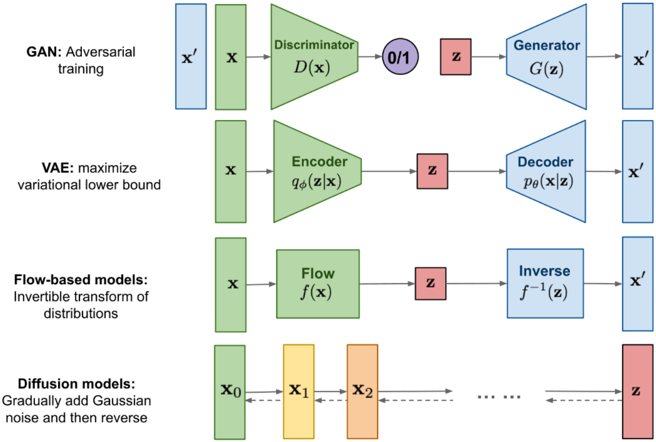


Showcase: Magic3D



Generative AI

- Classical Methodologies in the Field of Generative AI:



Source: Lilian Weng's Blog (https://lilianweng.github.io/posts/2021-07-11-diffusion-models/)

Diffusion-based Generative Models

- What are diffusion models in machine learning?

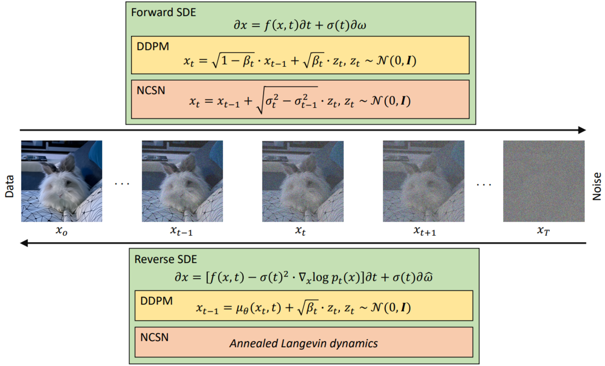


Fig. 2. A generic framework composing three alternative formulations of diffusion models based on: denoising diffusion probabilistic models (DDPMs), noise conditioned score networks (NCSNs), and stochastic differential equations (SDEs). The formulation based on SDEs is a generalization of the other two. In the forward process, Gaussian noise is gradually added to the input x_0 over T steps. In the reverse process, a model learns to restore the original input by gradually removing the noise. In the SDE formulation, the forward process is based on Eq. (11), while the reverse process is based on Eq. (12). In the DDPM version, the forward stage is based on Eq. (1), while the reverse stage uses Eq. (5). Analogously, in the NCSN version, the forward process is derived from Eq. (9), while the reverse process uses annealed Langevin dynamics. Best viewed in color.

Source: Croitoru et al. 2022, Diffusion Models in Vision: A Survey.

Diffusion-based Generative Models

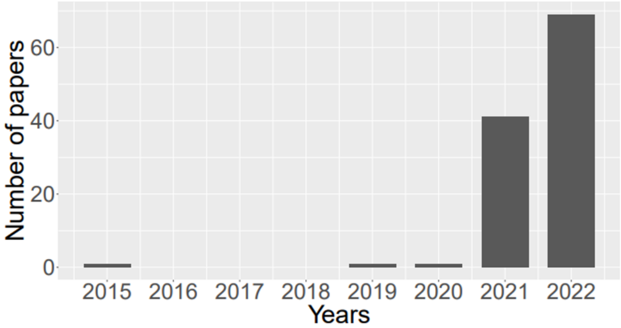


Fig. 1. The rough number of papers on diffusion models per year.

Source: Croitoru et al. 2022, Diffusion Models in Vision: A Survey.

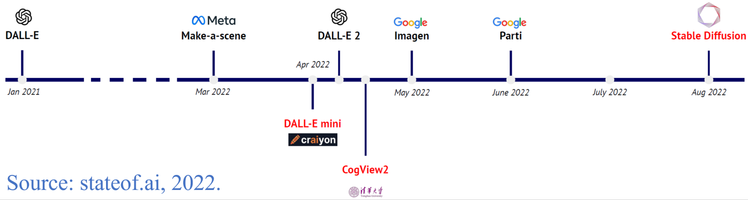
Diffusion-based Generative Models

Many diffusion-based generative models have been proposed with similar ideas underneath, including:

- *diffusion probabilistic models* (Sohl-Dickstein et al., 2015)
- *noise-conditioned score network* (NCSN; Yang & Ermon, 2019)
- *denoising diffusion probabilistic models* (DDPM; Ho et al. 2020).

Image-to-Image / Text-to-Image Generation

- *DALLE* (Ramesh et al. 2021) -- OpenAI
- *GLIDE* (DALLE 1.5) (Nichol, Dhariwal & Ramesh, et al. 2022) -- OpenAI
- *DALLE 2* (unCLIP) (Ramesh et al. 2022) -- OpenAI
- *Imagen* (Saharia et al. 2022) -- Google
- *Stable Diffusion* (Rombach et al. 2022) -- LMU, Runway; StabilityAI
- *Midjourney*, 2022-



Source: stateof.ai, 2022.

More Applications in Generative AI

- Text-to-Video Generation
  - *Make-A-Video* (Meta AI, Singer et al. (2022-09))
- 3D Generation
  - *DreamFusion*: Text-to-3D using 2D Diffusion(Google, Poole et al., (2022-09))
  - *GET3D*: A Generative Model of High Quality 3D Textured Shapes Learned from Images(NVIDIA, Gao et al., (2022-09))
  - *Magic3D*: High-Resolution Text-to-3D Content Creation(NVIDIA, Lin et al., (2022-11))

BSDE-based Generative Models

BSDEs

- Linear BSDEs, proposed by Bismut (1973)
- Pardoux and Peng (1990) established the existence and uniqueness of nonlinear BSDEs with Lipschitz condition
- Extensively studied and applied in various contexts: ...
- Applications in mathematical finance: El Karoui, Peng and Quenez (1997), Chen and Epstein (2002), ...
- Deep learning to solve BSDEs or PDEs: E, Han and Jentzen (2017, 2018), Ji, Peng, Peng and Zhang (2020, 2022), ...

BSDEs

- Consider the forward-backward stochastic differential equation (FBSDE)

$$\begin{aligned} X_t &= \zeta + \int_0^t b(s, X_s) ds + \int_0^t \sigma(s, X_s) dW_s, \\ Y_t &= \xi + \int_t^T f(s, X_s, Y_s, Z_s) ds - \int_t^T Z_s dW_s \end{aligned} \tag{1}$$

- $X_0 = \zeta$ is the initial condition, and $Y_T = \xi$ is the terminal condition.
- Solving a FBSDE involves finding the $\mathcal{F}_t$-adapted stochastic process (X_t, Y_t, Z_t) for all $t \in [0, T]$ in a suitable space that satisfies the equation above, given the functions b, σ, f, Brownian morion W_t, the initial condition ζ and the terminal condition ξ.

FBSDEs and Relationship with PDEs

- The FBSDEs are connected to semilinear parabolic PDEs through the Feynman-Kac formula under appropriate conditions. The processes Y_t and Z_t depend on the time variable t and the forward process X_t, rather than the entire path of X. Specifically,

$$Y_t = u(t, X_t) \text{ and } Z_t = \nabla u(t, X_t)^T \sigma(t, X_t),$$

  where $u(t, x)$ satisfies the corresponding PDE.
- This property is advantageous in designing deep neural networks for solving FBSDE models.

Discrete Scheme for FBSDEs

- Use the *Euler forward discrete scheme*:

$$\begin{aligned} X_{t_{n+1}} &\approx X_{t_n} + b(t_n, X_{t_n})\Delta t_n + \sigma(t_n, X_{t_n})\Delta W_{t_n} \\ Y_{t_{n+1}} &\approx Y_{t_n} - f(t_n, X_{t_n}, Y_{t_n}, Z_{t_n})\Delta t_n + Z_{t_n}\Delta W_{t_n} \end{aligned} \tag{2}$$

where $\Delta t_n = t_{n+1} - t_n$ and $\Delta W_{t_n} = W_{t_{n+1}} - W_{t_n}$.

Model Architecture

- Employ two deep neural networks to approximate the initial value Y_0 and the control process Z_t:

$$Y_0 \approx \mathcal{N}^{\theta_{Y_0}}(X_0), \quad Z_{t_n} \approx \mathcal{N}^{\theta_Z}(t_n, X_{t_n}), \tag{3}$$

  where the deep neural networks we used in our experiments in this paper are as follows:

$$\mathcal{N}^{\theta}(x) := \phi \circ \mathcal{L}_H \circ \tilde{\sigma}_{H-1} \circ \mathcal{L}_{H-1} \circ \cdots \circ \tilde{\sigma}_1 \circ \mathcal{L}_1(x),$$

  in which H is the depth of the neural network, $\mathcal{L}_h(x_h) = w_h x_h + \tilde{b}_h$ is the linear transformation, $\tilde{\sigma}_h$ is the nonlinear activation function, and ϕ is the mapping function to the state space.
- Regularization technique: dropout in neural networks.

Model Architecture

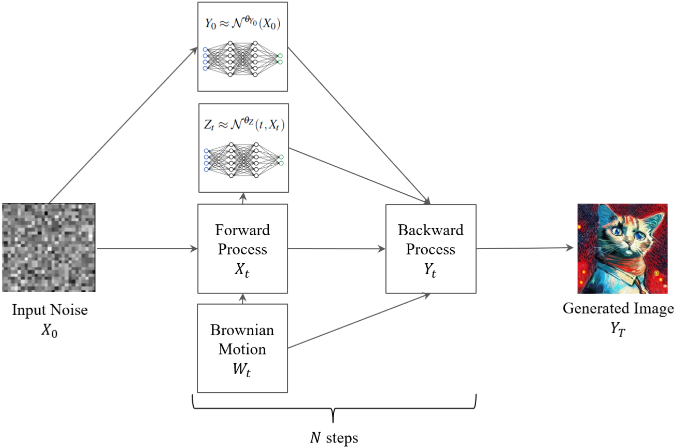


Figure: Model Architecture of the BSDE-Gen Models

Maximum Mean Discrepancy (MMD) Loss

- Assume there is a feature map $\psi : \mathcal{X} \to \mathcal{H}$ from the original space $\mathcal{X}$ to a Hilbert space $\mathcal{H}$, and the associated kernel is a function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with the property that $\langle \psi(x), \psi(y) \rangle_{\mathcal{H}} = K(x, y)$ for all x and y in $\mathcal{X}$.
- The MMD computes the distance between probability distributions as the distance between mean embeddings of features via reproducing kernel Hilbert space (RKHS) $\mathcal{H}$. Let $\mathbb{P}$ and $\mathbb{Q}$ be two probabilities of random elements on the space $\mathcal{X}$, the MMD is defined as

$$\mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}) = \|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|_{\mathcal{H}}^2,$$

  where $\mu_{\mathbb{P}} = \mathbb{E}_{X \sim \mathbb{P}}[\psi(X)]$ and $\mu_{\mathbb{Q}} = \mathbb{E}_{Y \sim \mathbb{Q}}[\psi(Y)]$ are the mean embeddings of probabilities $\mathbb{P}$ and $\mathbb{Q}$ in a RKHS $\mathcal{H}$, respectively.
- Under suitable conditions, $\mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}) = 0$ if and only if $\mathbb{P} = \mathbb{Q}$. See e.g. Gretton et al. (2012).

Training Strategy

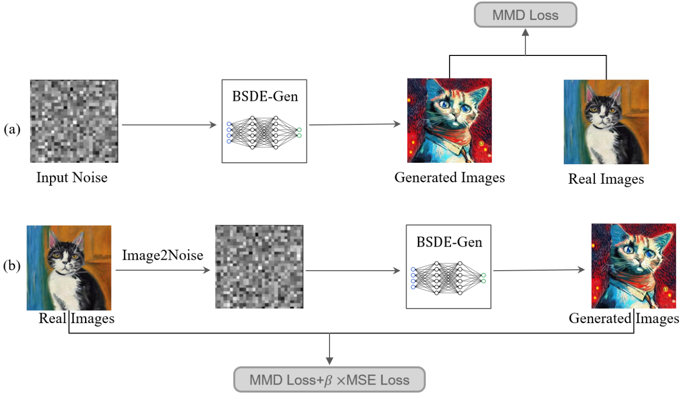


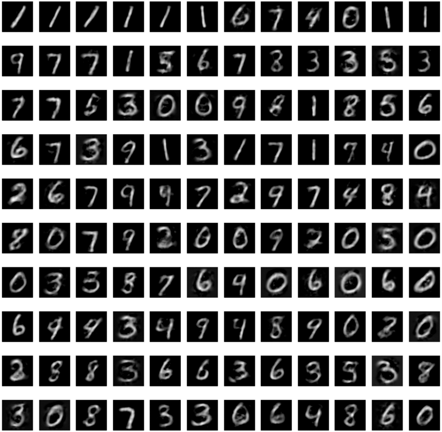
Figure: Training Strategies for BSDE-Gen Models

Experiments

Experiments

- Data: MNIST and FashionMNIST, each consisting of 60,000 grayscale images of 28×28 pixels
- The forward state process X_t is modeled as a stationary Ornstein-Uhlenbeck (OU) process starting from $d_X = 32$ dimensional standard normal distribution $\zeta \sim \mathscr{N}(0, I_{d_X})$, with the drift function $b(t,x) = -x$ and the diffusion function $\sigma(t,x) = \sqrt{2} I_{d_X}$.
- The generator function f of the backward process Y_t is defined as $f(t,x,y,z) = Ax + By + \kappa|z|$ where $|z| := (\sum_{j=1}^{d_W} |z_{ij}|)_{i=1,2,\cdots,d_Y}$, and A, B, κ are given.
- The deep neural networks $\mathcal{N}^{\theta_{Y_0}}(X_0)$ and $\mathcal{N}^{\theta_Z}(t_n, X_{t_n})$: three-hidden-layered architectures with the GELU activation function and dropout regularization with probability $p = 0.2$. The last mapping function ϕ is linear.
- Trained the BSDE-based deep generative model using the RMSprop optimizer with a learning rate of 1e-4, a batch size of 512, and 20,000 epochs with the PyTorch framework using 8 NVIDIA A100 GPUs.

Generated Examples



Discussion

Discussion

- Reduce Computational Complexity
- Build Better Diffusion Processes
  - Training under estimation of log-likelihood $\log p(Y_T)$ or score functions $\nabla \log p(Y_t)$.
- Use Better Model Architecture
  - U-Net Architecture (capable of capturing both local and global features in images)
- Conditional BSDE-Gen Models
- Extensive Applications in Machine Learning

Thanks!