ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

SCHOOL OF ENGINEERING AND ARCHITECTURE

Department of Electrical, Electronic, and Information Engineering “Guglielmo Marconi”- DEI

DEGREE PROGRAM

Electronic Technologies Big Data and Internet of Things

THESIS

In

91250 DEEP LEARNING M

THESIS TITLE

An Extensive Survey on Diffusion Models

CANDIDATE:

Jangir Kuldeep Kumar

Matricola:

0000936061

Supervisor: Prof. Andrea Asperti

Contents

List of Figures

ABSTRACT

Denoising Diffusion models are gaining growing popularity in the field of generative modeling for several reasons. These reasons include the straightforward and stable training, the outstanding generative quality, and the robust probabilistic foundation, picture synthesis, video production, and molecular design are all examples of what this tool can do. This thesis explores denoising diffusion models, which are statistical models that aim to remove noise from an image while preserving its important features. The study focuses on developing new techniques for improving the performance of denoising diffusion models, such as incorporating prior information about the image structure, designing more efficient numerical algorithms for solving the models, and evaluating the effectiveness of the denoising algorithms using various quality metrics.

The research also investigates the application of denoising diffusion models in various image processing tasks, such as image restoration, feature extraction, and segmentation. The performance of the proposed methods is evaluated on a variety of benchmark datasets, and the results demonstrate significant improvements in denoising accuracy compared to existing state-of-the-art techniques.

Overall, this thesis provides valuable insights into the development and application of denoising diffusion models, which have important applications in many fields, including medical imaging, computer vision, and remote sensing. The proposed techniques and algorithms can potentially lead to significant advances in image processing and analysis, with practical implications for improving the quality and reliability of image-based applications.

Chapter 1

INTRODUCTION

Diffusion models are a new class of state-of-the-art generative models that are capable of producing a wide variety of high-resolution images. They have already received a significant amount of attention ever since Open AI, Nvidia, and Google were successful in training large-scale models. GLIDE [100], DALLE-2[112], Imagen[120], and the full open-source stable diffusion are some examples of architectures that are based on diffusion models. They have defeated the long-standing dominance of generative adversarial networks (GANs) in the difficult task of image synthesis and have also demonstrated potential in a wide variety of fields, ranging from computer science, natural language processing, temporal data modeling, multi-modal modeling, and robust machine learning have been applied to interdisciplinary applications in fields such as computational chemistry and medical image reconstruction.

In this article, we begin by explaining the foundations of diffusion models (Chapter 2). We do so by offering a concise but self-contained introduction to the three predominant formulations: denoising diffusion probabilistic models (DDPMs), score-based generative models, and stochastic differential equations (Score SDEs). To generate new data samples, each of these methods relies on a similar principle: first, data is subjected to increasing levels of random noise (a process known as "diffusion"), and then, the noise is gradually reduced. We explain how these three models are connected and how they can be reduced to one another, as well as clarify how they work according to the same fundamental principle of diffusion.

Next, we will present a taxonomy of recent research that maps out the field of diffusion models, classifying it into three key areas: efficient sampling (Chapter 3), improved likelihood estimation (Chapter 4), and methods for handling data with special structures (Chapter 5), such as relational data, data with permutational/rotational invariance, and data residing on manifolds. Each of these key areas will be broken down into subcategories that will be described in the following We proceed with our investigation of the models by subdividing each category into even more specific subcategories, as shown in Figure 1.

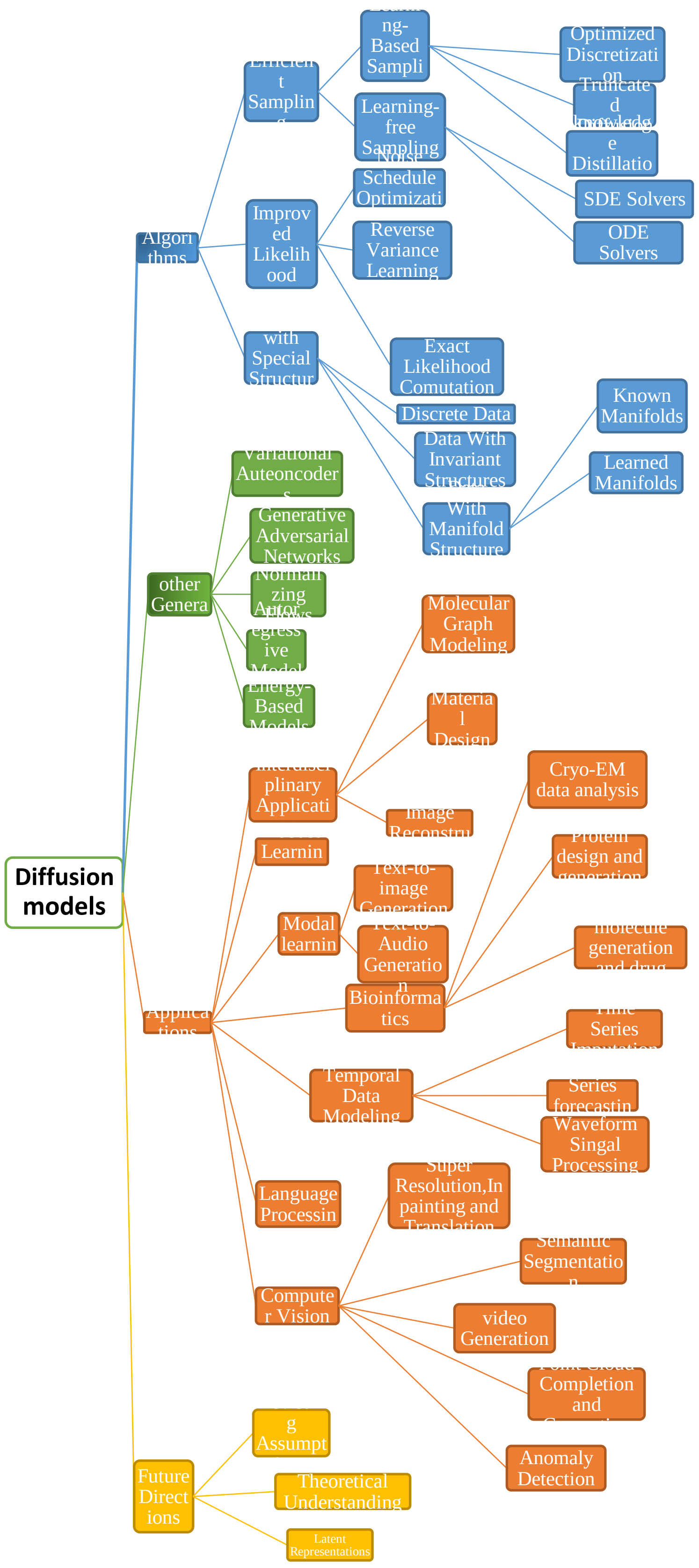

Diffusion models
Algori thms
Efficien t Samplin
ng- Based Sampli
Optimized Discretizati on
Truncate d
Knowledg e Distillatio
Learning- free Sampling
SDE Solvers
ODE Solvers
Improv ed Likelih ood
Noise Schedule Optimizati
Reverse Variance Learning
Exact Likelihood Comutation
with Special Structur
Discrete Data
Data With Invariant Structures
With Manifold Structure
Known Manifolds
Learned Manifolds
other Genera
Variational Autoencoder s
Generative Adversarial Networks
Normali zing Flows
Autor egress ive Model
Energy- Based Models
Applica tions
Interdisci plinary Applicati
Molecular Graph Modeling
Materia l Design
Image Reconstru
Learnin
Modal learnin
Text-to- image Generation
Text-to- Audio Generatio n
Bioinforma tics
Cryo-EM data analysis
Protein design and generation
molecule generation and drug
Temporal Data Modeling
Time Series Imputation
Series forecastin
Waveform Singal Processing
Language Processin
Compute r Vision
Super Resolution,In painting and Translation
Semantic Segmentatio n
video Generation
Point Cloud Completion and
Anomaly Detection
Future Direct ions
g Assumpt
Theoretical Understanding
Latent Representations

Fig. 1. The chapters titled "Typology of Diffusion Model Variants" (which can be found in Chapters 3 to5), "Links with Other Generative Models" (which can be found in Chapter 6), "Applications of Diffusion Models" (which can be found in Chapter 7), and "(in Chapter 8).

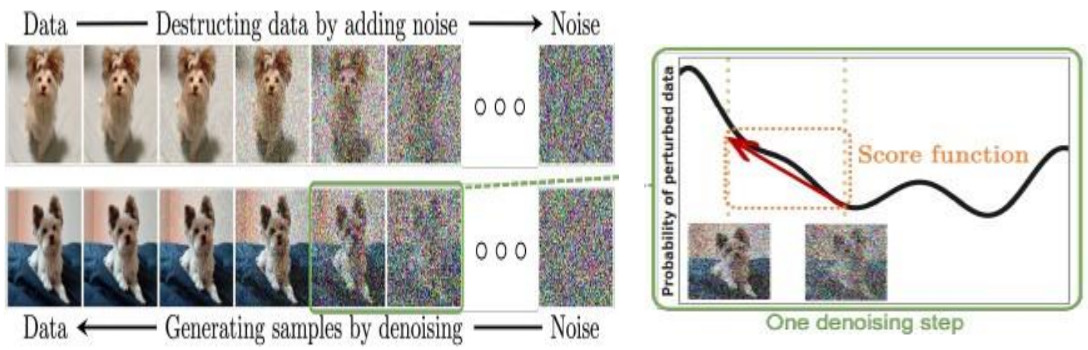

Data
Destructing data by adding noise
Noise
Data
Generating samples by denoising
Noise
Probability of perturbed data
Score function
One denoising step

Fig. 2. Diffusion models smoothly perturb data by adding noise, then reverse this process to generate new data from noise. Each denoising step in the reverse process typically requires estimating the score function (see the illustrative figure on the right), which is a gradient pointing to the directions of data with higher likelihood and less noise.

In addition, we discuss the connections of diffusion models to other deep generative models (Chapter 6), such as variational autoencoders (VAEs), generative adversarial networks (GANs), normalizing flows, autoregressive models and energy-based models (EBMs). We have the opportunity to achieve an even higher level of performance if they combine these models with diffusion models.

After that, our survey examines six major categories of applications to which diffusion models have been applied in the previous research (Chapter 7): computer vision, natural language process, temporal data modeling, multi-modal learning, robust learning, Bioinformatics and interdisciplinary applications. To accomplish each task, we first define it, then explain how diffusion models can be used to solve the problem, and finally summarize the work that has been done previously that is pertinent to the problem. In the final Chapters of this paper (Chapter 8 and 9), we provide an overview of potential future directions for this emerging field of research.

Chapter 2

The concept and foundation of diffusion models

The family of probabilistic generative models known as diffusion models progressively destroys data by inserting noise, then learns to reverse this process to generate samples. In Figure 2, we demonstrate the conceptual underpinnings of diffusion models. Denoising diffusion probabilistic models (DDPMs) [59, 99], score-based generative models (SGMs) [132, 133], and stochastic differential equations (Score SDEs) [131, 135] are the three predominant formulations that are used in the majority of the research that is being done on diffusion models today. In this Chapter, we provide an introduction to these three formulations that stand alone, while at the same time explaining how they are connected.

2.1. Denoising Diffusion Probabilistic Models (DDPMs)

Denoising Diffusion Probabilistic Models (DDPM) is a family of generative models for image and video generation. It was introduced in a 2021 paper by Ho et al. called "Denoising Diffusion Probabilistic Models"[59]. DDPM models are based on the concept of diffusion, which is a process where noise is added to an image or signal to create a sequence of noisy images. The goal of DDPM is to denoise this sequence of images to obtain a high-quality image.

DDPM models are trained using a denoising score-matching algorithm. This algorithm estimates the log-likelihood of the data distribution (for example, a standard Gaussian) by training a neural network to predict the score, which is the gradient of the log-density function. The score function is used to define a diffusion process, where the image is gradually transformed by adding noise to it.

DDPM models use the reverse diffusion process during inference to denoise the noisy images. Starting from a noisy image, the model applies a series of denoising steps to gradually remove the noise and obtain a high-quality image. The reverse diffusion process is performed using a Markov chain Monte Carlo (MCMC) sampling algorithm.

Forward diffusion process: Let's say we have a data point that was sampled from a real data distribution $\boldsymbol{x0} \sim \boldsymbol{q}(\boldsymbol{x})$. Then, let's say we want to define a forward diffusion process in which we take that sample and gradually add a small amount of Gaussian noise to it over the course of $\boldsymbol{T}$steps. This will result in a series of noisy samples $\boldsymbol{x1}, \ldots, \boldsymbol{xT}$. The step sizes are determined by a variance schedule that looks like this: $\{\beta_t \in (0,1)\}_{t=1}^{T}$.

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right) q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$$

As the size of the step increases, the distinct characteristics of the data sample x0 begin to disappear gradually. At some point in the future, when $\boldsymbol{T} \rightarrow \infty\, \boldsymbol{xT}$, the distribution will be equivalent to an isotropic Gaussian.

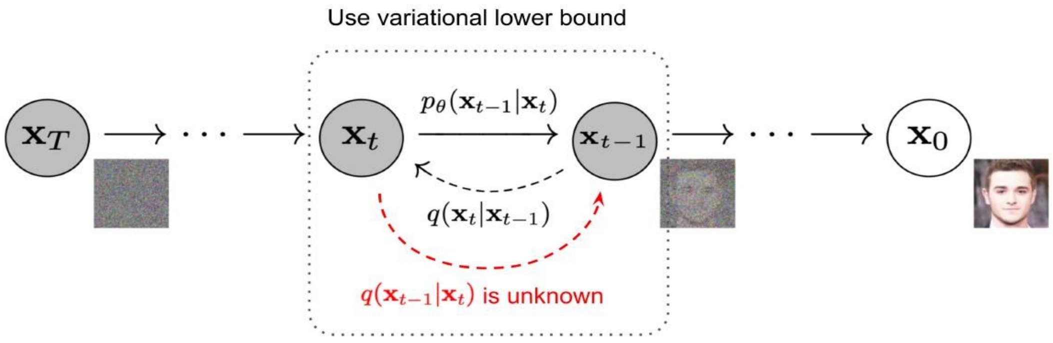

Use variational lower bound
$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$\mathbf{x}_{t-1}$
$\mathbf{x}_0$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$
$q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is unknown

Fig. 3. The Markov chain of forward (reverse) diffusion process of generating a sample by slowly adding (removing) noise. (Image source: Ho et al. 2020 with a few additional annotations). [59]

The fact that we can take a sample $\boldsymbol{xt}$ at any arbitrary time step $\boldsymbol{t}$ in a closed form by using the reparameterization trick is a useful feature of the process described above. Let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$:

$$
\begin{aligned}
\mathbf{x}_t &= \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \text{ ; where } \boldsymbol{\epsilon}_{t-1}, \boldsymbol{\epsilon}_{t-2}, \cdots \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\
&= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\boldsymbol{\epsilon}}_{t-2}; \text{ where } \bar{\boldsymbol{\epsilon}}_{t-2} \text{ merges two Gaussians } (*). \\
&= \cdots \\
&= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t\epsilon} \\
q(\mathbf{x}_t \mid \mathbf{x}_0) &= \mathcal{N}\left(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}\right)
\end{aligned}
$$

(*) Recall that when we merge two Gaussians with different variance, $\mathcal{N}(\mathbf{0}, \sigma_1^2\mathbf{I})$ and $\mathcal{N}(\mathbf{0}, \sigma_2^2\mathbf{I})$, the new distribution is $\mathcal{N}(\mathbf{0}, (\sigma_1^2 + \sigma_2^2)\mathbf{I})$. Here the merged standard deviation is $\sqrt{(1-\alpha_t) + \alpha_t(1-\alpha_{t-1})} = \sqrt{1-\alpha_t\alpha_{t-1}}$.

In most cases, we are able to make use of a more significant update step when the sample becomes more noisy, so $\beta_1 < \beta_2 < \cdots < \beta_T$ and therefore $\bar{\alpha}_1 > \cdots > \bar{\alpha}_T$.

Connection with stochastic gradient Langevin dynamics: A concept from the field of physics known as Langevin dynamics was developed for the purpose of statistically modeling molecular systems. Stochastic gradient Langevin dynamics is a technique that, when combined with stochastic gradient descent, can produce samples from a probability density $p(\mathbf{x})$ using only the gradients $\nabla_{\mathbf{x}}\log\ p(\mathbf{x})$ in a Markov chain of updates.

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \frac{\delta}{2}\nabla_{\mathbf{x}}\log p(\mathbf{x}_{t-1}) + \sqrt{\delta}\boldsymbol{\epsilon}_t, \text{ where } \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

where $\boldsymbol{\delta}$ is the step size. When $\boldsymbol{T} \to \infty, \boldsymbol{\epsilon} \to \mathbf{0}, \mathbf{x}_{\boldsymbol{T}}$ equals to the true probability density $p(\mathbf{x})$.

Stochastic gradient Langevin dynamics, in comparison to standard SGD, injects Gaussian noise into the parameter updates in order to avoid collapses into local minima.

Reverse diffusion process: If we are able to reverse the process described above and take a sample from $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ we will be able to recreate the true sample that was generated from an input of

Gaussian noise, which is denoted by $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. It is important to keep in mind that if t is sufficiently small, then $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ will also be Gaussian. Because it needs to use the entire dataset, unfortunately, we are unable to easily estimate $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$. As a result, in order to run the reverse diffusion process, we need to learn a model $\boldsymbol{p_\theta}$ that can approximate these conditional probabilities.

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\big(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\big)$$

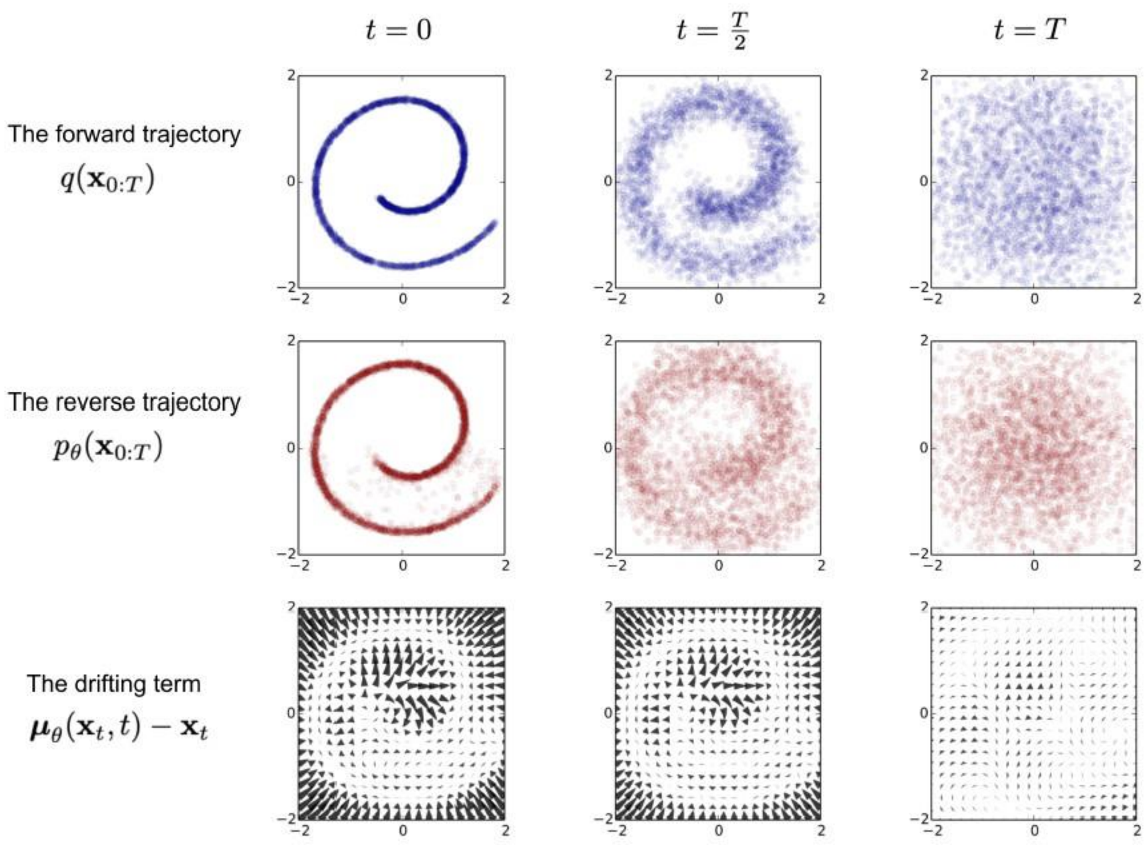


Fig. 4. An example of training a diffusion model for modeling a 2D Swiss roll data. (Image source: Sohl-Dickstein et al., 2015). [128]

It is worthy of note that the reverse conditional probability can be solved when it is conditioned on $\mathbf{x}_0$:

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\big(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}\big).$$

Using Bayes' rule, we have:

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = q(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{x}_0)\frac{q(\mathbf{x}_{t-1} \mid \mathbf{x}_0)}{q(\mathbf{x}_t \mid \mathbf{x}_0)}$$

$$\propto \exp\left(-\frac{1}{2}\left(\frac{\left(\mathbf{x}_t - \sqrt{\alpha_t}\mathbf{x}_{t-1}\right)^2}{\beta_t} + \frac{\left(\mathbf{x}_{t-1} - \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0\right)^2}{1-\bar{\alpha}_{t-1}} - \frac{\left(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0\right)^2}{1-\bar{\alpha}_t}\right)\right)$$

$$= \exp\Bigg(-\frac{1}{2}\Bigg(\frac{\mathbf{x}_t^2 - 2\sqrt{\alpha_t}\mathbf{x}_t\mathbf{x}_{t-1} + \alpha_t\mathbf{x}_{t-1}^2}{\beta_t} + \frac{\mathbf{x}_{t-1}^2 - 2\sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0\mathbf{x}_{t-1} + \bar{\alpha}_{t-1}\mathbf{x}_0^2}{1-\bar{\alpha}_{t-1}} - \frac{\big(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}^2}{1-\bar{\alpha}_t}$$

$$= \exp\left(-\frac{1}{2}\left(\left(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right)\mathbf{x}_{t-1}^2 - \left(\frac{2\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0\right)\mathbf{x}_{t-1} + C(\mathbf{x}_t, \mathbf{x}_0)\right)\right)$$

Where $C(\mathbf{x}_t, \mathbf{x}_0)$is some function not involving $\mathbf{x}_{t-1}$ and details are omitted. Following the standard Gaussian density function, the mean and variance can be parameterized as follows (recall that $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{1} \alpha_i$).

$$\tilde{\beta}_t = \frac{1}{\left(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right)} = \frac{1}{\left(\frac{\alpha_t - \bar{\alpha}_t + \beta_t}{\beta_t(1-\bar{\alpha}_{t-1})}\right)} = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t$$

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\left(\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0\right)}{\left(\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}}\right)}$$

$$= \left(\frac{\sqrt{\alpha_t}}{\beta_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}\mathbf{x}_0\right)\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\cdot\beta_t$$

$$= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0$$

Thanks to the nice property, we can represent $\mathbf{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t\right)$ and plug it into the above equation and obtain:

$$\tilde{\boldsymbol{\mu}}_t = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t\right)$$

$$= \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right)$$

As demonstrated in Fig. 3., such a setup is very similar to VAE and thus we can use the variational lower bound to optimize the negative log-likelihood.

$$
\begin{aligned}
-\log p_\theta(\mathbf{x}_0) &\leq -\log p_\theta(\mathbf{x}_0) + D_{\mathrm{KL}}\big(q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{1:T} \mid \mathbf{x}_0)\big) \\
&= -\log p_\theta(\mathbf{x}_0) + \mathbb{E}_{\mathbf{x}_{1:T} \sim q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)}\left[\log \frac{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)}{\frac{p_\theta(\mathbf{x}_{0:T})}{p_\theta(\mathbf{x}_0)}}\right] \\
&= -\log p_\theta(\mathbf{x}_0) + \mathbb{E}_q\left[\log \frac{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})} + \log p_\theta(\mathbf{x}_0)\right] \\
&= \mathbb{E}_q\left[\log \frac{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\right] \\
\text{Let } L_{\mathrm{VLB}} &= \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log \frac{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\right] \geq -\mathbb{E}_{q(\mathbf{x}_0)} \log p_\theta(\mathbf{x}_0)
\end{aligned}
$$

It is also straightforward to get the same result using Jensen's inequality. Say we want to minimize the cross entropy as the learning objective,

$$
\begin{aligned}
L_{\mathrm{CE}} &= -\mathbb{E}_{q(\mathbf{x}_0)} \log p_\theta(\mathbf{x}_0) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log\big(\textstyle\int p_\theta(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T}\big) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log\left(\int q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)} d\mathbf{x}_{1:T}\right) \\
&= -\mathbb{E}_{q(\mathbf{x}_0)} \log\left(\mathbb{E}_{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)} \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)}\right) \\
&\leq -\mathbb{E}_{q(\mathbf{x}_{0:T})} \log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)} \\
&= \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log \frac{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\right] = L_{\mathrm{VLB}}
\end{aligned}
$$

The objective can be further rewritten to be a combination of several KL-divergence and entropy terms in order to convert each term in the equation to be analytically computable. This can be accomplished by using the following strategy.

$$
\begin{aligned}
L_{\mathrm{VLB}} &= \mathbb{E}_{q(\mathbf{x}_{0:T})}\left[\log \frac{q(\mathbf{x}_{1:T} \mid \mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\right] \\
&= \mathbb{E}_q\left[\log \frac{\prod_{t=1}^T q(\mathbf{x}_t \mid \mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_T) \prod_{t=1}^T p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)}\right] \\
&= \mathbb{E}_q\left[-\log p_\theta(\mathbf{x}_T) + \sum_{t=1}^T \log \frac{q(\mathbf{x}_t \mid \mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)}\right] \\
&= \mathbb{E}_q\left[-\log\ p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log\ \frac{q(\mathbf{x}_t \mid \mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)} + \log\ \frac{q(\mathbf{x}_1 \mid \mathbf{x}_0)}{p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1)}\right]
\end{aligned}
$$

$$= \mathbb{E}_q\left[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^{T} \log\left(\frac{q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)} \cdot \frac{q(\mathbf{x}_t \mid \mathbf{x}_0)}{q(\mathbf{x}_{t-1} \mid \mathbf{x}_0)}\right) + \log \frac{q(\mathbf{x}_1 \mid \mathbf{x}_0)}{p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1)}\right]$$

$$= \mathbb{E}_q\left[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^{T} \log\frac{q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)} + \sum_{t=2}^{T} \log \frac{q(\mathbf{x}_t \mid \mathbf{x}_0)}{q(\mathbf{x}_{t-1} \mid \mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1 \mid \mathbf{x}_0)}{p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1)}\right]$$

$$= \mathbb{E}_q\left[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^{T} \log\frac{q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)} + \log \frac{q(\mathbf{x}_T \mid \mathbf{x}_0)}{q(\mathbf{x}_1 \mid \mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1 \mid \mathbf{x}_0)}{p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1)}\right]$$

$$= \mathbb{E}_q\left[\log \frac{q(\mathbf{x}_T \mid \mathbf{x}_0)}{p_\theta(\mathbf{x}_T)} + \sum_{t=2}^{T} \log\frac{q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)} - \log p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1)\right]$$

$$= \mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}\big(q(\mathbf{x}_T \mid \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T)\big)}_{L_T} + \sum_{t=2}^{T} \underbrace{D_{\mathrm{KL}}\big(q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)\big)}_{L_{t-1}} - \underbrace{\log p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1)}_{L_0}\right]$$

Let's label each component in the variational lower bound loss separately:

$$\begin{aligned} L_{\mathrm{VLB}} &= L_T + L_{T-1} + \cdots + L_0 \\ \text{where } L_T &= D_{\mathrm{KL}}\big(q(\mathbf{x}_T \mid \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T)\big) \\ L_t &= D_{\mathrm{KL}}\big(q(\mathbf{x}_t \mid \mathbf{x}_{t+1}, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_t \mid \mathbf{x}_{t+1})\big) \text{ for } 1 \le t \le T-1 \\ L_0 &= -\log p_\theta(\mathbf{x}_0 \mid \mathbf{x}_1) \end{aligned}$$

Because each and every KL term in $\mathbf{L_{VLB}}$ (with the exception of $\mathbf{L_0}$) involves a comparison of two Gaussian distributions, it is possible to compute them using a closed form $\mathbf{L_T}$ is unchanging and can be disregarded during the training process $\mathbf{q}$ due to the fact that it is a Gaussian noise and has no learnable parameters $\boldsymbol{xT}$. Ho et al. (2020) [59] models $\boldsymbol{L_0}$ using a separate discrete decoder derived from $\mathcal{N}\big(\mathbf{x}_0; \boldsymbol{\mu}_\theta(\mathbf{x}_1, 1), \boldsymbol{\Sigma}_\theta(\mathbf{x}_1, 1)\big)$.

Parameterization of $\boldsymbol{Lt}$ for Training Loss:

It is important to keep in mind that we need to train a neural network in order to create an approximation of the conditioned probability distributions on the reverse diffusion process

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

We would like to train $\boldsymbol{\mu\theta}$ to predict $\tilde{\boldsymbol{\mu}}_t = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right)$. Because $\boldsymbol{x_t}$ is available as input at training time, we can reparametrize the Gaussian noise term instead to make it predict $\boldsymbol{\epsilon_t}$ from the input $\boldsymbol{x_t}$ at time step $\boldsymbol{t}$:

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)$$

$$\text{Thus } \mathbf{x}_{t-1} = \mathcal{N}\left(\mathbf{x}_{t-1}; \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right)$$

The loss term $\boldsymbol{L_t}$ is parameterized to minimize the difference from $\underline{\tilde{\boldsymbol{\mu}}}$:

$$\begin{aligned}
L_t &= \mathbb{E}_{\mathbf{x}_0,\epsilon}\left[\frac{1}{2\|\boldsymbol{\Sigma}_\theta(\mathbf{x}_t,t)\|_2^2}\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t,t)\|^2\right] \\
&= \mathbb{E}_{\mathbf{x}_0,\epsilon}\left[\frac{1}{2\|\boldsymbol{\Sigma}_\theta\|_2^2}\left\|\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_t\right) - \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t)\right)\right\|^2\right] \\
&= \mathbb{E}_{\mathbf{x}_0,\epsilon}\left[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\|\boldsymbol{\Sigma}_\theta\|_2^2}\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t)\|^2\right] \\
&= \mathbb{E}_{\mathbf{x}_0,\epsilon}\left[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\|\boldsymbol{\Sigma}_\theta\|_2^2}\left\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t\right)\right\|^2\right]
\end{aligned}$$

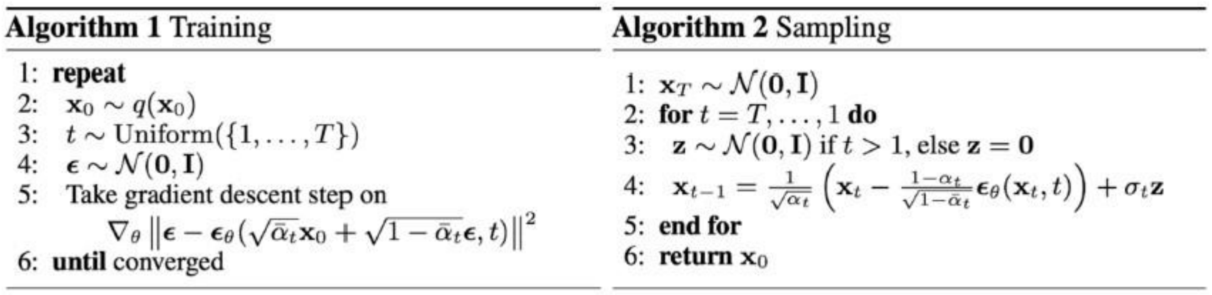
Simplification: From an empirical standpoint, Ho et al (2020) [59] discovered that training the diffusion model with a simplified objective that ignores the weighting term results in better performance:

$$\begin{aligned}
L_t^{\text{simple}} &= \mathbb{E}_{t\sim[1,T],\mathbf{x}_0,\epsilon_t}\left[\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t)\|^2\right] \\
&= \mathbb{E}_{t\sim[1,T],\mathbf{x}_0,\epsilon_t}\left[\left\|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t\right)\right\|^2\right]
\end{aligned}$$

The ultimate aim, which is quite straightforward, is

$$L_{\text{simple}} = L_t^{\text{simple}} + C$$

Where C is a constant not depending on θ.



Algorithm 1 Training

```
1: repeat
2:   $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3:   $t \sim \mathrm{Uniform}(\{1,\ldots,T\})$
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

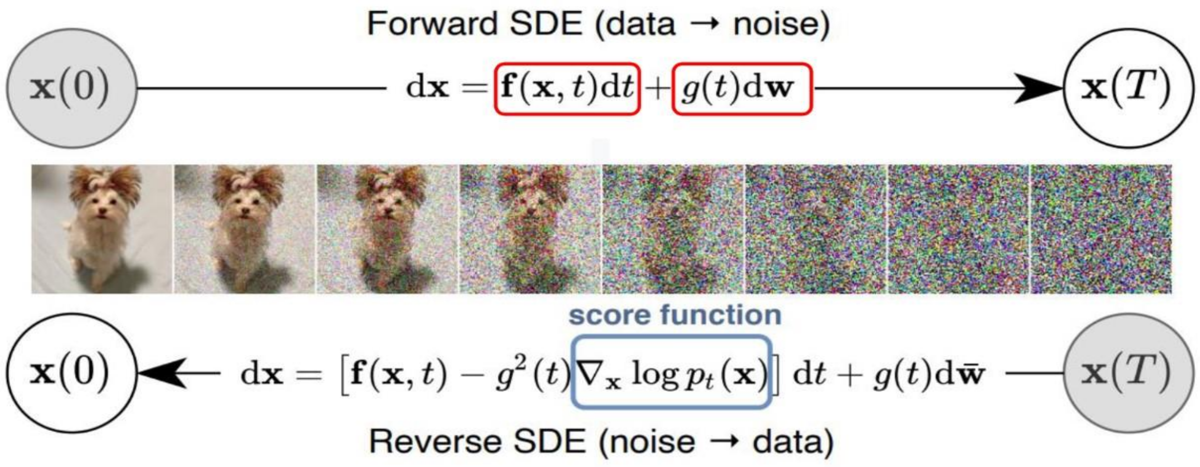
```
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0},\mathbf{I})$
2: for $t = T,\ldots,1$ do
3:   $\mathbf{z} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4:   $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,t)\right)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

Fig. 5. The training and sampling algorithms in DDPM (Image source: Ho et al.2020)

2.2. Score-Based Generative Models (SGMs)

Score-based generative models are a type of generative model that uses the score function, which is a vector field that gives the direction of the steepest ascent of the probability density function, to define a diffusion process that transforms a simple base distribution into the target distribution. Score-based models are known for their ability to generate high-quality samples with high diversity, and for their flexibility in modeling complex data distributions.

Score-based generative models (SGMs) are a class of generative models that learn to generate samples by directly estimating a score function. The score function gives a high score to samples that are likely to be generated from the underlying distribution and a low score to samples that are unlikely to be generated.



Forward SDE (data → noise)
$\mathbf{x}(0)$
$\mathrm{d}\mathbf{x} = \boxed{\mathbf{f}(\mathbf{x},t)\mathrm{d}t} + \boxed{g(t)\mathrm{d}\mathbf{w}}$
$\mathbf{x}(T)$
score function
$\mathbf{x}(0)$
$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x},t) - g^2(t)\boxed{\nabla_\mathbf{x} \log p_t(\mathbf{x})}]\,\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$
$\mathbf{x}(T)$
Reverse SDE (noise → data)

Fig. 6. Score-Based Generative Models (SGMs)

SGMs can be trained using only samples from the underlying distribution, and they do not require access to the density function or the partition function of the distribution. This makes them particularly useful for modeling complex high-dimensional distributions, where computing the partition function may be intractable or impossible.

One popular method for training SGMs is the maximum likelihood estimation (MLE) approach. The model is trained to maximize the log-likelihood of the data, which is defined in terms of the score function. The score function is typically parameterized by a neural network, and the parameters of the network are learned through backpropagation.

One advantage of SGMs is that they can generate high-quality samples even when the training data is limited or noisy. However, they may suffer from mode collapse, where the model learns to generate a small set of similar samples, ignoring other possible modes of the distribution. To address this issue, various techniques such as entropy regularization, diversity promotion, and adversarial training have been proposed.

Overall, SGMs are a promising approach for generative modeling, and they have shown impressive results in various applications such as image generation, text generation, and molecular design.

There are several types of score-based generative models, including:

A. Generative Score Networks (GSNs): These models use a neural network to directly learn the score function of a probability density function. The score function is used to define a diffusion process that transforms a noise distribution into the target distribution.
B. Noise-Conditional Score Networks (NCSNs): These models are similar to GSNs, but they are conditioned on a noise vector. The noise vector is used to control the diversity of the generated samples.
C. Hamiltonian Generative Networks (HGNs): These models use a Hamiltonian dynamics simulation to generate samples. The score function is used to define the dynamics of the system, and the Hamiltonian dynamics simulation is used to generate samples from the target distribution.
D. Energy-Based Generative Models (EBMs): These models use an energy function to model the score function of a probability density function. The energy function is used to define a Markov chain that samples from the target distribution.

Score-based generative models have shown promising results on a variety of datasets, including images, audio, and text. They are an active area of research in the field of generative modeling, and new variants and improvements are continuously being developed.

2.3 Noise-conditioned score network

Noise-conditioned score networks (NCSNs) are a type of generative model that uses a neural network to model the score function of a diffusion process. The score function is a vector field that gives the direction of the steepest ascent of the probability density function, and it can be used to define a diffusion process that transforms a noise distribution into the target distribution.

The score function of a probability density function, denoted by $p(x)$, is represented in NCSNs by the gradient of the log density with respect to the input, denoted as $\nabla_x \log p(x)$. A score-matching neural network denoted by the symbol s is trained in order to learn and estimate the score function. The objective of this neural network is to ensure that $S_\theta(x) \approx \nabla_x \log p(x)$.As a result, the objective function of the scoring network can be defined as the following:

$$\mathbb{E}_{x\sim p(x)} \| s_\theta(x) - \nabla_x \log p(x) \|_2^2$$

Even though the problem has been clearly defined, numerically optimizing above equation is nearly impossible because there is no way to determine the value of $\nabla_x \log p(x)$. This is despite the fact that the problem has been clearly defined. There are, however, some tried-and-true methods for learning score functions from data, such as score matching [67], denoising score matching[140], and sliced score matching. Score matching is one of these methods.

In addition, we emphasized that the primary challenge of training lay in the fact that the trained score functions were unreliable in a low-dimension manifold. This is due to the fact that data are typically located in a low-dimension manifold that is embedded in a high-dimension space (the manifold hypothesis). They also demonstrated that these problems could be resolved by including Gaussian noise in the data at various scales. This resulted in an improvement in the suitability of the data distribution for score-based generative modeling.

As a result, they suggested using a single noise-conditioned score network, or NCSN, to estimate the score that was associated with each noise level. Let there be a sequence of Gaussian noise levels that goes from$0 < \sigma_1 < \sigma_2 < \cdots < \sigma_t < \cdots < \sigma_T$. This sequence should be constructed in such a way that $p_{\sigma_t}(x_t \mid x) = \mathcal{N}(x_t; x, \sigma_t^2\mathbf{I}), p_{\sigma_1}(x) \approx p(x_0)$, and$p_{\sigma_\tau}(x) \approx \mathcal{N}(0, \mathbf{I})$.

The noise-conditioned score network $s_\theta(x, \sigma_t)$ with the denoising score matching is able to approximate the gradient log density function, which results in the equation $s_\theta(x, \sigma_t) \approx \nabla_x \log\left(p_{\sigma_t}(x)\right), \forall t \in \{1, \ldots, T\}$ And for x_t, $\nabla_x \log\left(p_{\sigma_t}(x)\right)$is derived as .

$$\nabla_{x_t} \log p_{\sigma_t}(x_t \mid x) = -\frac{x_t - x}{\sigma_t}$$

As a consequence of this, the objective function for optimization found in above equations can be rewritten as:

$$\frac{1}{T}\sum_{t=1}^{T} \lambda(\sigma_t)\mathbb{E}_{p(x)}\mathbb{E}_{x_t \sim p_t(x_t|x)} \left\| s_\theta(x_t, \sigma_t) + \frac{x_t - x}{\sigma_t} \right\|_2^2$$

in which the $\lambda(\sigma t)$ is a weighting function.

The annealed Langevin dynamics algorithm is utilized by NCSNs during the sampling phase. This algorithm makes use of a Markov Chain Monte Carlo (MCMC) procedure to take a simple sample from a distribution in accordance with its score function, which is denoted by $\nabla_x \log p(x)$. Calculating x_i using the Langevin method in a recursive manner looks like this:

$$x_i = x_{i-1} + \frac{\gamma}{2}\nabla_x \log p(x) + \sqrt{\gamma}\omega_i$$

Where γ determines the amplitude of the update in the direction of the score, x_0 is sampled from the prior distribution, and the noise is drawn according to $\omega_i \sim \mathcal{N}(0, I)$ where I is the iterative norm of the normal distribution.

Advantage of NCSNs over DDPMs is that they can generate more diverse samples by conditioning the diffusion process on a noise vector. This allows the model to learn a richer distribution and generate samples with more variability. NCSNs have shown promising results on a variety of datasets, including images and audio, and are an active area of research in the field of generative modeling.

2.4. Stochastic Differential Equations

Stochastic differential equations (SDEs) are a type of differential equation that incorporates random noise into the dynamics of the system. They are widely used in many areas of science and engineering to model systems that are subject to random fluctuations or noise.

DDPMs and NCSNs can be further generalized to the situation in which the perturbation and denoising processes can be described as stochastic differential equations. This situation arises when either the time steps or the noise levels are unlimited (SDEs). The formulation score SDE refers to this generalized approach [135], which gradually transforms the data into noise. A stochastic differential equation is used in the forward process of the formulation score SDE. Additionally, an estimated score function of the noisy data distribution is required for this process. It is the same as the solution found in the Itô SDE, which includes a drift component for mean transformation and a diffusion coefficient for describing noise:

$$dx = f(x,t)dt + g(t)dw, t \in [0,T]$$

where w stands for the standard Wiener process also known as Brownian motion, $f(x,t)$ and $g(t)$ are the drift and diffusion coefficients of SDE, and x represents the time variable. A particular instance of the discretizational SDE is represented by the forward process in DDPMs and SGMs.

Below equation in [162], also known as the reverse-time SDE, provides the formulation of the SDE's reverse diffusion process, which is as follows:

$$dx = [f(x,t) - g^2(t)\nabla_x \log p_t(x)]dt + g(t)d\overline{w}$$

where $\overline{w}$ is the standard Brownian motion running backward in time and dt is the infinitesimally small negative time step represents the standard Brownian motion. The marginal densities of the reverse SDE and the forward SDE are identical; the only difference is that they are expressed in the opposite time direction. A trainable neural network $s_\theta(x,t)$ is used, in the same way that DDPMs and NCSNs are, in order to numerically solve the reverse-time SDE. This is done in order to estimate the actual score function $\nabla_x \log p_t(x)$. One way to define the objective function is as follows:

$$\mathbb{E}_{x(t)\sim p}\big(\, x(t) \mid x(0)\,\big) x(0) \sim p_{\text{data}} \left[\frac{\lambda(t)}{2} \left\| s_\theta(x(t),t) - \nabla_{x(t)} \log p_t\big(\, x(t) \mid x(0)\,\big) \right\|_2^2\right]$$

where $t \sim \mathcal{U}([0,T])$ denotes the uniform distribution over $[0,T]$ and λ is a weighting function. This distribution is denoted by the notation. In addition, Song et al. suggested a number of methods for collecting samples, such as the Predictor-Corrector sampler, in order to produce high-quality data. After using a numerical approach to sample data from the reverse-time SDE, this procedure uses a score-based method (i.e., annealed Langevin dynamics) as a corrector. This method is called annealed Langevin dynamics [132].

SDEs can be used to model a wide range of phenomena, such as the diffusion of particles, the evolution of populations, the spread of epidemics, and the dynamics of financial markets. SDEs can also be used to model the behavior of neural networks and other machine learning models.

SDEs are a powerful tool for modeling systems that are subject to random fluctuations or noise, and they have many applications in science and engineering. They are an active area of research in the field of stochastic processes and have the potential to revolutionize the way we model and understand complex systems.

Chapter 3

DIFFUSION MODELS WITH EFFICIENT SAMPLING

In most cases, it is necessary to use iterative methods that consist of a great number of assessment steps in order to generate samples from diffusion models. Recent research has spent a significant amount of effort on developing methods to simultaneously accelerate the sampling process and improve the overall sample quality. We divide these effective sampling strategies into two primary categories: those that do not involve learning (learning-free sampling), and those that require an additional learning process after the diffusion model has been trained. We call the former "learning-free sampling," and the latter "learning-required sampling" (learning-based sampling).

3.1 Learning-Free Sampling

Many samplers used in diffusion models are dependent on discretizing either the reverse-time SDE found in above equation in chapter 2.4, or the probability flow ODE derived from equation in chapter 2.4. Because the cost of sampling rises in direct proportion to the number of time steps that are discretized, a significant number of researchers have been concentrating their efforts on developing discretization schemes that reduce the total number of time steps while simultaneously reducing the number of discretization errors.

3.1.1 Solvers for SDEs. One such discretization of the reverse-time stochastic differential equation can be seen as the generation process of DDPM [59, 128]. As was covered in chapter 2.4, the stochastic differential equation (SDE) represented by Equation is discretized by the forward process of DDPM, whose corresponding reverse SDE is represented as

$$\mathrm{dx} = -\frac{1}{2}\beta(t)\big(\mathrm{x}_t - \nabla_{\mathbf{x}_t} \log q_t(\mathrm{x}_t)\big)\mathrm{d}t + \sqrt{\beta(t)}\mathrm{dw}$$

The authors Song et al. (2020) [135] demonstrate that the numerical SDE solver for Equation is equivalent to the reverse Markov chain that is specified by the equation in above chapter 2.1.

Both Critically-Damped Langevin Diffusion (CLD) [35] and Noise-Conditional Score Networks (NCSNs) [132] are able to solve the reverse-time SDE by drawing inspiration from Langevin dynamics. In particular, NCSNs make use of annealed Langevin dynamics (ALD, see Chapter 2.1) to iteratively produce data while gradually reducing noise level. This continues until the distribution of the generated data converges to the distribution of the original data. Despite the fact that the sampling trajectories of ALD are not precise solutions to the reverse-time SDE, they have the correct marginals and, as a result, provide the correct samples. This is based on the premise that Langevin dynamics converges to its equilibrium at any noise level. Consistent Annealed Sampling (CAS) [72], a score-based MCMC strategy that features improved scaling of time steps and additional noise, is a further improvement that may be made to the ALD methodology. CLD presents an augmented SDE with an auxiliary velocity term that resembles underdamped Langevin diffusion. This idea comes from statistical mechanics, which inspired CLD's work. It is possible that learning scores of data directly is more difficult than learning the score function of the conditional distribution of velocity given data. However, in order for CLD to obtain the time reversal of the extended SDE, it is only necessary to learn the score function of the conditional distribution of velocity given data. It has been found that the additional velocity term improves both the sampling speed and the quality.

The reverse diffusion method that is proposed in [135] discretizes the reverse-time SDE in the same way as the forward diffusion method does.

For any discretization of the forward SDE that only requires one step, one can express the general form as follows:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \mathbf{f}_i(\mathbf{x}_i) + \mathbf{g}_i\mathbf{z}_i, \boldsymbol{i} = \mathbf{0}, \mathbf{1}, \cdots, \boldsymbol{N} - \mathbf{1}$$

Where $\mathrm{Z}_i \sim \mathcal{N}(0, \mathrm{I})$, f_i and g_i are constants that are dependent on the drift/diffusion coefficients of the SDE as well as the discretization scheme.

The idea behind reverse diffusion is to discretize the reverse-time SDE in the same way that the forward SDE is discretized, that is,

$$\mathbf{x}_i = \mathbf{x}_{i+1} - \mathbf{f}_{i+1}(\mathbf{x}_{i+1}) + \mathbf{g}_{i+1}\mathbf{g}_{i+1}^{\boldsymbol{t}}\mathbf{s}_{\theta^*}(\mathbf{x}_{i+1}, \boldsymbol{t}_{i+1}) + \mathbf{g}_{i+1}\mathbf{z}_i\boldsymbol{i} = \mathbf{0}, \mathbf{1}, \cdots, \boldsymbol{N} - \mathbf{1}$$

where $\boldsymbol{s}_{\theta^*}(\mathbf{x}_i, t_i)$ represents the noise-conditional score model that has been trained. The authors Song et al. (2020) [135] provide evidence that the reverse diffusion method is a numerical SDE solver that can be applied to the equation containing the reverse-time SDE. Empirical data indicate that the performance of this sampler is somewhat better than that of DDPM [135] for a certain type of SDEs that is referred to as the VP-SDE. This technique can be used to any sorts of forward SDEs.

Jolicoeur-Martineau et al. (2021) [71] create an SDE solver that has adaptable step sizes for the purpose of producing results more quickly. A comparison is made between the output of a high-order SDE solver and the output of a low-order SDE solver in order to determine the appropriate step size. Each time there is a new time step, the high-order and low-order solvers generate a new sample $\boldsymbol{x}'_{\mathbf{high}}$ and $\boldsymbol{x}'_{low}$ accordingly, based on the sample $\boldsymbol{x}'_{prev}$ that came before. After that, the step size is modified by analysing the difference in the results of the two samples. In the event that $\boldsymbol{x}'_{\mathbf{high}}$ and $\boldsymbol{x}'_{low}$ produce comparable results, the algorithm will return $\boldsymbol{x}'_{\mathbf{high}}$ and then proceed to raise the step size. The resemblance between $\boldsymbol{x}'_{\mathbf{high}}$ and $\boldsymbol{x}'_{low}$ is as follows:

$$E_q = \left\| \frac{x'_{low} - x'_{high}}{\delta(x', x'_{prev})} \right\|^2$$

where $\boldsymbol{\delta}(\boldsymbol{x}'_{low}, \boldsymbol{x}'_{prev}) := max\left(\boldsymbol{\epsilon}_{abs}, \boldsymbol{\epsilon}_{\mathbf{rel}}\, max(|\boldsymbol{x}'_{low}, |\boldsymbol{x}'_{prev}|)\right)$,where $\boldsymbol{\epsilon}_{abs}$ and $\boldsymbol{\epsilon}_{\mathbf{rel}}$ are absolute and relative tolerances respectively, and where $max(|\boldsymbol{x}'_{low}, |\boldsymbol{x}'_{prev}|)$ is the maximum value.

The reverse SDE can be solved using the predictor-corrector method that was developed in [135]. This method combines numerical SDE solvers (referred to as the "predictor") and iterative Markov chain Monte Carlo (MCMC) procedures (referred to as the "corrector"). At each time step, the predictor-corrector technique first uses a numerical SDE solver to produce a coarse sample. Next, a "corrector" is used to correct the sample's marginal distribution using score-based MCMC.

Finally, the method concludes with a "predictor" that generates a refined sample. The final samples have the same time-marginals as the solution trajectories of the reverse-time SDE, which means that their distributions are comparable at each and every time step. The use of an additional predictor that does not include correctors is shown to be less effective than adding a corrector that is based on the Langevin Monte Carlo algorithm, as shown by empirical results [135]. Karras et al. (2022) [74] further improved the Langevin dynamics corrector in [135] by suggesting a Langevin-like "churn" step of adding and eliminating noise. As a result, they were able to achieve a new state-of-the-art sample quality on datasets such as CIFAR-10 and ImageNet-64.

3.1.2 ODE solvers. Solving the probability flow ODE, which was presented in Part 2.3, is the foundation of a significant portion of the research on faster diffusion samplers. ODE solvers, in contrast to SDE solvers, have deterministic trajectories, which means that they are unaffected by the fluctuations caused by stochastic processes. In general, these deterministic ODE solvers converge substantially more quickly than their stochastic analogues, although at the expense of slightly lower sample quality.

One of the earliest works on accelerating diffusion model sampling is called Denoising Diffusion Implicit Models (DDIM), which was published in [129]. The first goal was to extend the original DDPM to non-Markovian cases using the Markov chain that is presented in the following sentence.

$$q(\mathbf{x}_1, \ldots, \mathbf{x}_T \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{x}_0)$$
$$q_\sigma(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1} \mid \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0), \sigma_t^2 \mathrm{I})$$
$$\tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) := \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} \cdot \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0}{\sqrt{1-\bar{\alpha}_t}}$$

This formulation contains DDPM and DDIM as special cases. More specifically, DDPM corresponds to the setting $\sigma_t^2 = \frac{\tilde{\beta}_{t-1}}{\tilde{\beta}_t}\beta_t$, whereas DDIM corresponds to the setting $\sigma_t^2 = 0$. To reverse this non-Markov perturbation process, which is totally predictable when $\sigma_t^2 = 0$, DDIM learns a Markov chain to use as a learning tool. According to the findings presented in [88] and [129], the DDIM sampling process may be equated to a unique discretization scheme of the probability flow ODE. These findings were derived from the fact that these papers. generalized Denoising Diffusion Implicit Models (gDDIM) [159] proposes a modified parameterization of the score network that enables deterministic sampling for more general diffusion processes, such as the one in Critically-Damped Langevin Diffusion (CLD) [35]. This modification was inspired by an analysis of DDIM on a singleton dataset. A pseudo numerical method is suggested in PNDM [85] for generating samples along a certain manifold in $\mathcal{R}^N$. After solving differential equations on manifolds with a numerical solver that has a nonlinear transfer portion, it generates a sample that incorporates DDIM as a special case. This process takes place in two steps.

Karras et al. (2022) [74] demonstrate that Heun's 2nd order approach [3] offers a fantastic compromise between the sample quality and the sampling speed that is required. These findings are based on extensive experimental research. The higher-order solver results in less discretization error, but at the expense of one more evaluation of the learnt score function for each time step. While Euler's approach produces samples of equivalent quality, Heun's method generates samples of comparable or even superior quality with fewer sampling steps.

Both the Diffusion Exponential Integrator Sampler [158] and the DPM-solver [88] take advantage of the semi-linear nature of the probability flow ODE in order to construct specialised ODE solvers that are more effective than general-purpose Runge-Kutta methods.

To be more specific, the linear portion of the probability flow ODE can be estimated analytically, and the non-linear portion of the ODE can be solved using methods that are analogous to exponential integrators in the field of ODE solvers. These techniques utilize DDIM as a first-order approximation in their calculations. However, because of this, higher order integrators are now possible. These integrators can produce high-quality samples in as few as 10 to 20 iterations, which is a significant reduction from the hundreds of iterations that are typically required by diffusion models that do not use accelerated sampling.

3.2 Learning-Based Sampling

Another effective method for constructing diffusion models is one that relies on learning-based sampling. This method offers higher sampling speeds by utilizing partial steps or by training a sampler for the reverse process. Nevertheless, the modest decrease in sample quality is a price that must be paid for these faster sampling speeds. Learning-based sampling, as opposed to learning-free approaches, often entails selecting steps by optimizing particular learning objectives, as opposed to learning-free approaches, which use handcrafted steps.

3.2.1 Optimized Discretization: Discretization that has been optimized Given a diffusion model that has already been trained, Watson et al. (2021) [143] proposed a method for determining the best discretization scheme by choosing the best K time steps to maximize the training objective for DDPMs. This method requires the use of a pre-trained diffusion model. The realization that the DDPM objective may be partitioned into a total of its constituent parts—which makes it an excellent candidate for dynamic programming—is essential to the implementation of this strategy. On the other hand, it is common knowledge that the variational lower bound that is employed for DDPM training does not correspond directly with the quality of the samples. This problem was solved in a subsequent piece of research referred to as Differentiable Diffusion Sampler Search [142], which focused on directly improving a standard measure for measuring sample quality known as the Kernel Inception Distance (KID). With the assistance of reparameterization [117] and gradient rematerialization, it is possible to carry out this optimization. Dockhorn et al. (2022) [36] design a second-order solver for accelerating synthesis using truncated Taylor techniques. This is accomplished by training an additional head on top of the first-order scoring network.

3.2.2 Truncated Diffusion: Truncated diffusion is the third topic. By cutting short the forward and reverse diffusion processes, one can make the sample process go much more quickly [96]. The most important step is to terminate the forward diffusion process as soon as possible, preferably within the first few steps, and then to start the reverse denoising process with a distribution that is not Gaussian. Diffusion of samples from pre-trained generative models, such as variational autoencoders [117] or generative adversarial networks [49], is an effective method for obtaining samples from this distribution. This method allows for the efficient collection of data.

3.2.3 Knowledge Distillation: The Process of Distilling Knowledge Methods that rely on the distillation of knowledge [90] have the potential to dramatically boost the sampling speed of diffusion models. To be more specific, in Progressive Distillation, the authors propose condensing the entire process of sampling into a speedier sampler that requires only half as many steps as the current method. The authors are able to train the new sampler to match the input and output produced by the DDIM sampling method because they have parameterized the new sampler as a deep neural network.

The sampling stages can be lowered even further by repeating this technique; however, the quality of the samples may suffer if there are fewer steps total. The authors provide new weighting schemes for the objective function as well as new parameterizations for the diffusion models so that they can solve this issue.

Chapter 4

DIFFUSION MODELS WITH IMPROVED LIKELIHOOD

According to what was covered in Chapter 2.1, the goal of training diffusion models is to achieve a variational lower bound (VLB) on the log-likelihood that is negative. This bound, on the other hand, might not be very tight in many situations [77], which could result in diffusion models producing log-likelihoods that are less than ideal. In this part, we will review recent research on maximizing the likelihood of events occurring in diffusion models. Our primary concentration is placed on the following three categories of methods: noise schedule optimization, reverse variance learning, and exact loglikelihood evaluation.

4.1 Noise Schedule Optimization

In the traditional approach of modelling diffusion, the noise schedules for the forward process are hand-created rather than using trainable parameters. One can further maximize the VLB in order to attain greater log-likelihood values by optimizing the forward noise schedule in conjunction with other parameters of diffusion models [77, 99]. This allows one to further maximize the VLB.

The research conducted by iDDPM [99] shows that a particular cosine noise schedule has the potential to boost log-likelihoods. To be more specific, the cosine noise schedule that they developed takes the form:

$$\bar{\alpha}_t = \frac{h(t)}{h(0)}, h(t) = \cos\left(\frac{\frac{t}{T}+m}{1+m}\cdot\frac{\pi}{2}\right)^2$$

where $\bar{\alpha}_t$ and β_t are described in earlier equations, respectively, and m is a hyperparameter that controls the noise scale when t = 0 is considered. In addition to this, they suggest a parameterization for the inverse variance that involves an interpolation in the log domain between the values β_t and $1-\bar{\alpha}_t$.

The authors of Variational Diffusion Models (VDMs) [77] propose that in order to increase the likelihood of continuous-time diffusion models, it is possible to improve the likelihood of these models by jointly training the noise schedule and other diffusion model parameters to maximize the VLB. [77]

They construct the forward perturbation process in accordance with and use a monotonic neural network to parameterize the noise schedule.

$$\sigma_t^2 = \text{sigmoid}\left(\gamma_\eta(t)\right), q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\bar{\alpha}_t \mathbf{x}_0, \sigma_t^2 \mathrm{I}), \text{ and } \bar{\alpha}_t = \sqrt{(1-\sigma_t^2)}$$

In addition, the authors demonstrate that the VLB for data point x may be reduced to a form that is solely reliant on the signal-to-noise ratio $\mathrm{R}(t) := \frac{\bar{\alpha}_t^2}{\sigma_t^2}$. This form can be found by using the aforementioned formula. In specifically, the L_{VLB} is capable of being broken down into

$$L_{VLB} = -\mathbb{E}_{\mathbf{x}_0} \mathrm{KL}\big(q(\mathbf{x}_T \mid \mathbf{x}_0) \parallel p(\mathbf{x}_T)\big) + \mathbb{E}_{\mathbf{x}_0,\mathbf{x}_1} \log p(\mathbf{x}_0 \mid \mathbf{x}_1) - L_D,$$

where the first and second terms can be optimized directly in a manner analogous to how variational autoencoders are trained. The following constitutes an additional simplification of the third term:

$$L_D = \frac{1}{2}\mathbb{E}_{\mathrm{x}_0,\epsilon\sim\mathcal{N}(0,\mathrm{I})}\int_{\mathrm{R}_{min}}^{\mathrm{R}_{max}} \|\mathrm{x}_0 - \tilde{\mathbf{x}}_\theta(\mathbf{x}_v, v)\|_2^2 dv,$$

where $\mathrm{R}_{max} = R(1)$ and $\mathrm{R}_{min} = R(T), \mathbf{x}_v = \bar{\alpha}_v\mathbf{x}_0 + \sigma_v\epsilon$ signifies a noisy data point that was acquired by diffusing x0 with the forward perturbation process until $t = R^{-1}(v)$, and $\tilde{\mathbf{x}}_\theta$ denotes the noise-free data point that was predicted by the diffusion model. As a consequence of this, noise schedules do not have an effect on the VLB so long as they have the same values at R_{min} and R_{max}. Instead, noise schedules will only have an effect on the variance of the Monte Carlo estimators for the VLB.

4.2 Reverse Variance Learning

In the traditional formulation of diffusion models, it is assumed that the variance parameters of Gaussian transition kernels in the reverse Markov chain are fixed. You may recall that we formed the reverse kernel as $q_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)\right)$ in equation, but that we frequently fixed the reverse variance $\Sigma_\theta(\mathbf{x}_t, t)$ to $\beta_t\mathbf{I}$. A great number of techniques advocate training the reverse variances in addition to the forward ones in order to further maximise VLB and log-likelihood values.

Nichol and Dhariwal present a proposal in iDDPM [99] to learn the reverse variances by first parameterizing them with a type of linear interpolation and then training them with a hybrid objective. This would allow the reverse variances to be learned. This leads to larger log-likelihoods as well as faster sampling without a reduction in the quality of the samples collected. To be more specific, they parametrize the reverse variance in equation as follows:

$$\Sigma_\theta(\mathrm{x}_t, t) = \exp\left(\theta \cdot \log\beta_t + (1-\theta)\cdot\log\tilde{\beta}_t\right),$$

where $\tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}$. β_t and θ are both trained together to optimize VLB. This straightforward parameterization sidesteps the instability introduced by estimating more involved versions of $\Sigma_\theta(\mathbf{x}_t, t)$ and is said to result in improved probability values.

Analytic-DPM [6] reveals an astounding finding, namely that the optimal reverse variance may be derived from a pre-trained score function using the analytic form that can be found below:

$$\Sigma_\theta(\mathrm{x}_t, t) = \sigma_t^2 + \left(\sqrt{\frac{\bar{\beta}_t}{\alpha_t}} - \sqrt{\bar{\beta}_{t-1} - \sigma_t^2}\right)^2 \cdot \left(1 - \bar{\beta}_t\mathbb{E}_{q_t(\mathrm{x}_t)}\frac{\|\nabla_{\mathrm{x}_t}\log q_t(\mathrm{x}_t)\|^2}{d}\right)$$

As a consequence of this, if we are provided with a pre-trained score model, we are able to estimate its first- and second-order moments in order to acquire the optimal reverse variances. Using them in the VLB analysis can result in more accurate VLBs and higher likelihood values.

4.3 Exact Likelihood Computation

In the Score SDE [135] formulation, samples are created by solving the following reverse SDE, where $\nabla_{\mathbf{x}_t} \log p_\theta(\mathbf{x}_t, t)$ is substituted by the learnt noise-conditional score models $\mathbf{S}_\theta(\mathbf{x}_t, t)$

$$\mathrm{dx} = f(\mathrm{x}_t, t) - g(t)^2 \mathrm{s}_\theta(\mathrm{x}_t, t)\mathrm{d}t + g(t)\mathrm{dw}.$$

For the sake of this discussion, we will refer to the distribution of samples produced by solving the SDE shown above as p_θ^{sde}. Data can also be generated by entering the score model into the probability flow ODE found in above equations, which produces the following results:

$$\frac{\mathrm{dx}_t}{\mathrm{d}t} = \underbrace{f(\mathrm{x}_t, t) - \frac{1}{2} g^2(t) \mathrm{s}_\theta(\mathrm{x}_t, t)}_{:= \tilde{f}_\theta(\mathrm{x}_t, t)}$$

In a similar vein, we will use p_θ^{ode} to indicate the distribution of samples that were produced by solving this ODE, we will use the notation ode. According to the theory of neural ODEs[20] and continuous normalizing flows[17], p_θ^{ode} can be accurately computed, despite the fact that it requires a lot of processing power. Several concurrent works [65, 87, 131] demonstrate that there is an efficiently computable variational lower bound for p_θ^{sde}. Furthermore, we can directly train our diffusion models to maximize p_θ^{sde} utilizing modified diffusion losses. This is possible because of the existence of an efficiently computable variational lower bound.

In particular, Song et al. (2021) [131] demonstrate that using a particular weighting function known as likelihood weighting, the aim that is used for training score SDEs implicitly optimizes the predicted value of p_θ^{sde} on data. This was demonstrated by using a specific weighting function. The evidence suggests that

$$\mathrm{D}_{KL}\left(q_0 \parallel p_\theta^{\mathrm{sde}}\right) \leq \mathcal{L}(\theta; g(\cdot)^2) + \mathrm{D}_{KL}(q_T \parallel \pi),$$

where $\mathcal{L}(\theta; g(\cdot)^2)$ is the Score SDE objective, and $\lambda(t) = g(t)^2$ is the variable being evaluated. Since $\mathrm{D}_{KL}\left(q_0 \parallel p_\theta^{\mathrm{sde}}\right) = -\mathbb{E}_{q_0} \log\left(p_\theta^{\mathrm{sde}}\right) +$ const and $\mathrm{D}_{KL}(qT \parallel \pi)$ is a constant, training with $\mathcal{L}(\theta; g(\cdot)^2)$ amounts to minimizing $-\mathbb{E}_{q_0} \log\left(p_\theta^{\mathrm{sde}}\right)$, the expected negative log-likelihood on data. This can be thought of as minimizing the expected negative log-likely In addition, the following bound for $p_\theta^{\mathrm{sde}}(\mathrm{x})$ is provided by Song et al. (2021) and Huang et al. (2021) [65

, 131]:

$$-\log p_\theta^{\mathrm{sde}}(\mathbf{x}) \leq \mathcal{L}'(\mathbf{x})$$

where the expression $\mathcal{L}'(\mathbf{x})$ is defined as

$$\mathcal{L}'(\mathbf{x}) := \int_0^T \mathbb{E}\left[\frac{1}{2}\|g(t)\mathbf{s}_\theta(\mathbf{x}_t, t)\|^2 + \nabla \cdot (g(t)^2 \mathbf{s}_\theta(\mathbf{x}_t, t) - f(\mathrm{x}_\mathbf{t}), \mathrm{t}) \mid \mathbf{x}_0 = \mathrm{x}\right] dt$$
$$- \mathbb{E}_{\mathbf{x}_T}\left[\log p_\theta^{\mathrm{sde}}(\mathbf{x}_T) \mid \mathbf{x}_0 = x\right]$$

The initial half of above equation is similar to implicit score matching [67], and the entire bound can be effectively determined through the use of Monte Carlo methods.

As the probability flow ODE is a specific instance of neural ODEs or continuous normalizing flows, we may use well-established methods from those domains to reliably compute $\log p_\theta^{\mathrm{ode}}$. This is because the probability flow ODE is a special case of neural ODEs or continuous normalizing flows. To be more specific, we have

$$\log p_\theta^{\mathrm{ode}}(\mathrm{x}_0) = \log p_T(\mathrm{x}_T) + \int_{t=0}^{T} \nabla \cdot \tilde{f}_\theta(\mathrm{x}_t, t)\mathrm{d}t$$

With the use of numerical ODE solvers and the Skilling-Hutchinson trace estimator [127], it is possible to construct the one-dimensional integral that was discussed earlier. Unfortunately, this formula cannot be directly optimized to maximize p_θ^{ode} on data since it requires invoking expensive ODE solvers for each data point x0. As a result, direct optimization of this formula to maximize p_θ^{ode} on data is not possible. Song et al. (2021) [131] propose to maximize the variational lower bound of p_θ^{sde} as a proxy for maximizing p_θ^{ode}, which gives rise to a family of diffusion models called Score Flows. This is done to reduce the expense of directly maximizing p_θ^{ode} with the aforementioned formula.

Lu et al. (2022) [87] propose a further improvement to Score Flows by minimizing not just the vanilla score matching loss function, but also its higher order generalizations. This is done in order to get a lower overall loss. They demonstrate that a bound may be placed on log p_θ^{ode} using the first-, second-, and third-order score matching mistakes. Using this theoretical notion as a jumping off point, the authors go on to offer efficient training techniques for minimizing high order score matching losses and report improved p_θ^{ode} on data.

Chapter 5

DIFFUSION MODELS FOR DATA WITH SPECIAL STRUCTURES

Even though diffusion models have been quite successful in the past for data domains such as images and audio, it is not always the case that they can be effortlessly transferred to other modalities. In order for diffusion models to perform their functions accurately, it is necessary to take into account the unique structures of a great number of significant data domains. When models rely on score functions that can only be constructed on continuous data domains, for instance, or when data reside on low-dimensional manifolds, this might give rise to a number of challenges and complications. The models of dispersal need to be modified in a number of different ways in order to accommodate these issues.

5.1 Discrete Data

Because the Gaussian noise perturbation that is used in DDPMs does not have a natural fit for discrete data, and because the score functions that are required by SGMs and Score SDEs can only be defined on continuous data domains, the vast majority of diffusion models are designed for continuous data domains. Many research [6, 53] expand on Sohl-Dickstein et al. (2015) [128] to generate discrete data of high dimensions in order to get around this challenge. To be more specific, the Gaussian noise is replaced with a random walk on the discrete data space, also known as a random masking operation, when using the VQ-Diffusion [53] algorithm. The transition kernel for the forward process that was produced as a result takes the form of

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathbf{v}^\top(\mathbf{x}_t) Q_t \mathbf{v}(\mathbf{x}_{t-1})$$

where $\mathbf{Q}_t$ represents the transition kernel of a lazy random walk and **v(x)** is a one-hot column vector. This is an example of a lazy random walk. Discrete Denoising Diffusion Probabilistic Models (D3PM) is able to account for discrete data in diffusion models by constructing the forward noise process with discretized absorbing state kernels.

Gaussian kernels. The first continuous-time framework for discrete diffusion models is presented in Campbell's et al (2022) [15] research. They were able to derive efficient samplers that outperformed their discrete counterparts by utilizing Continuous Time Markov Chains. Additionally, they were able to provide a theoretical analysis on the error that existed between the sample distribution and the true data distribution.

5.2 Data with Invariant Structures

The structures of the data that are used in many significant domains remain constant. Examples of things that are invariant under permutation include graphs and point clouds, which can also withstand translation and rotation. These invariances are frequently disregarded in diffusion models, which can result in performance that is less than optimal. In an effort to solve this problem, a number of works [31, 104] have proposed equipping diffusion models with the capacity to take into account invariances in data.

Niu et al. (2020) [104] begin by addressing the issue of generation of permutation-invariant graphs using diffusion models. They are able to do this by parameterizing the noise-conditional score model with the help of an EDP-GNN, which is an abbreviation for a permutation equivariant graph neural network. This concept is developed further in GDSS [70], which suggests a graph diffusion process that operates in continuous time. The joint distribution of nodes and edges is modelled by this process using a system of stochastic differential equations (SDEs). Message-passing operations are used to guarantee that the model is permutation-invariant.

In a similar manner, Shi et al. (2021) [124] and Xu et al. (2022) [172] make it possible for diffusion models to generate molecular conformations that are invariant to translation as well as rotation. For instance, Xu et al. (2022) [172] demonstrates that Markov chains beginning with an invariant prior and evolving with equivariant Markov kernels can induce an invariant marginal distribution. This distribution can then be used to enforce appropriate data invariance when it comes to the generation of molecular conformations. In technical parlance, we will refer to the rotation or translation operation as T. Given the above, $p(\mathbf{x}_T) = p(\mathcal{T}(\mathbf{x}_T)), p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = p_\theta(\mathcal{T}(\mathbf{x}_{t-1}) \mid \mathcal{T}(\mathbf{x}_t))$ Xu et al. (2022) [172] Demonstrate that the sample distribution is guaranteed to be invariant to $\mathcal{T}$; more specifically, show that $p_0(\mathbf{x}) = p_0(\mathcal{T}(\mathbf{x})) p_0(\mathbf{x}) = p_0(\mathcal{T}(\mathbf{x}))$.As a consequence of this, it is possible to construct a diffusion model that results in rotation- and translation-invariant molecular conformations, provided that both the prior and transition kernels enjoy the same degree of invariance.

5.3 Data with Manifold Structures

In machine learning, it is common to work with data that has many different structures. Natural data frequently exist on manifolds that have a lower intrinsic dimensionality, as the manifold hypothesis [41] postulates. In addition, many data domains have manifold structures that are well known to the public. Because our planet is shaped like a sphere, certain data, such as those pertaining to climate and earth, naturally lie on the sphere. Developing diffusion models for data on manifolds has been the focus of a great deal of research. We classify them according to whether the manifolds are known or learned, and then we present some representative works in the following chapter.

5.3.1 Known Manifolds: Manifolds That Are Already Known Recent research has extended the Score SDE formulation to a variety of different manifolds that are already known. The generalization of neural ODEs[20] and continuous normalizing flows[17] to Riemannian manifolds [86, 97] is analogous to this adaptation. Score matching and score functions have also been adapted to Riemannian manifolds by the researchers so that these models can be trained using them.

The Riemannian Score-Based Generative Model (RSGM) [31] is able to take into account a wide variety of manifolds, such as spheres and toruses, so long as these manifolds satisfy some relatively simple requirements. The Riemannian Smooth Group Model (RSGM) provides evidence that diffusion models can be extended to compact Riemannian manifolds. In addition, the model supplies a formula for inverting the process of diffusion on a manifold.

Taking an intrinsic point of view, the Riemannian Structure Generation Method (RSGM) uses a Geodesic Random Walk to approximate the sampling process on Riemannian manifolds. It is trained with the objective of matching the generalized denoising score.

In contrast, the Riemannian Diffusion Model (RDM) [64] uses a variational framework to generalize the continuous-time diffusion model to Riemannian manifolds. This was accomplished by taking the continuous-time diffusion model and applying it to Riemannian manifolds. The variational lower bound (VLB) of the log-likelihood is employed by the RDM in its capacity as a loss function. The researchers who developed the RDM model demonstrated that achieving the maximum value of this VLB is equivalent to achieving the minimum value of a Riemannian score-matching loss. In contrast to the RSGM, the RDM adopts an extrinsic point of view, working under the assumption that the relevant Riemannian manifold is contained within a Euclidean space with a higher dimension.

5.3.2 Learned Manifolds: Acquired Knowledge of Manifolds The majority of natural data is thought to lie on manifolds that have significantly reduced intrinsic dimensionality, in accordance with the manifold hypothesis [41]. Because of this, locating these manifolds and then training diffusion models directly on them can be beneficial due to the lower dimensionality of the data. Many recent works have expanded upon this concept, beginning with the application of an autoencoder to compress the data into a lower dimensional manifold, then training the model to make it more accurate in this latent space, diffusion models are used. When this occurs, the manifold is discovered through the process of reconstruction loss and is considered to be implicitly defined by the autoencoder. To achieve one's goals, it is essential to develop a loss function that enables the simultaneous training of both the autoencoder and the diffusion models. This is a prerequisite for success.

Combining a Score SDE diffusion model with a variational autoencoder (VAE) [117] is what the Latent Score-Based Generative Model (LSGM) [138] does in order to try and solve the issue of joint training. In this configuration, it is the responsibility of the diffusion model to learn the prior distribution. The authors of the LSGM propose a joint training objective in which the evidence lower bound from the VAE and the score matching objective from the diffusion model are merged into one. As a consequence of this, a new lower bound has been established for the data log-likelihood. By placing the diffusion model within the latent space, the latent space generalized mixture model (LSGM) [138] is able to generate samples more quickly than traditional diffusion models. In addition to this, the LSGM has the capability of managing discrete data by converting it into continuous latent codes.

The Latent Diffusion Model (LDM) [118] takes a component-by-component approach to training the autoencoder and the diffusion model rather than training both components together. In the beginning, a low-dimensional latent space is produced through the training of an autoencoder. The diffusion model is then instructed to produce latent codes through the process of training. DALLE-2 [112] utilizes a strategy that is very similar by first training a diffusion model on the CLIP image embedding space, and then secondly training a separate decoder to create images based on the CLIP image embeddings. Both of these steps take place in the training phase of the algorithm. A study by Asperti et al. (2022) [5] has demonstrated that the image embedding process for denoising generative models produces very good results. Image embedding is a powerful technique that is used in denoising generative models to represent images in a lower-dimensional space. This representation of images in a lower-dimensional space can be used to remove noise and generate clean images. The process of transforming the input images into a lower-dimensional embedding space, which allows for the noise to be removed by using denoising autoencoders, principal component analysis, or diffusion models. The use of image embedding helps to reduce the complexity of the computational work required, improves image quality, and enables the creation of new images.

Chapter 6

CONNECTIONS WITH OTHER GENERATIVE MODELS

First, in this chapter, we will discuss five additional important classes of generative models and then examine the benefits and drawbacks of each of those classes. Then, we demonstrate how the incorporation of diffusion models can help advance these generative models by introducing how they are connected with diffusion models and how they are connected with them. Additionally, Fig. 6 contains a schematic illustration of these algorithms in action.

6.1 Variational Autoencoders and Connections with Diffusion Models

Variational autoencoders [37, 78, 117] are designed to learn both an encoder and a decoder for the purpose of mapping input data to values in a continuous latent space. In these models, the embedding can be interpreted as a latent variable in a probabilistic generative model, and a probabilistic decoder can be formulated by a parameterized likelihood function. Alternatively, the embedding can be interpreted as a hidden variable in a probabilistic generative model. In addition, it is presumed that the data set x was produced by some unobserved latent variable z, and this assumption is supported by the conditional distribution $p_\theta(x \mid z)$.

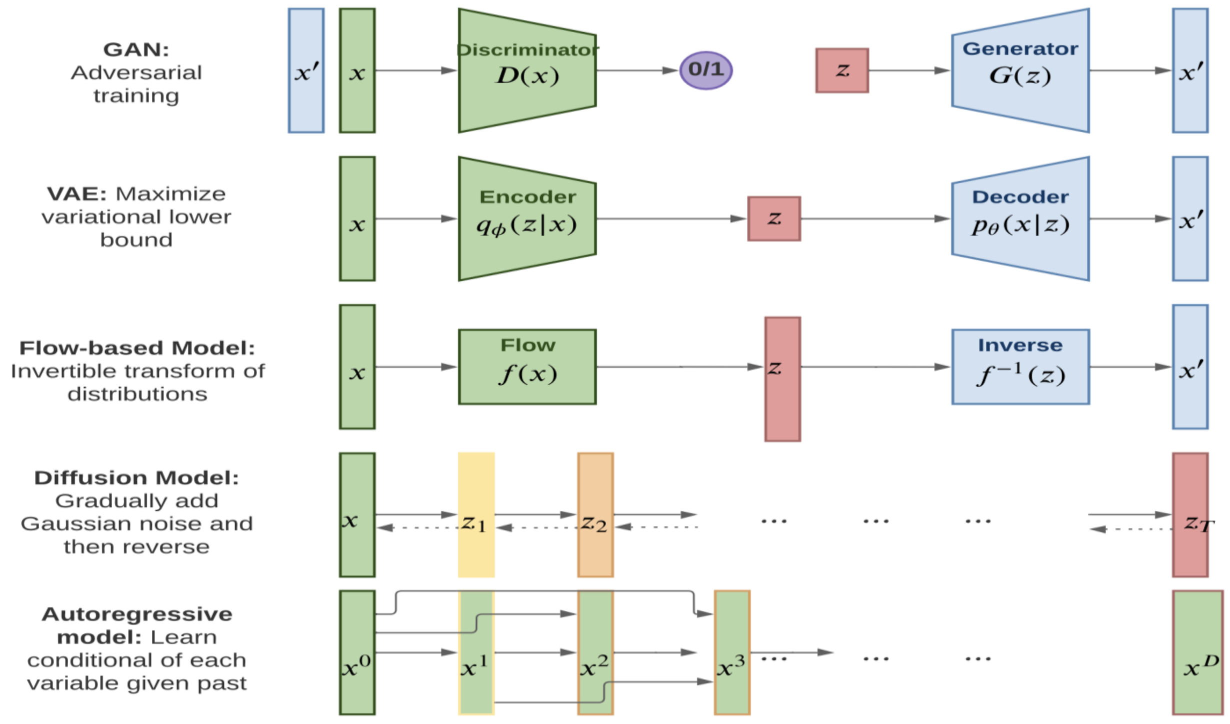

GAN: Adversarial training
x'
x
Discriminator $D(x)$
0/1
z
Generator $G(z)$
x'
VAE: Maximize variational lower bound
x
Encoder $q_\phi(z|x)$
z
Decoder $p_\theta(x|z)$
x'
Flow-based Model: Invertible transform of distributions
x
Flow $f(x)$
z
Inverse $f^{-1}(z)$
x'
Diffusion Model: Gradually add Gaussian noise and then reverse
x
z_1
z_2
z_T
Autoregressive model: Learn conditional of each variable given past
x^0
x^1
x^2
x^3
x^D

Fig.7. Illustrations about works incorporating diffusion models with other generative models.

Moreover, $q_\phi(\mathbf{z} \mid \mathbf{x})$ is applied in order to approximate the value of z. In order to ensure that the inference will be accurate, a variational Bayes method is applied in order to maximise the evidence lower bound:

$$\mathcal{L}(\phi, \theta; \mathbf{x}) = \mathbb{E}_{q(\mathrm{z}|\mathrm{x})}\left[\log p_\theta(\mathrm{x}, \mathrm{z}) - \log q_\phi(\mathrm{z} \mid \mathrm{x})\right]$$

using the formula $\mathcal{L}(\phi, \theta; \mathbf{x}) \leq \log p_\theta(\mathbf{x})$. If the parameterized likelihood function $p_\theta(\mathrm{x} \mid \mathrm{z})$ and the parameterized posterior approximation $q_\phi(\mathbf{z} \mid \mathbf{x})$ can both be computed in a point-wise manner and are differentiable with their parameters, then the ELBO can be maximized with gradient descent. This is provided that both of these functions are differentiable with their parameters. This formulation makes it possible to select encoder and decoder models with a degree of flexibility. These models are typically represented by exponential family distributions, and the parameters of those distributions are generated by multi-layer neural networks.

A hierarchical Markovian VAE with a fixed encoder is one way to think about the DDPM, which can be conceptualized as such. To be more specific, the DDPM's forward process serves as the encoder, and this process is structured as a linear Gaussian model. On the other hand, the DDPM's reverse process corresponds to the decoder, which is shared across multiple steps of the decoding process. The sample data and the latent variables contained within the decoder each have the same amount of information.

Song et al. (2021) [135], Huang et al. (2021) [65], and Kingma et al. (2021) [77] demonstrate that the score matching objective may be approximated by the Evidence Lower Bound (ELBO) of a deep hierarchical VAE in a continuous-time setting. [135], [65], and [77] respectively. As a consequence of this, optimizing a diffusion model can be interpreted in the same manner as training an infinitely deep hierarchical VAE. This finding lends credence to the widely held notion that Score SDE diffusion models can be construed as the continuous limit of hierarchical VAEs.

This line of research is advanced by the Latent Score-Based Generative Model (LSGM) [138], which demonstrates that the ELBO can be regarded as a specialized score matching objective when applied to the problem of latent space diffusion. [LSGM stands for "Latent Score-Based Generative Model"]. Even though the cross-entropy term in the ELBO cannot be solved, it is still possible to transform it into a score matching objective that can be solved by considering the score-based generative model to be an infinitely deep VAE.

6.2 Generative Adversarial Networks and Connections with Diffusion Models

Generative Adversarial Networks, also known as GANs [28, 49, 54], are primarily made up of two different models: a generator G and a discriminator D.

Both of these models are typically built using neural networks, but they could be implemented using any kind of differentiable system that maps input data from one space to another. The optimization of GANs can be thought of as a mini-max optimization problem with the value function V (G, D), which reads as follows:

$$\underset{G}{min}\underset{D}{max}\,\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}(\text{x})}[\log D(\text{x})] + \mathbb{E}_{\text{z}\sim p_{\mathbf{z}}(\text{z})}\left[\log\left(1 - D\big(G(\text{z})\big)\right)\right].$$

The purpose of the generator G is to produce fresh examples while also providing an implicit model of the data distribution. The discriminator D is typically a binary classifier that is utilized in order to differentiate between generated examples and true examples with the highest degree of precision possible. A saddle point is reached at the end of the process of optimization. This point produces a minimum about the generator and a maximum about the discriminator. To be more specific, the objective of GAN optimization is to reach a state of Nash equilibrium [116]. When this condition is met, the generator can be thought of as having successfully captured the accurate distribution of real examples.

One of the problems with GAN is that the training process is unstable, which is primarily caused by the fact that the distribution of input data and that of generated data do not overlap. This is one of the main causes of the instability. One possible solution is to introduce noise into the discriminator input in order to broaden the distributions that can be supported by the generator as well as the discriminator. Wang et al. (2022) [174] inject noise into the discriminator using an adaptive noise schedule that is determined by a diffusion model. This is done in order to take advantage of the flexible diffusion model. On the other hand, GAN can make sampling speed for diffusion models more manageable. Xiao et al. (2021) [148] demonstrate that the Gaussian assumption made during the denoising step, which is justified only for steps with small sizes, is the root cause of slow sampling. As a consequence of this, each step of the denoising process is modelled by a conditional GAN, which enables larger step sizes.

6.3 Normalizing Flows and Connections with Diffusion Models

Normalizing flows are examples of generative models that can model high-dimensional data [53,122]. These models generate tractable distributions. When flows are normalized, a simple probability distribution can be transformed into an extremely complex probability distribution. This extremely complex probability distribution has applications in generative models, reinforcement learning, variational inference, and other areas of study.

The change of variable formula is used as the foundation for the construction of existing normalizing flows [175]. A differential equation is used to describe the trajectory that is produced by normalizing flows. In the setting of discrete time, the mapping from data x to latent z in normalizing flows is a composition of a sequence of bijections, taking the form of $F = F_N \circ F_{N-1} \circ \ldots \circ F_1$. In other words, the mapping is a sequence of bijections.

In normalizing flows, the following conditions must be met for the trajectory denoted by $\{x_1, x_2, \ldots x_N\}$ etc.:

$$\mathbf{x}_i = F_i(\mathbf{x}_{i-1}, \theta), \mathbf{x}_{i-1} = F_i^{-1}(\mathbf{x}_i, \theta)$$

for all $i \leq N$

In a manner analogous to that of the continuous setting, normalizing flows make it possible to retrieve the exact log-likelihood by modifying the formula for the variable in question. The modelling of complex data is hindered, however, by the bijection requirement, which applies in both practical and theoretical settings [27, 144]. Several works [34, 93] make an effort to modify this bijection requirement in some way. For instance, in [157], a generative modelling algorithm called DiffFlow is presented. This algorithm combines the advantages of flow-based models with those of diffusion models. As a consequence of this, DiffFlow generates boundaries that are more distinct than those produced by normalizing flow, and it discovers more general distributions with fewer discretization steps than diffusion probabilistic models.

6.4 Autoregressive Models and Connections with Diffusion Models

Autoregressive Models (ARMs) work by decomposing the joint distribution of data into a product of conditional distributions using the probability chain rule:

$$\log p(\mathbf{x}_{1:T}) = \sum_{t=1}^{T} \log p(x_t \mid \mathbf{x}_{<t})$$

where $\mathbf{x}_{<t}$ is an abbreviation for $x_1, x_2, \dots, x_{t-1}$[80]. Recent developments in deep learning have made it possible for significant progress to be made across a wide variety of data modalities, including image, audio, and text.

The application of a single neural network allows autoregressive models, also known as ARMs, to provide generative capabilities. Taking a sample from one of these models requires the same amount of network calls regardless of the dimensionality of the data. Although ARMs are reliable density estimators, the sampling process is iterative and time-consuming, which is especially problematic when dealing with high-dimensional data.

On the other hand, the Autoregressive Diffusion Model (ARDM) [63] is able to generate arbitrary-order data, which can include order-agnostic autoregressive models and discrete diffusion models as special cases [96, 128]. Instead of employing causal masking on representations such as ARMs, the ARDM is trained with an effective objective that is analogous to that of diffusion probabilistic models. During the testing phase, the ARDM is capable of generating data in parallel, which enables its application to a variety of tasks involving arbitrary generation.

6.5 Energy-based Models and Connections with Diffusion Models

Energy-based Models (EBMs) [18, 45, 103, 110, 160] can be viewed as one generative version of discriminators [51], while also having the capability of learning from unlabeled input data. [45]. A training example is denoted by the notation $\mathbf{x} \sim p_{\text{data}}(\mathbf{x})$, and the probability density function that attempts to approximate pdata is denoted by the notation p_θ(x).A model that is based on energy can be defined as:

$$p_\theta(\mathbf{x}) = \frac{1}{Z_\theta} \exp\big(f_\theta(\mathbf{x})\big)$$

where $Z_\theta = \exp \int \big(f_\theta(\mathbf{x})\big) d\mathbf{x}$ is the partition function, and it is mathematically impossible to analyze for high-dimensional x. In the case of images, the parameterization of f' (x) is handled by a convolutional neural network that has a scalar output. Salimans et al. (2021) [121] compare both constrained score models and energy-based models for modelling the score of the data distribution. They come to the conclusion that constrained score models, i.e., energy based models, can perform just as well as unconstrained models when employing a comparable model structure. This finding was made possible by the fact that unconstrained models were compared to both constrained score models and energy-based models.

There are still two obstacles to overcome when modelling high-dimensional data, despite the fact that EBMs have a number of properties that are desirable. To begin, learning EBMs by maximizing the likelihood requires the use of the MCMC method in order to generate samples from the model, which can be very computationally expensive. Second, as shown in [102], the energy potentials that are learned with non-convergent MCMC are not stable. This means that samples from long-run Markov chains can be significantly different from the observed samples, and as a result, it is difficult to evaluate the learned energy potentials. In a recent piece of research, Gao et al. (2021) [46] present a diffusion recovery likelihood method to learn samples from a sequence of EBMs in a way that is tractable. This method is used in the reverse process of the diffusion model. Each EBM is trained using recovery likelihood, which seeks to maximize the conditional probability of the data at a certain noise level, given their noisy versions at a higher noise level. This is done by comparing the data at the lower noise level with the noisier versions at the higher noise level. EBMs aim to maximize the recovery likelihood rather than the marginal likelihood because the former is easier to work with. This is due to the fact that sampling from conditional distributions is much less difficult than sampling from marginal distributions. This model is capable of generating samples of a high quality, and even after a long run, MCMC samples derived from conditional distributions still look like realistic pictures.

Chapter 7

APPLICATIONS OF DIFFUSION MODELS

Because of their adaptability and robustness, diffusion models have recently begun to be used to tackle a wide range of difficult problems that arise in the real world. We have separated these applications into the following six distinct categories according to the task at hand: computer vision, natural language processing, temporal data modelling, multi-modal learning, and robust learning applications. For each category, we begin with a condensed introduction to the activity, then move on to a comprehensive discussion of the ways in which diffusion models have been utilized to enhance performance. The many different applications of diffusion models are discussed in this chapter.

7.1 Computer Vision

7.1.1. Super Resolution, Inpainting, and Translation. Several image restoration tasks, such as super-resolution, inpainting, and translation, have been successfully tackled by generative models [8, 40, 68, 82,113]. Image super-resolution aims to restore high-resolution images from low-resolution inputs, whereas image inpainting focuses on reconstructing missing or damaged regions in an image. Both of these techniques are used in digital photography.

Diffusion models are utilized by multiple methodologies to accomplish the aforementioned goals. One application of DDPM that enables conditional image generation is Super-Resolution via Repeated Refinement (SR3), for instance. A stochastic and iterative denoising process is used to carry out super-resolution calculations in SR3. The Cascaded Diffusion Model (CDM) [60] is made up of multiple diffusion models that are run one after another, with each one producing images with a higher resolution than the previous one. Both the SR3 and the CDM perform the diffusion process on the input images in a direct manner, which results in more extensive evaluation steps.

Some methods [118, 138] have moved the diffusion process to the latent space using pre-trained autoencoders in order to make it possible to train diffusion models despite having limited computational resources. This makes it possible to train diffusion models. The Latent Diffusion Model (LDM) [118] is a method that improves the quality of denoising diffusion models while simultaneously making the training and sampling processes more efficient. Repaint [89] has an improved denoising strategy that works well for inpainting tasks. This strategy makes use of resampling iterations in order to better condition the image (see Figure Fig. 7). In the meantime, Palette [119] makes use of conditional diffusion models to develop a unified framework for the generation of four different types of images, namely colorization, inpainting, upcropping, and JPEG restoration.

Image translation is primarily concerned with the creation of images that are modelled after particular aesthetic preferences [68]. To achieve a higher level of fidelity, SDEdit [98] applies a stochastic differential equation (SDE) prior. To be more specific, it starts by adding noise to the image that was input, and then it denoises the image by running it through the SDE.

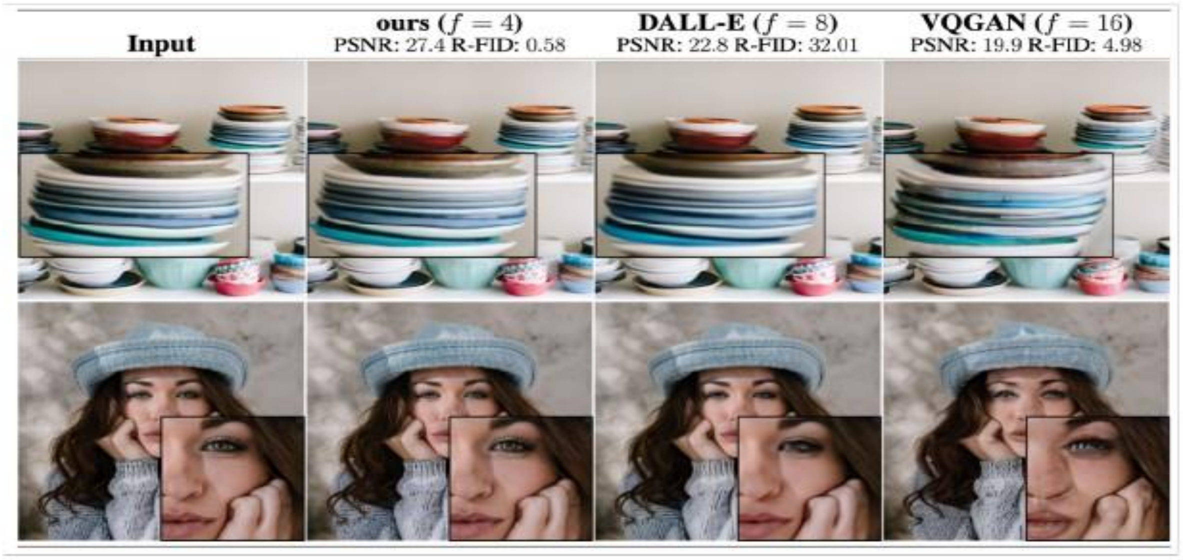


Fig. 8. Image super resolution results produced by LDM [118].

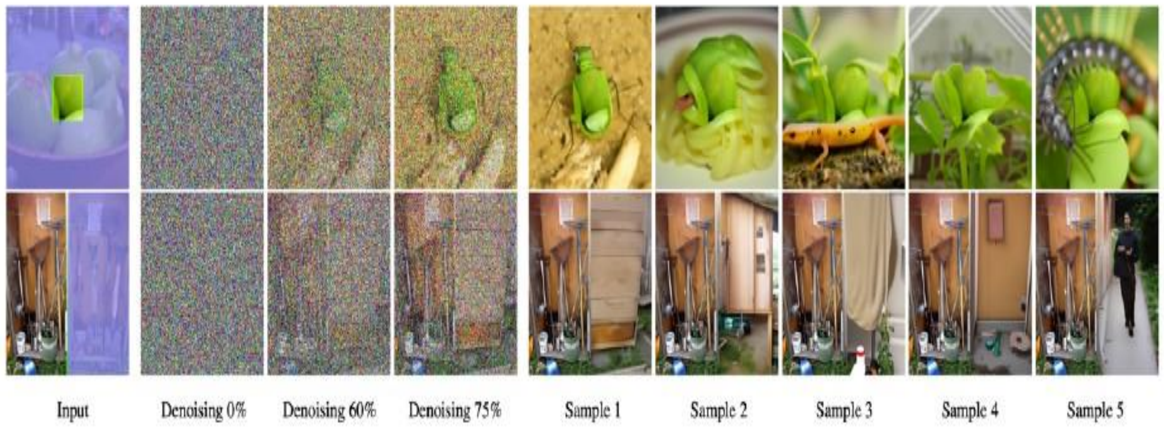


Fig. 9. Image inpainting results produced by Repaint [89].

7.1.2. Semantic Segmentation. The goal of semantic segmentation is to assign a label to each pixel in an image in accordance with predetermined object categories. Recent research has shown that representations learned through DDPM contain high-level semantic information that is useful for segmentation tasks [7, 50, 170]. Generative pre-training can improve the label utilization of semantic segmentation models, and this research has also shown that representations learned through DDPM contain high-level semantic information. Alternatives such as VDVAE and ALAE haven't been able to compete with the success of the few-shot method that makes use of these learned representations. Similar to DDP, Decoder Denoising Pretraining (DDeP) [12] combines diffusion models and denoising autoencoders [141] to achieve label-efficient semantic segmentation with promising results.

7.1.3. Video Generation. Producing videos of a high quality is still difficult in this day and age of deep learning due to the complexity and spatio-temporal continuity of video frames [103, 152]. Diffusion models have been the focus of recent research that aims to improve the overall quality of generated videos [61, 156, 168]. For instance, the Flexible Diffusion Model (FDM) makes use of a generative model to enable the sampling of any arbitrary subset of video frames, given any other subset. This makes it possible to detect motion blur in videos. A is also included in the FDM.

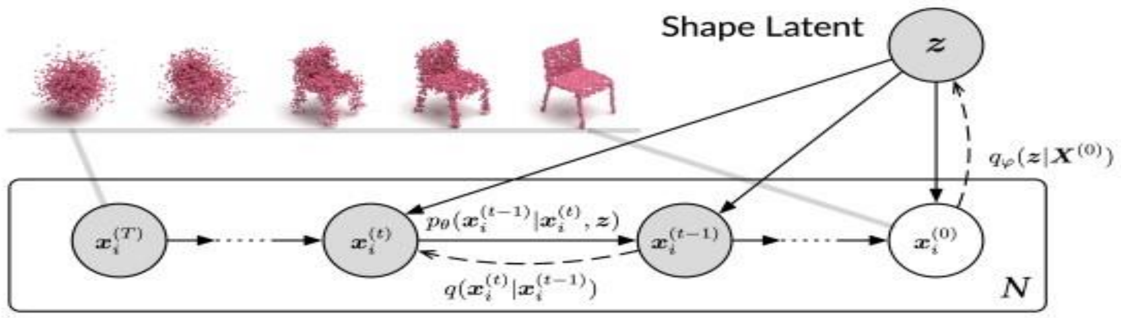


Fig. 10. The directed graphical model of the diffusion process for point clouds [91].

architecture that is specifically tailored to serve this function. Additionally, the Residual Video Diffusion (RVD) model [153] utilizes an autoregressive, end-to-end optimized video diffusion model. It does this by modifying a deterministic prediction of the next frame and making use of a stochastic residual that is produced through an inverse diffusion process. This results in the generation of future frames.

7.1.4. Point Cloud Completion and Generation. When it comes to capturing real-world objects, point clouds are an essential form of 3D representation to have. Nevertheless, scans frequently produce point clouds that are missing points due to either partial observation or self-occlusion.

Recent studies have applied diffusion models to address this challenge, using them to infer missing parts in order to reconstruct complete shapes. This work has repercussions for a variety of tasks that come after it, including 3D reconstruction, augmented reality, and scene comprehension [92].

The approach that was taken by Luo et al. 2021 [91] was to treat point clouds as particles in a thermodynamic system. They used a heat bath to facilitate diffusion from the original distribution to a noise distribution. This approach was successful. In the meantime, the Point-Voxel Diffusion (PVD) model [161] combines denoising diffusion models with the point voxel representation of three-dimensional shapes. The Point Diffusion-Refinement (PDR) model establishes a point-wise mapping between the generated point cloud and the ground truth. It does this by utilizing a conditional DDPM to generate a coarse completion from partial observations.

7.1.5. Anomaly Detection. The Detection of Anomalies Both machine learning and computer vision face the crucial and difficult problem of anomaly detection. It has been demonstrated that generative models are in possession of a potent mechanism for anomaly detection [47, 57, 147], which involves modelling normal or healthy reference data. AnoDDPM [147] makes use of DDPM to corrupt the input image and then reconstructs the image into a more accurate representation of itself. These methods may perform better than alternatives that are based on adversarial training because they can model smaller datasets more effectively and have more stable training schemes. This is because they use effective sampling.

The training process used by DDPM is enhanced by the addition of a substantial number of unsupervised remote sensing images provided by DDPM-CD [47]. Using a pre-trained DDPM and applying the multi-scale representations from the diffusion model decoder, changes in remotely sensed images can be identified and analyzed.

7.2 Natural Language Processing

The goal of natural language processing is to comprehend, model, and manage human languages derived from a variety of sources including text and sound. Text generation has emerged as one of the natural language processing field's most important and difficult tasks [83]. Its goal is to generate plausible and readable text in the human language based on a set of input data, such as a sequence of keywords, or a random noise stream. Text generation has been approached from a number of different angles, many of which are based on diffusion models. Discrete Denoising Diffusion Probabilistic Models, or D3PM, [6] is an algorithm that introduces diffusion-like generative models for character-level text generation. It goes beyond corruption processes with uniform transition probabilities, which is how the multinomial diffusion model [96] is generalized by this theory. It has been demonstrated that large autoregressive language models (LMs) are able to produce high-quality text [13, 23,174]. In order to reliably deploy these LMs in applications that run in the real world, it is typically expected that the text generation process will be able to be controlled. This means that we need to generate text that can meet the requirements that are desired (e.g., topic, syntactic structure).

The problem of controlling the behavior of language models without resorting to retraining is a significant one that arises frequently in the process of text generation. Even though more recent methods have had a lot of success controlling simple sentence attributes (like sentiment) [151], there hasn't been a lot of progress made on more complex and fine-grained controls (e.g., syntactic structure). In order to deal with controls that are more complex, Diffusion-LM [84] suggests a new language model that is based on continuous diffusion. The Diffusion-LM algorithm begins with a string of Gaussian noise vectors and gradually denoises them until they are transformed into vectors that correspond to words. In order to produce hierarchical continuous latent representations, the gradual denoising steps are helpful. This hierarchical and continuous latent variable has the potential to make it possible for simple methods that are based on gradients to accomplish complex control. In Analog Bits [22], the analogue bits that represent the discrete variables are generated, and the sample quality is further improved with self-conditioning and asymmetric time intervals. A new conditional diffusion model is proposed in DiffuSeq [58] as a means of completing more difficult text generation tasks.

7.3 Temporal Data Modeling

7.3.1. Time Series Imputation. Time series data are utilized in a variety of significant applications that take place in the real world [39, 106, 152]. Despite this, time series frequently have values that are absent for a variety of reasons, including those that are the result of mechanical or artificial errors [100]. Imputation techniques have undergone significant development in recent years, particularly in the areas of deterministic imputation [95] and probabilistic imputation [42] as well as diffusion-based methods. This paper presents a novel method for imputation of time series that makes use of score-based diffusion models and is referred to as Conditional Score-based Diffusion models for Imputation (CSDI) [137].

To be more specific, it takes on the form of self-supervised training to optimize diffusion models in order to achieve the goal of capitalizing on correlations that are present within temporal data. The use of it in some real-world datasets has shown that it is superior to methods that have been used in the past.

A novel probabilistic framework is proposed in Controlled Stochastic Differential Equation (CSDE) [107] for modelling stochastic dynamics with a neural-controlled stochastic differential equation. Structured State Space Diffusion (SSSD) [1] is an integration of conditional diffusion models and structured state-space models [52]. Its primary purpose is to specifically capture long-term dependencies in time series. It is effective in both the imputation of time series and the forecasting of future events.

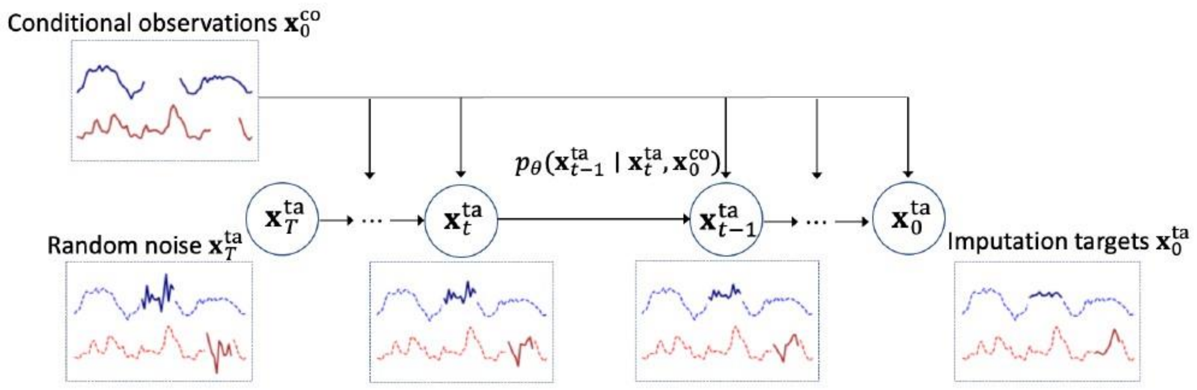

Conditional observations $\mathbf{x}_0^{\text{co}}$
$p_\theta(\mathbf{x}_{t-1}^{\text{ta}} \mid \mathbf{x}_t^{\text{ta}}, \mathbf{x}_0^{\text{co}})$
Random noise $\mathbf{x}_T^{\text{ta}}$
$\mathbf{x}_T^{\text{ta}}$
$\mathbf{x}_t^{\text{ta}}$
$\mathbf{x}_{t-1}^{\text{ta}}$
$\mathbf{x}_0^{\text{ta}}$
Imputation targets $\mathbf{x}_0^{\text{ta}}$

Fig. 11. The procedure of time series imputation with CSDI [137].

7.3.2 Time Series Forecasting. Forecasting based on Time Series The process of forecasting or predicting the future value over a specified amount of time is referred to as time series forecasting. In recent years, neural methods have gained widespread popularity as a solution to the problem of making accurate predictions using univariate point forecasting methods [105] or univariate probabilistic methods [122]. In the setting of multiple variables, we also have point forecasting methods and probabilistic methods, which explicitly model the data distribution by utilizing Gaussian copulas [123], generalized additive models [155], or normalizing flows [115]. An autoregressive model is presented in TimeGrad [114] for the purpose of forecasting multivariate probabilistic time series. This model takes a sample from the data distribution at each time step by estimating the gradient of that distribution. It makes use of diffusion probabilistic models, which are intrinsically linked to score matching and energy-based approaches. Specifically, it learns gradients by optimizing a variational bound on the likelihood of the data, and during inference time it converts white noise into a sample of the distribution of interest through a Markov chain using Langevin sampling [132]. Both of these processes take place through a Markov chain.

7.3.3. Waveform Signal Processing. In electronics, acoustics, and other fields that are related, the waveform of a signal is denoted by the shape of the graph that represents the signal as a function of time. This shape is independent of the signal's time scale and magnitude scale. Wave Grad [19] is a publication that presents a conditional model for the generation of waveforms. This model estimates gradients of the data density. It takes in a signal that is characterized by Gaussian white noise as its input and then uses a gradient-based sampler to iteratively improve the signal. Wave Grad makes a connection between nonauto regressive and autoregressive models in terms of audio quality. This is accomplished by adjusting the number of refinement steps, which causes a natural tradeoff between inference speed and sample quality. DiffWave [79] demonstrates a diffusion probabilistic model that is both flexible and efficient, and it can generate waveforms either conditionally or unconditionally. The model is non-autoregressive and has an effective training process that involves efficiently optimizing a variational bound on the data likelihood. In addition to this, it is capable of producing audio of a high-fidelity in a variety of waveform generation tasks, such as class-conditional generation and unconditional generation.

7.4 Multi-Modal Learning

7.4.1. Text-to-Image Generation. Vision-language models have attracted a lot of attention recently due to the number of potential applications. Text-to-Image generation is the task of generating a corresponding image from a descriptive text [75, 139]. An example is shown in Fig. 8. Blended diffusion [4] utilizes both pre-trained DDPM [33] and CLIPmodels, and it proposes a solution for region-based image editing for general purposes, which uses natural language guidance and is applicable to real and diverse images. On the other hand, unCLIP (DALLE-2) [112] proposes a two-stage approach, a prior model that can generate a CLIP-based image embedding conditioned on a text caption, and a diffusion-based decoder that can generate an image conditioned on the image embedding. Recently, Imagen [120] proposes a text-to-image diffusion model and a comprehensive benchmark for performance evaluation. It shows that Imagen performs well against the state-of-the-art approaches including VQ-GAN+CLIP [29], Latent Diffusion Models [88], and DALL-E 2 [112]. Inspired by the ability of guided diffusion models [33] to generate photorealistic samples and the ability of text-to-image models to handle free-form prompts, GLIDE [100] applies guided diffusion to the application of text-conditioned image synthesis. Many researches like VQ-Diffusion [53] proposes a vector-quantized diffusion model for text-to-image generation, and Q-Diffusion [169] to text-guided image generation it eliminates the unidirectional bias and avoids accumulative prediction errors.

7.4.2. Text-to-Audio Generation. Generation of Audio from Written Text The process of converting regular language texts into voice outputs is referred to as text-to-audio generation [81, 145]. A novel text-to-speech model called Grad-TTS [109] is presented, which includes a score-based decoder and diffusion models. It gradually alters the noise that was predicted by the encoder, and after that, it is aligned with the text that was input using a technique called Monotonic Alignment Search [111]. Grad-TTS2 [76] is an adaptive improvement that was made to Grad-TTS. Diffsound presents a non-autoregressive decoder that is based on the discrete diffusion model [6, 128]. This decoder predicts all of the mel-spectrogram tokens in each and every step, and then refines the predicted tokens in the steps that come after them. EdiTTS [136] makes use of a score-based text-to-speech model in order to fine-tune a mel-spectrogram prior that has been roughly altered. ProDiff [66] parameterizes the denoising diffusion model by directly predicting the clean data. This is in contrast to the traditional method of estimating the gradient of data density.

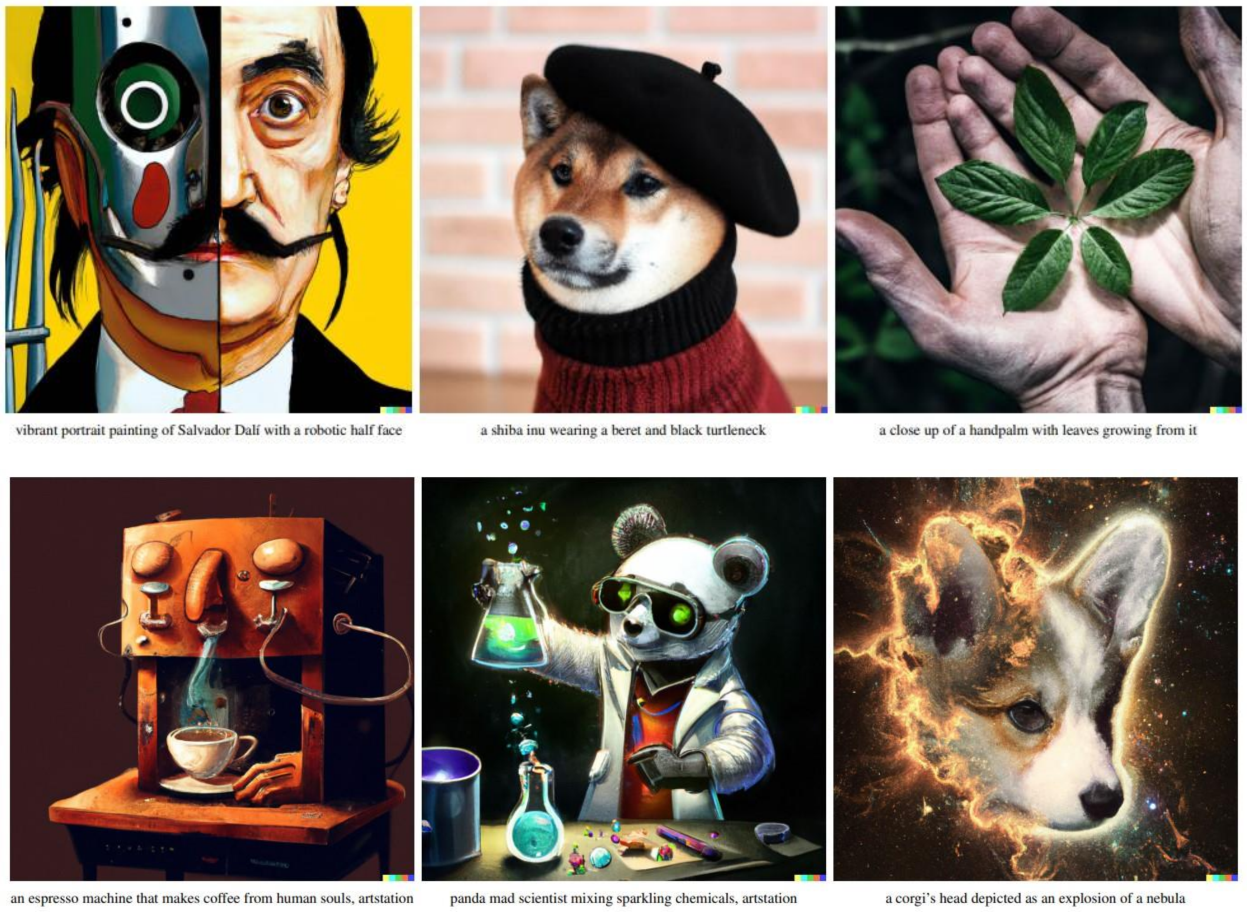

vibrant portrait painting of Salvador Dalí with a robotic half face
a shiba inu wearing a beret and black turtleneck
a close up of a handpalm with leaves growing from it
an espresso machine that makes coffee from human souls, artstation
panda mad scientist mixing sparkling chemicals, artstation
a corgi's head depicted as an explosion of a nebula

Fig. 12. Text-to-image result generated by DALL-E-2 [112].

7.5 Robust Learning

Learning networks that are resilient in the face of adversarial perturbations or noises are referred to as "robust" [11, 101, 149, 171]. Robust learning is a class of defensive methods that helps learning networks. While adversarial training is considered to be a standard method for defending against adversarial attacks for image classifiers, adversarial purification [101], which transforms attacked images into clean images using a standalone purification model, has demonstrated significant performances as an alternative method for defending against adversarial attacks [154]. After receiving an adversarial example, the DiffPure [101] algorithm first applies a forward diffusion process to it, which generates a small amount of noise, and then it applies a reverse generative process, which restores the original, clean image. Adaptive Denoising Purification, or ADP [154], demonstrates that an EBM that has been trained with denoising score matching [140] can effectively purify images that have been attacked in a relatively short amount of time. In addition to this, it suggests an efficient method of randomised purification, which involves the addition of random noises to images prior to purification. [9] presents a new stochastic diffusion-based pre-processing robustification called Projected Gradient Descent (PGD). Its goal is to be a model-agnostic adversarial defence and produce a high-quality denoised outcome. In addition, some works suggest using a guided diffusion process for advanced adversarial purification [149].

7.6 Bioinformatics

In recent years, a great number of diffusion models have begun to be utilized in an effort to solve a variety of issues in the field of bioinformatics [43]. These issues include denoising cryo-EM data, single-cell gene expression analysis, protein design, drug and small molecule design, and modeling of protein-ligand interactions. This demonstrates the enormous potential that diffusion-based models have in the field of bioinformatics because these models routinely perform better than their forerunners, such as VAE and GAN. The purpose of this review is to provide a comprehensive analysis of diffusion models as well as a survey of their most recent applications in bioinformatics.

7.6.1. Cryo-EM data analysis. Cryo-EM, which stands for single particle cryo-electron microscopy, is one of the most important imaging techniques for determining and visualizing the three-dimensional conformation (structure) of large biomolecular complexes (such as protein complexes) at the atomic resolution. Cryo-electron microscopy (cryo-EM) images of protein complexes can be used to reconstruct their three-dimensional conformation, which is represented by three-dimensional density maps.

CryoDRGN [164] is a method for the reconstruction of the structure of protein complexes. As part of this method, a latent variable Z was introduced to define a conformational space V for a protein complex based on cryo-EM density maps. The conventional CryoDRGN algorithm, which is based on the Variational Autoencoder (VAE) framework, is trained to discover a continuous distribution in the latent space for protein structures using cryo-EM data. However, the fact that the Gaussian prior distribution of VAE does not match the aggregate approximation posterior restricts the generative capability of the model, despite the fact that CryoDRGN is capable of simulating complicated structural dynamics. A high-quality generative model for protein conformations can now be learned directly from cryo-EM imaging data thanks to the recent addition of a continuous-time diffusion model (also known as Score SDEs) into CryoDRGN. With its expressive denoising diffusion generative models, the brand-new CryoDRGN [163] is able to better capture the 3D conformations of protein complexes and reconstruct 3D conformation of a higher quality.

7.6.2. Protein design and generation. In order to design novel proteins, such as enzymes, that are more efficient at performing certain functions than natural proteins, it is necessary to first determine the structure of a protein and then design protein sequences that can fold into that structure (also known as the inverse protein folding problem). RFDiffusion is capable of constructing a DDPM generative model of protein backbones. This model can then be put to use in the process of designing protein monomers, symmetric protein oligomers, enzyme active site scaffolding, and symmetric motif scaffolding. It has been put to use in the development of therapeutic proteins and proteins that bind metals. Some of the proteins that were designed using RFDiffusion have been validated experimentally.

To solve the motif-scaffolding problem, which requires designing a scaffold structure based on a protein motif, another diffusion model called SMCDiff was proposed. After employing the diffusion-based model ProtDiff to make an initial prediction of the protein backbone, it then makes use of the SMCDiff algorithm in order to produce scaffolds for arbitrary motifs. It is capable of producing scaffolds that are longer and more diverse than the methods that came before it.

ProteinSGM[165] is a score-based generative diffusion model that generates images-like 2D matrices consisting of inter-residue pairwise distances and angles to represent protein backbone structures. These matrices can be viewed as a representation of protein backbone structures. After that, the matrices are used to generate native-like protein3D structures, which can then be utilized in the design of novel proteins. In a similar manner, Foldingdiff makes use of a DDPM model in conjunction with a vanilla transformer model in order to generate protein sequences that can fold into a backbone protein structure. Even though the folds of the designed protein sequences had not been verified by experiments such as x-ray crystallography, some of the generated protein sequences were put through protein structure prediction tools (such as AlphaFold2) to see if they were capable of folding into the desired protein structure. These tools checked to see if the generated protein sequences could form the desired protein structure.

Chorma [166] is a diffusion-based generative model that can generate large protein singletons (with more than 3000 residues) as well as protein complexes. To control the generation of the structure, the model can take into account a variety of constraints, such as the distance between adjacent residues, symmetry, and shape. It does this by employing a backbone diffusion network to model the transformation of a collapsed polymer system into a protein complex backbone and a graph-based design network to generate protein sequences conditionally on the sampled backbone. Both of these networks are connected via a graph. It makes use of a random graph neural network, which, in comparison to some earlier work, allows it to reduce the time complexity from O(N2) or O(N3) down to O(N). As a result, it is able to generate large proteins and complexes in an effective manner.

A generative model that was designed in previous works introduces a fully data-driven DDPM 1 model for the purpose of generating realistic proteins across the full range of structural domains in the Protein Data Bank (PDB). This model was designed for use in previous work. It is able to achieve equivariance to rotations as well as translations of protein structures thanks to its invariant point attention modules. In order to generate discrete protein sequences, it makes use of an approach that is analogous to masked language modeling. In contrast to some of the methods that came before it, it is capable of producing large proteins with multiple domain topologies.

The Dynamic Graph Score Matching (DGSM) algorithm was initially developed for the purpose of predicting stable 3D conformations from 2D molecular graphs in the field of computational chemistry. Subsequently, the algorithm was extended to the prediction of protein sidechain conformation and multi-molecular complex prediction. It does this by dynamically building graph structures according to the atom-atom spatial proximity, which allows it to model both local and long-range interactions. The model can directly estimate the gradient fields of the logarithm density of atomic coordinates, particularly when the score matching method is used. The model is capable of receiving instruction from beginning to end. The new model has the potential to address, to a large extent, the limitation of traditional experimental and physics-based simulation methods, which is the tendency to ignore the long-range interactions that can occur between non-bounded atoms.

7.6.3. Small molecule generation and drug design. It is essential for both fundamental biomedical research and the development of new drugs to design small molecules (such as drugs) that can mediate the structure and function of proteins. One of the methods that can be utilized in order to find new small molecules, such as potential drug candidates, in three-dimensional space is known as fragment-based drug design. When you start with molecular fragments, the objective is to design linkers that are composed of atoms so that you can combine the fragments into a functionally complete molecule. An E(3)-equivariant 3D-conditional diffusion model is utilized by DiffLinker[167] in order to generate molecular linkers for the purpose of connecting the various molecular fragments. A graph neural network is used to predict the size of the linker, and an equivariant diffusion model is used to generate the linker that connects the input fragments. Together, these two components make up the system. Not only is it able to generate linkers for multiple fragments, but it can also generate the number of atoms contained within the linker as well as the attachment points for the fragments that are fed into it.

In order to produce small molecules, the E(3)-Equivariant Diffusion Model (EDM) applies the diffusion process to the atom coordinates and atom types in the Euclidean space. The previous methods had a limit of nine heavy atoms for the largest molecule structures they could generate, but this new method can generate molecules with as many as 29 atoms.

The equivariant graph neural network (EGNN) and the diffusion process are both incorporated into this solution. The first method uses geometric symmetries to model the structures of molecules, while the second method simplifies training in order to improve both performance and scalability. A simple yet novel diffusion-based generative model called the Diffusion Informative Prior Bridge was designed to guide the diffusion process in order to generate high-quality and realistic molecules. This model was inspired by the physics that govern the formation of small molecules, and it was created in order to guide the diffusion process and generate high-quality molecules. In order to improve the generation of molecules as well as uniformity-enhanced 3D point clouds, several energy functions are integrated with the physical and statistical prior information.

7.7 Interdisciplinary Applications

7.7.1. Molecular Graph Modeling. The Molecular Graph Modeling Procedure Graph Neural Networks [55] and corresponding representation learning [56] techniques have had a lot of success [10, 146] in a lot of different fields, including modelling molecule graph in various tasks ranging from property prediction [38, 48] to molecule generation [69, 73, 93, 125], where a molecule is naturally represented by a node-edge graph. This includes modelling molecule graph In spite of the fact that they are effective in a variety of applications, more fundamental and informative properties are beginning to be combined with diffusion models in order to improve molecular graph modelling.

The paper "Torsional Diffusion" presents a new diffusion framework that performs operations on the space of torsion angles using a diffusion process on the hyperspace and an extrinsic-to-intrinsic scoring model. Torsional diffusion [173] can be found in aforementioned paper. Markov chains evolving with equivariant Markov kernels can produce an invariant distribution, as demonstrated by GeoDiff [172], which also designs blocks for the Markov kernels to maintain the desirable equivariance property. There are also other works that incorporate the equivariance property into the generation of 3D molecules [62] and proteins [2, 9]. ConfGF [124] directly estimates the gradient fields of the log density of atomic coordinates in molecular conformation generation. This method was motivated by the classical force field methods for simulating molecular dynamics.

7.7.2. Material Design. The Design of Materials Solid state materials are the essential building blocks for the foundation of a great many important technologies [14]. Crystal Diffusion Variational Autoencoder (CDVAE) [149] is an algorithm that incorporates stability as an inductive bias by proposing a noise.

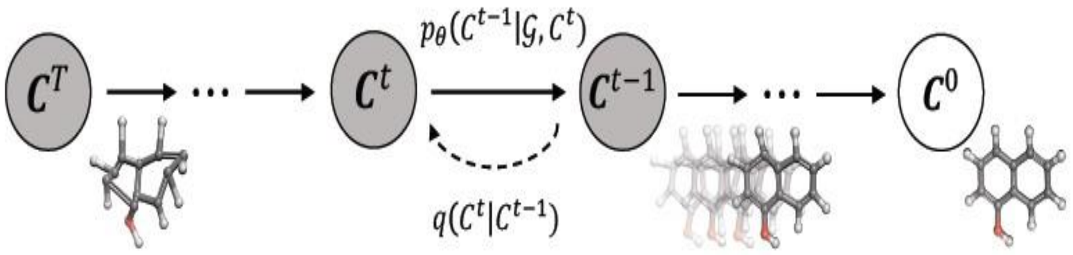


Fig. 13. Molecule-to-conformation diffusion process in GeoDiff [172]

conditional score network that makes use of permutation, translation, rotation, and periodic invariance properties all at the same time. To generate antibodies with atomic resolution, Luo et al. (2022) [94] model sequences and structures of complementarity-determining regions using equivariant diffusion. Additionally, they explicitly target specific antigen structures.

7.7.3. Medical Image Reconstruction. The process of recovering an unknown signal from observed measurements is referred to as an inverse problem. This is a significant challenge in the field of medical image reconstruction, specifically for computed tomography (CT) and magnetic resonance imaging (MRI) [16, 24, 25, 108, 134, 150, 160]. A score-based generative model was used by Song et al. (2021) [134] in order to reconstruct an image that is consistent with both the prior measurements and the observed measurements. Chung et al. (2022) [26] train a continuous time-dependent score function with denoising score matching and iterate between the numerical SDE solver and data consistency step for reconstruction at the evaluation stage. This is done in order to reconstruct the data. Peng et al. (2022) [108] perform MR reconstruction by gradually guiding the reverse-diffusion process given observed k-space signal, and they propose a coarse-to-fine sampling algorithm for efficient sampling. This method was developed to improve the accuracy of MR imaging.

Chapter 8

FUTURE DIRECTIONS and CONCLUSION

FUTURE DIRECTIONS: The study of diffusion models is still in its infancy, and there is a significant amount of room for development in both the theoretical and empirical facets of the subject. As was covered in earlier chapters, key directions for future research include efficient sampling and improved likelihood, as well as the investigation of how diffusion models can handle special data structures, interface with various other types of generative models, and be adapted to a variety of applications. In addition, we believe that the scope of future research on diffusion models will most likely extend into the following areas of investigation.

Revisiting Assumptions. A Look Back at the Presumptions It is necessary to reconsider and evaluate a great number of the standard assumptions found in diffusion models. For instance, the assumption that the forward process of diffusion models completely erases any information in data and renders it equivalent to a prior distribution may not always hold true. This is because the forward process of diffusion models occurs before the prior distribution. In reality, it is impossible to erase all traces of information in a period of time that is finite. It is of great interest to understand when to stop the forward noise process in order to strike a balance between sampling efficiency and sample quality [44]. Understanding when to stop the process is essential. Recent developments in Schrodinger bridges and optimal transport [21, 30, 32, 126, 130] provide promising alternative solutions. These developments suggest new formulations for diffusion models that are capable of converging to a specified prior distribution in finite time. The use of diffusion models has significantly advanced the state of the art in a number of significant areas of bioinformatics and has gained traction in a number of important bioinformatics domains. The use of diffusion models will continue to rapidly expand in the field of bioinformatics due to their superior capabilities, in comparison to those of other generative models, of both denoising existing data and generating new data that is more realistic. The field of bioinformatics possesses a vast quantity of issues that can be solved by using diffusion models.

Theoretical Understanding. Theoretical Comprehending of the Situation Diffusion models have recently emerged as a powerful framework, particularly because they are the only ones that can compete with generative adversarial networks (GANs) in the majority of applications without the need to engage in adversarial training. Understanding why and when diffusion models are more effective than alternative approaches for particular tasks is essential to making effective use of this potential. It is essential to determine the fundamental characteristics that distinguish diffusion models from other types of generative models, such as variational autoencoders, energy-based models, or autoregressive models.

If you are able to grasp these distinctions, you will be better able to comprehend why diffusion models are able to produce samples of such high quality while still achieving the highest likelihood. To a similar degree of significance is the requirement to develop theoretical guidance for selecting and determining various hyperparameters of diffusion models in a systematic manner.

Latent Representations. Latent representations are an essential aspect of diffusion models, as they capture the underlying structure of the data and allow for efficient modeling and prediction. In diffusion models, a latent representation is a low-dimensional space that describes the variations in the observed data. By projecting the data into this space, the model can capture the essential features of the data and perform tasks such as dimensionality reduction, clustering, and classification.

The construction of latent representations in diffusion models can be achieved using various techniques such as principal component analysis (PCA), autoencoders, and variational autoencoders. PCA is a linear technique that decomposes the data into its principal components, which are the directions of maximum variance. Autoencoders and variational autoencoders are non-linear techniques that learn a mapping between the observed data and the latent space using neural networks. These techniques can learn complex non-linear relationships in the data, making them particularly useful for modeling complex systems.

Once the latent representation is constructed, it can be used to model the diffusion process. In diffusion models, the observed data is assumed to be generated by a diffusion process that evolves over time. The diffusion process can be modeled using a stochastic differential equation, where the latent representation is the state variable. By modeling the diffusion process using the latent representation, the model can capture the temporal dependencies in the data and make predictions about future observations.

Overall, latent representations are a critical component of diffusion models, as they allow for efficient modeling and prediction of complex systems. These representations can be constructed using various techniques such as PCA, autoencoders, and variational autoencoders and can be used to model the diffusion process and capture the underlying structure of the data.

CONCLUSION

We have presented an extensive analysis of diffusion models from a variety of perspectives. diffusion models, including DDPM, SGM, NCNS, and SDE, have become a popular tool for modeling complex systems in various fields. These models have benefited from recent advances in sampling efficiency, likelihood maximization, and new techniques for handling data with special structures.

The use of techniques such as Hamiltonian Monte Carlo, Langevin dynamics, and Stein variational gradient descent has improved the sampling efficiency of diffusion models. This has allowed for the generation of high-quality samples from complex distributions, which is critical for accurate modeling and prediction.

Likelihood maximization has also been an essential aspect of diffusion modeling, and methods such as maximum likelihood estimation, variational inference, and Bayesian inference have been used to estimate model parameters and compare different models.

New techniques for handling data with special structures have also been developed, including those for time-series data, high-dimensional data, and data with missing values. These techniques have improved the ability of diffusion models to accurately model and predict complex systems.

DDPM has gained popularity in the machine learning community due to its ability to handle multi-modal distributions and generate high-quality images. SGM has been applied to study the dynamics of physical systems and is known for its ability to capture non-Gaussian behavior. NCNS has been used to model biological systems, such as neuronal activity, and incorporates nonparametric noise models. Finally, SDE has been extensively used to model financial markets and can capture the stochastic dynamics of asset prices.

In summary, diffusion models have benefited from recent advancements in sampling efficiency, likelihood maximization, and new techniques for handling data with special structures. These models, including DDPM, SGM, NCNS, and SDE, have become a powerful tool for modeling complex systems, and future research will undoubtedly continue to refine and expand these models to better understand the behavior of complex systems.

Bibliography

1. Alcaraz, J. M., & Strodthoff, N. (2022). Diffusion-based Time Series Imputation and Forecasting with Structured State Space Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2208.09399.
2. Anand, N., & Achim, T. (2022). Protein Structure and Sequence Generation with Equivariant Denoising Diffusion Probabilistic Models. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2205.15019.
3. Uri M Ascher and Linda R Petzold. 1998. Computer methods for ordinary differential equations and differential-algebraic equations. ISBN 161197139X, 9781611971392 [Vol. 61] Siam.
4. Avrahami, O., Lischinski, D., & Fried, O. (2021). Blended Diffusion for Text-driven Editing of Natural Images. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2111.14818.
5. Asperti, A., Evangelista, D., Marro, S., & Merizzi, F. (2022). Image Embedding for Denoising Generative Models. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2301.07485.
6. Bao, F., Li, C., Zhu, J., & Zhang, B. (2022). Analytic-DPM: An Analytic Estimate of the Optimal Reverse Variance in Diffusion Probabilistic Models. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2201.06503.
7. Baranchuk, D., Rubachev, I., Voynov, A., Khrulkov, V., & Babenko, A. (2021). Label-Efficient Semantic Segmentation with Diffusion Models. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2112.03126.
8. Batzolis, G., Stanczuk, J., Schönlieb, C., & Etmann, C. (2021). Conditional Image Generation with Score-Based Diffusion Models. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2111.13606.
9. Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., & Bourne, P. E. (2000). The Protein Data Bank. Nucleic acids research, 28(1), 235–242. https://doi.org/10.1093/nar/28.1.235.
10. Bielak, P., Kajdanowicz, T., & Chawla, N. V. (2021). Graph Barlow Twins: A self-supervised representation learning framework for graphs. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2106.02466.
11. Blau, T., Ganz, R., Kawar, B., Bronstein, A., & Elad, M. (2022). Threat Model-Agnostic Adversarial Defense using Diffusion Models. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2207.08089.
12. Brempong, E. A., Kornblith, S., Chen, T., Parmar, N., Minderer, M., & Norouzi, M. (2022). Denoising Pretraining for Semantic Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 4175-4186).
13. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., . . . Amodei, D. (2020).

Language Models are Few-Shot Learners. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2005.14165.
14. Butler, K.T., Davies, D.W., Cartwright, H. et al. Machine learning for molecular and materials science. Nature 559, 547–555 (2018). https://doi.org/10.1038/s41586-018-0337-2.
15. Campbell, A., Benton, J., De Bortoli, V., Rainforth, T., Deligiannidis, G., & Doucet, A. (2022). A Continuous Time Framework for Discrete Denoising Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2205.14987.
16. Cao, C., Cui, Z., Liu, S., Zheng, H., Liang, D., & Zhu, Y. (2022). High-Frequency Space Diffusion Models for Accelerated MRI. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2208.05481.
17. Grathwohl, W., Chen, R. T., Bettencourt, J., & Duvenaud, D. (2019, May). Scalable reversible generative models with free-form continuous dynamics. In International Conference on Learning Representations.
18. Che, T., Zhang, R., Larochelle, H., Paull, L., Cao, Y., & Bengio, Y. (2020). Your GAN is Secretly an Energy-based Model and You Should use Discriminator Driven Latent Sampling. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2003.06060.
19. Chen, N., Zhang, Y., Zen, H., Weiss, R. J., Norouzi, M., & Chan, W. (2020). WaveGrad: Estimating Gradients for Waveform Generation. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2009.00713.
20. Chen, R. T., Rubanova, Y., Bettencourt, J., & Duvenaud, D. (2018). Neural Ordinary Differential Equations. [Vol-5] ArXiv. https://doi.org/10.48550/arXiv.1806.07366.
21. Chen, T., Liu, G., & Theodorou, E. A. (2021). Likelihood Training of Schr\"odinger Bridge using Forward-Backward SDEs Theory. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2110.11291.
22. Chen, T., Zhang, R., & Hinton, G. (2022). Analog Bits: Generating Discrete Data using Diffusion Models with Self-Conditioning. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2208.04202.
23. Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., . . . Fiedel, N. (2022). PaLM: Scaling Language Modeling with Pathways. [Vol-5] ArXiv. https://doi.org/10.48550/arXiv.2204.02311.
24. Chung, H., Lee, E. S., & Ye, J. C. (2022). MR Image Denoising and Super-Resolution Using Regularized Reverse Diffusion. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2203.12621.
25. Chung, H., Sim, B., & Ye, J. C. (2021). Come-Closer-Diffuse-Faster: Accelerating Conditional Diffusion Models for Inverse Problems through Stochastic Contraction. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2112.05146.
26. Chung, H., & Ye, J. C. (2022). Score-based diffusion models for accelerated MRI. Medical Image Analysis, 80, 102479. https://doi.org/10.1016/j.media.2022.102479.

27. Cornish, R., Caterini, A. L., Deligiannidis, G., & Doucet, A. (2019). Relaxing Bijectivity Constraints with Continuously Indexed Normalising Flows. [Vol-5] ArXiv. https://doi.org/10.48550/arXiv.1909.13833.
28. Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., & Bharath, A. A. (2017). Generative Adversarial Networks: An Overview. [Vol-1] ArXiv. https://doi.org/10.1109/MSP.2017.2765202.
29. Crowson, K., Biderman, S., Kornis, D., Stander, D., Hallahan, E., Castricato, L., & Raff, E. (2022). VQGAN-CLIP: Open Domain Image Generation and Editing with Natural Language Guidance. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2204.08583.
30. Heng, J., De Bortoli, V., Doucet, A., & Thornton,J. (2021). Simulating Diffusion Bridges with Score Matching. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2111.07243.
31. De Bortoli, V., Mathieu, E., Hutchinson, M., Thornton, J., Teh, Y. W., & Doucet, A. (2022). Riemannian Score-Based Generative Modelling. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2202.02763.
32. De Bortoli, V., Thornton, J., Heng, J., & Doucet, A. (2021). Diffusion Schr\"odinger Bridge with Applications to Score-Based Generative Modeling. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2106.01357.
33. Dhariwal, P., & Nichol, A. (2021). Diffusion Models Beat GANs on Image Synthesis. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2105.05233.
34. Dinh, L., Larochelle, H., & Pascanu, R. (2019). A RAD approach to deep mixture models. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.1903.07714.
35. Dockhorn, T., Vahdat, A., & Kreis, K. (2021). Score-Based Generative Modeling with Critically-Damped Langevin Diffusion. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2112.07068.
36. Dockhorn, T., Vahdat, A., & Kreis, K. (2022). GENIE: Higher-Order Denoising Diffusion Solvers. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2210.05475.
37. Doersch, C. (2016). Tutorial on Variational Autoencoders. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.1606.05908.
38. Duvenaud, D., Maclaurin, D., Hirzel, T., & Adams, R. P. (2015). Convolutional Networks on Graphs for Learning Molecular Fingerprints. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.1509.09292.
39. Eldele, E., Ragab, M., Chen, Z., Wu, M., Kwoh, C. K., Li, X., & Guan, C. (2021). Time-Series Representation Learning via Temporal and Contextual Contrasting. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2106.14112.
40. Esser, P., Rombach, R., & Ommer, B. (2020). Taming Transformers for High-Resolution Image Synthesis. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2012.09841.
41. Fefferman, C., Mitter, S., & Narayanan, H. (2013). Testing the Manifold Hypothesis. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.1310.0425.
42. Fortuin, V., Baranchuk, D., Rätsch, G., & Mandt, S. (2019). GP-VAE: Deep Probabilistic Time Series Imputation. [Vol-5] ArXiv. https://doi.org/10.48550/arXiv.1907.04155.
43. Guo, Z., Liu, J., Wang, Y., Chen, M., Wang, D., Xu, D., & Cheng, J. (2023). Diffusion Models in Bioinformatics: A New Wave of Deep Learning Revolution in Action. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2302.10907.

44. Franzese, G., Rossi, S., Yang, L., Finamore, A., Rossi, D., Filippone, M., & Michiardi, P. (2022). How Much is Enough? A Study on Diffusion Times in Score-based Generative Models. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2206.05173.
45. Gao, R., Nijkamp, E., Kingma, D. P., Xu, Z., Dai, A. M., & Wu, Y. N. (2019). Flow Contrastive Estimation of Energy-Based Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.1912.00589.
46. Gao, R., Song, Y., Poole, B., Wu, Y. N., & Kingma, D. P. (2020). Learning Energy-Based Models by Diffusion Recovery Likelihood. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2012.08125.
47. Bandara, W. G., Nair, N. G., & Patel, V. M. (2022). DDPM-CD: Remote Sensing Change Detection using Denoising Diffusion Probabilistic Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2206.11892.
48. Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural Message Passing for Quantum Chemistry. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.1704.01212.
49. Goodfellow, I. J., Mirza, M., Xu, B., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Networks. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.1406.2661.
50. Graikos, A., Malkin, N., Jojic, N., & Samaras, D. (2022). Diffusion models as plug-and-play priors. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2206.09012.
51. Grathwohl, W., Wang, K., Jacobsen, J., Duvenaud, D., & Zemel, R. (2020). Learning the Stein Discrepancy for Training and Evaluating Energy-Based Models without Sampling. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2002.05616.
52. Gu, A., Goel, K., & Ré, C. (2021). Efficiently Modeling Long Sequences with Structured State Spaces. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2111.00396.
53. Gu, S., Chen, D., Bao, J., Wen, F., Zhang, B., Chen, D., Yuan, L., & Guo, B. (2021). Vector Quantized Diffusion Model for Text-to-Image Synthesis. ArXiv. https://doi.org/10.48550/arXiv.2111.14822 Gu, S., Chen, D., Bao, J., Wen, F., Zhang, B., Chen, D., Yuan, L., & Guo, B. (2021). Vector Quantized Diffusion Model for Text-to-Image Synthesis. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2111.14822.
54. Gui, J., Sun, Z., Wen, Y., Tao, D., & Ye, J. (2020). A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2001.06937.
55. Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Inductive Representation Learning on Large Graphs. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.1706.02216.
56. Hamilton, W. L., Ying, R., & Leskovec, J. (2017). Representation Learning on Graphs: Methods and Applications. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.1709.05584.
57. Han, S., Hu, X., Huang, H., Jiang, M., & Zhao, Y. (2022). ADBench: Anomaly Detection Benchmark. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2206.09426.
58. Gong, S., Li, M., Feng, J., Wu, Z., & Kong, L. (2022). DiffuSeq: Sequence to Sequence Text Generation with Diffusion Models. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2210.08933.

59.]Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2006.11239
60. Ho, J., Saharia, C., Chan, W., Fleet, D. J., Norouzi, M., & Salimans, T. (2021). Cascaded Diffusion Models for High Fidelity Image Generation. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2106.15282.
61. Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., & Fleet, D. J. (2022). Video Diffusion Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2204.03458.
62. Hoogeboom, E., Satorras, V. G., Vignac, C., & Welling, M. (2022). Equivariant Diffusion for Molecule Generation in 3D. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2203.17003.
63. Hoogeboom, E., Gritsenko, A. A., Bastings, J., Poole, B., Berg, R. V., & Salimans, T. (2021). Autoregressive Diffusion Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2110.02037.
64. Huang, C., Aghajohari, M., Bose, A. J., Panangaden, P., & Courville,A. (2022). Riemannian Diffusion Models.[Vol-1]ArXiv. https://doi.org/10.48550/arXiv.2208.07949.
65. Huang, C., Lim, J. H., & Courville, A. (2021). A Variational Perspective on Diffusion-Based Generative Models and Score Matching. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2106.02808.
66. Huang, R., Zhao, Z., Liu, H., Liu, J., Cui, C., & Ren, Y. (2022). ProDiff: Progressive Fast Diffusion Model For High-Quality Text-to-Speech. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2207.06389.
67. Aapo Hyvärinen. 2005. Estimation of Non-Normalized Statistical Models by Score Matching. J. Mach. Learn. Res. 6 (12/1/2005), 695–709.
68. Isola, P., Zhu, J., Zhou, T., & Efros, A. A. (2016). Image-to-Image Translation with Conditional Adversarial Networks. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.1611.07004.
69. Jin, W., Barzilay, R., & Jaakkola, T. (2018). Junction Tree Variational Autoencoder for Molecular Graph Generation. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.1802.04364.
70. Jo, J., Lee, S., & Hwang, S. J. (2022). Score-based Generative Modeling of Graphs via the System of Stochastic Differential Equations. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2202.02514.
71. Li, K., Kachman, T., & Mitliagkas, I. (2021). Gotta Go Fast When Generating Data with Score-Based Models. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2105.14080.
72. Combes, R. T., & Mitliagkas, I. (2020). Adversarial score matching and improved sampling for image generation. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2009.05475.
73. Jumper, J., Evans, R., Pritzel, A. et al. Highly accurate protein structure prediction with AlphaFold. Nature 596, 583–589 (2021). https://doi.org/10.1038/s41586-021-03819-2.
74. Karras, T., Aittala, M., Aila, T., & Laine, S. (2022). Elucidating the Design Space of Diffusion-Based Generative Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2206.00364.

75. Kawar, B., Zada, S., Lang, O., Tov, O., Chang, H., Dekel, T., Mosseri, I., & Irani, M. (2022). Imagic: Text-Based Real Image Editing with Diffusion Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2210.09276.
76. Kim, S., Kim, H., & Yoon, S. (2022). Guided-TTS 2: A Diffusion Model for High-quality Adaptive Text-to-Speech with Untranscribed Data. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2205.15370.
77. Kingma, D. P., Salimans, T., Poole, B., & Ho, J. (2021). Variational Diffusion Models. [Vol-5] ArXiv. https://doi.org/10.48550/arXiv.2107.00630.
78. Kingma, D. P., & Welling, M. (2019). An Introduction to Variational Autoencoders. [Vol-3] ArXiv. https://doi.org/10.1561/2200000056.
79. Kong, Z., Ping, W., Huang, J., Zhao, K., & Catanzaro, B. (2020). DiffWave: A Versatile Diffusion Model for Audio Synthesis. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2009.09761.
80. Uria, B., Côté, M., Gregor, K., Murray, I., & Larochelle, H. (2016). Neural Autoregressive Distribution Estimation. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.1605.02226.
81. Levkovitch, A., Nachmani, E., & Wolf, L. (2022). Zero-Shot Voice Conditioning for Denoising Diffusion TTS Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2206.02246.
82. Li, H., Yang, Y., Chang, M., Feng, H., Xu, Z., Li, Q., & Chen, Y. (2021). SRDiff: Single Image Super-Resolution with Diffusion Probabilistic Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2104.14951.
83. Li, J., Tang, T., Zhao, W. X., & Wen, J. (2021). Pretrained Language Models for Text Generation: A Survey. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2105.10311.
84. Li, X. L., Thickstun, J., Gulrajani, I., Liang, P., & Hashimoto, T. B. (2022). Diffusion-LM Improves Controllable Text Generation. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2205.14217.
85. Liu, L., Ren, Y., Lin, Z., & Zhao, Z. (2022). Pseudo Numerical Methods for Diffusion Models on Manifolds. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2202.09778.
86. Lou, A., Lim, D., Katsman, I., Huang, L., Jiang, Q., Lim, S., & De Sa, C. (2020). Neural Manifold Ordinary Differential Equations. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2006.10254.
87. Lu, C., Zheng, K., Bao, F., Chen, J., Li, C., & Zhu, J. (2022). Maximum Likelihood Training for Score-Based Diffusion ODEs by High-Order Denoising Score Matching. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2206.08265.
88. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., & Zhu, J. (2022). DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2206.00927.
89. Lugmayr, A., Danelljan, M., Romero, A., Yu, F., Timofte, R., & Van Gool, L. (2022). RePaint: Inpainting using Denoising Diffusion Probabilistic Models. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2201.09865.
90. Luhman, E., & Luhman, T. (2021). Knowledge Distillation in Iterative Generative Models for Improved Sampling Speed. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2101.02388.

91. Luo, S., & Hu, W. (2021). Diffusion Probabilistic Models for 3D Point Cloud Generation. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2103.01458.
92. Luo, S., & Hu, W. (2021). Score-Based Point Cloud Denoising (Learning Gradient Fields for Point Cloud Denoising). [Vol-4] ArXiv. https://doi.org/10.1109/ICCV48922.2021.00454.
93. Luo, S., Shi, C., Xu, M., & Tang, J. (2021). Predicting molecular conformation via dynamic graph score matching. Advances in Neural Information Processing Systems, 34, 19784-19795.
94. Shitong Luo, Yufeng Su, Xingang Peng, Sheng Wang, Jian Peng, Jianzhu Ma (2022). Antigen-Specific Antibody Design and Optimization with Diffusion-Based Generative Models for Protein Structures **doi:** https://doi.org/10.1101/2022.07.10.499510.
95. Luo, Y., Cai, X., Zhang, Y., & Xu, J. (2018). Multivariate time series imputation with generative adversarial networks. Advances in neural information processing systems, 31.
96. Lyu, Z., XU, X., Yang, C., Lin, D., & Dai, B. (2022). Accelerating Diffusion Models via Early Stop of the Diffusion Process. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2205.12524.
97. Mathieu, E., & Nickel, M. (2020). Riemannian Continuous Normalizing Flows. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2006.10605.
98. Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J., & Ermon, S. (2021). SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2108.01073.
99. Nichol, A., & Dhariwal, P. (2021). Improved Denoising Diffusion Probabilistic Models. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2102.09672.
100. Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., & Chen, M. (2021). GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2112.10741.
101. Nie, W., Guo, B., Huang, Y., Xiao, C., Vahdat, A., & Anandkumar, A. (2022). Diffusion Models for Adversarial Purification. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2205.07460.
102. Nijkamp, E., Hill, M., Han, T., Zhu, S., & Wu, Y. N. (2019). On the Anatomy of MCMC-Based Maximum Likelihood Learning of Energy-Based Models.[Vol-4] ArXiv. https://doi.org/10.48550/arXiv.1903.12370.
103. Nijkamp, E., Hill, M., Zhu, S., & Wu, Y. N. (2019). Learning Non-Convergent Non-Persistent Short-Run MCMC Toward Energy-Based Model. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.1904.09770.
104. Niu, C., Song, Y., Song, J., Zhao, S., Grover, A., & Ermon, S. (2020). Permutation Invariant Graph Generation via Score-Based Generative Modeling. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2003.00638.
105. Oreshkin, B. N., Carpov, D., Chapados, N., & Bengio, Y. (2019). N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.1905.10437.
106. Oreshkin, B. N., Carpov, D., Chapados, N., & Bengio, Y. (2019). N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.1905.10437.

107. Park, S. W., Lee, K., & Kwon, J. (2022). Neural markov controlled SDE: Stochastic optimization for continuous-time data. In International Conference on Learning Representations.
108. Peng, C., Guo, P., Zhou, S. K., Patel, V., & Chellappa, R. (2022). Towards performant and reliable undersampled MR reconstruction via diffusion model sampling. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2203.04292.
109. Popov, V., Vovk, I., Gogoryan, V., Sadekova, T., & Kudinov, M. (2021). Grad-TTS: A Diffusion Probabilistic Model for Text-to-Speech. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2105.06337.
110. Qiu, Y., Zhang, L., & Wang, X. (2020). Unbiased contrastive divergence algorithm for training energy-based latent variable models. In International Conference on Learning Representations.
111. L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," in Proceedings of the IEEE, vol. 77, no. 2, pp. 257-286, Feb. 1989, doi: 10.1109/5.18626.
112. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., & Chen, M. (2022). Hierarchical Text-Conditional Image Generation with CLIP Latents. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2204.06125.
113. Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., & Sutskever, I. (2021). Zero-Shot Text-to-Image Generation. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2102.12092.
114. Rasul, K., Seward, C., Schuster, I., & Vollgraf, R. (2021). Autoregressive Denoising Diffusion Models for Multivariate Probabilistic Time Series Forecasting. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2101.12072.
115. Rasul, K., Sheikh, A., Schuster, I., Bergmann, U., & Vollgraf, R. (2020). Multivariate Probabilistic Time Series Forecasting via Conditioned Normalizing Flows. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2002.06103.
116. Ratliff, L. J., Burden, S. A., & Sastry, S. S. (2014). On the Characterization of Local Nash Equilibria in Continuous Games. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.1411.2168.
117. Rezende, D. J., Mohamed, S., & Wierstra, D. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.1401.4082.
118. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2021). High-Resolution Image Synthesis with Latent Diffusion Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2112.10752.
119. Saharia, C., Chan, W., Chang, H., Lee, C. A., Ho, J., Salimans, T., Fleet, D. J., & Norouzi, M. (2021). Palette: Image-to-Image Diffusion Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2111.05826.
120. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., Salimans, T., Ho, J., Fleet, D. J., & Norouzi, M. (2022). Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2205.11487.
121. Tim Salimans and Jonathan Ho. 2021. Should EBMs model the energy or the score?. In Energy Based Models Workshop-International Conference on Learning Representations.

122. Salinas, D., Callot, L., Medico, R., & Gasthaus, J. (2019). High-Dimensional Multivariate Forecasting with Low-Rank Gaussian Copula Processes. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.1910.03002.
123. Salinas, D., Flunkert, V., & Gasthaus, J. (2017). DeepAR: Probabilistic Forecasting with Autoregressive Recurrent Networks. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.1704.04110.
124. Shi, C., Luo, S., Xu, M., & Tang, J. (2021). Learning Gradient Fields for Molecular Conformation Generation. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2105.03902.
125. Shi, C., Xu, M., Zhu, Z., Zhang, W., Zhang, M., & Tang, J. (2020). GraphAF: A Flow-based Autoregressive Model for Molecular Graph Generation. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2001.09382.
126. Shi, Y., De Bortoli, V., Deligiannidis, G., & Doucet, A. (2022). Conditional Simulation Using Diffusion Schrödinger Bridges. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2202.13460.
127. John Skilling. 1989. The eigenvalues of mega-dimensional matrices. In Maximum Entropy and Bayesian Methods. Springer, 455–466. https://doi.org/10.1007/978-94-015-7860-8_48.
128. Weiss, E. A., Maheswaranathan, N., & Ganguli, S. (2015). Deep Unsupervised Learning using Nonequilibrium Thermodynamics. [Vol-8] ArXiv. https://doi.org/10.48550/arXiv.1503.03585.
129. Song, J., Meng, C., & Ermon, S. (2020). Denoising Diffusion Implicit Models. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2010.02502.
130. Song, K. (2022). Applying Regularized Schrodinger-Bridge-Based Stochastic Process in Generative Modeling. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2208.07131.
131. Song, Y., Durkan, C., Murray, I., & Ermon, S. (2021). Maximum Likelihood Training of Score-Based Diffusion Models. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2101.09258.
132. Song, Y., & Ermon, S. (2019). Generative Modeling by Estimating Gradients of the Data Distribution. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.1907.05600.
133. Song, Y., & Ermon, S. (2020). Improved Techniques for Training Score-Based Generative Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2006.09011.
134. Song, Y., Shen, L., Xing, L., & Ermon, S. (2021). Solving Inverse Problems in Medical Imaging with Score-Based Generative Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2111.08005.
135. Song, Y., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-Based Generative Modeling through Stochastic Differential Equations. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2011.13456.
136. Tae, J., Kim, H., & Kim, T. (2021). EdiTTS: Score-based Editing for Controllable Text-to-Speech. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2110.02584.
137. Tashiro, Y., Song, J., Song, Y., & Ermon, S. (2021). CSDI: Conditional Score-based Diffusion Models for Probabilistic Time Series Imputation. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2107.03502.
138. Vahdat, A., Kreis, K., & Kautz, J. (2021). Score-based Generative Modeling in Latent Space. ArXiv. https://doi.org/10.48550/arXiv.2106.05931.

139. Valevski, D., Kalman, M., Matias, Y., & Leviathan, Y. (2022). UniTune: Text-Driven Image Editing by Fine Tuning an Image Generation Model on a Single Image. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2210.09477.
140. P. Vincent, "A Connection Between Score Matching and Denoising Autoencoders," in Neural Computation, vol. 23, no. 7, pp. 1661-1674, July 2011, doi: 10.1162/NECO_a_00142.
141. Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P. A. (2008, July). Extracting and composing robust features with denoising autoencoders. In Proceedings of the 25th international conference on Machine learning (pp. 1096-1103).
142. Watson, D., Chan, W., Ho, J., & Norouzi, M. (2022). Learning Fast Samplers for Diffusion Models by Differentiating Through Sample Quality. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2202.05830.
143. Watson, D., Ho, J., Norouzi, M., & Chan, W. (2021). Learning to Efficiently Sample from Diffusion Probabilistic Models. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2106.03802.
144. Wu, H., Köhler, J., & Noé, F. (2020). Stochastic Normalizing Flows. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2002.06707.
145. Wu, S., & Shi, Z. (2021). ItôTTS and ItôWave: Linear Stochastic Differential Equation Is All You Need For Audio Generation. [Vol-5] ArXiv. https://doi.org/10.48550/arXiv.2105.07583.
146. Wu, S., Sun, F., Zhang, W., Xie, X., & Cui, B. (2020). Graph Neural Networks in Recommender Systems: A Survey. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2011.02260.
147. J. Wyatt, A. Leach, S. M. Schmon and C. G. Willcocks, "AnoDDPM: Anomaly Detection with Denoising Diffusion Probabilistic Models using Simplex Noise," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), New Orleans, LA, USA, 2022, pp. 649-655, doi: 10.1109/CVPRW56347.2022.00080.
148. Xiao, Z., Kreis, K., & Vahdat, A. (2021). Tackling the Generative Learning Trilemma with Denoising Diffusion GANs. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2112.07804.
149. Xie, T., Fu, X., Ganea, O., Barzilay, R., & Jaakkola, T. (2021). Crystal Diffusion Variational Autoencoder for Periodic Material Generation. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2110.06197.
150. Xie, Y., & Li, Q. (2022). Measurement-conditioned Denoising Diffusion Probabilistic Model for Under-sampled Medical Image Reconstruction. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2203.03623.
151. Kevin Yang and Dan Klein. 2021. FUDGE: Controlled Text Generation With Future Discriminators. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3511–3535, Online. Association for Computational Linguistics.
152. Yang, L., & Hong, S. (2022). Unsupervised Time-Series Representation Learning with Iterative Bilinear Temporal-Spectral Fusion. [Vol-3]ArXiv. https://doi.org/10.48550/arXiv.2202.04770.
153. Yang, R., Srivastava, P., & Mandt, S. (2022). Diffusion Probabilistic Modeling for Video Generation. [Vol-5] ArXiv. https://doi.org/10.48550/arXiv.2203.09481.

154. Yoon, J., Hwang, S. J., & Lee, J. (2021). Adversarial purification with Score-based generative models. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2106.06041.
155. Yoon, J., Jarrett, D., & Van der Schaar, M. (2019). Time-series generative adversarial networks. Advances in neural information processing systems, 32.
156. Yu, S., Tack, J., Mo, S., Kim, H., Kim, J., Ha, J., & Shin, J. (2022). Generating Videos with Dynamics-aware Implicit Generative Adversarial Networks. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2202.10571.
157. Zhang, Q., & Chen, Y. (2021). Diffusion Normalizing Flow. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2110.07579.
158. Zhang, Q., & Chen, Y. (2022). Fast Sampling of Diffusion Models with Exponential Integrator. [Vol-4] ArXiv. https://doi.org/10.48550/arXiv.2204.13902.
159. Zhang, Q., Tao, M., & Chen, Y. (2022). GDDIM: Generalized denoising diffusion implicit models. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2206.05564.
160. Zhao, J., Mathieu, M., & LeCun, Y. (2016). Energy-based Generative Adversarial Network.[Vol-4] ArXiv. https://doi.org/10.48550/arXiv.1609.03126.
161. Zhou, L., Du, Y., & Wu, J. (2021). 3D Shape Generation and Completion through Point-Voxel Diffusion. ArXiv. https://doi.org/10.48550/arXiv.2104.03670.
162. Anderson, B. D. (1982). Reverse-time diffusion equation models. Stochastic Processes and their Applications, 12(3), 313-326. https://doi.org/10.1016/0304-4149(82)90051-5.
163. Kreis, K., Dockhorn, T., Li, Z., & Zhong, E. (2022). Latent Space Diffusion Models of Cryo-EM Structures. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2211.14169.
164. Zhong, E.D., Bepler, T., Berger, B. et al. CryoDRGN: reconstruction of heterogeneous cryo-EM structures using neural networks. Nat Methods 18, 176–185 (2021). https://doi.org/10.1038/s41592-020-01049-4.
165. Jin Sub Lee, View ORCID ProfilePhilip M. Kim (2022). ProteinSGM: Score-based generative modeling for de novo protein design doi: https://doi.org/10.1101/2022.07.13.499967.
166. John Ingraham, Max Baranov, Zak Costello, Vincent Frappier, Ahmed Ismail, Shan Tie, Wujie Wang, Vincent Xue, Fritz Obermeyer, Andrew Beam, Gevorg Grigoryan (2022). Illuminating protein space with a programmable generative model. doi: https://doi.org/10.1101/2022.12.01.518682.
167. Igashov, I., Stärk, H., Vignac, C., Satorras, V. G., Frossard, P., Welling, M., Bronstein, M., & Correia, B. (2022). Equivariant 3D-Conditional Diffusion Models for Molecular Linker Design. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2210.05274.
168. Yu, S., Sohn, K., Kim, S., & Shin, J. (2023). Video Probabilistic Diffusion Models in Projected Latent Space. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2302.07685.
169. Li, X., Lian, L., Liu, Y., Yang, H., Dong, Z., Kang, D., Zhang, S., & Keutzer, K. (2023). Q-Diffusion: Quantizing Diffusion Models. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2302.04304.
170. Tan, H., Wu, S., & Pi, J. (2023). Semantic Diffusion Network for Semantic Segmentation. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2302.02057.

171. Sahak, H., Watson, D., Saharia, C., & Fleet, D. (2023). Denoising Diffusion Probabilistic Models for Robust Image Super-Resolution in the Wild. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2302.07864.
172. Xu, M., Yu, L., Song, Y., Shi, C., Ermon, S., & Tang, J. (2022). GeoDiff: A Geometric Diffusion Model for Molecular Conformation Generation. [Vol-1] ArXiv. https://doi.org/10.48550/arXiv.2203.02923.
173. Jing, B., Corso, G., Chang, J., Barzilay, R., & Jaakkola, T. (2022). Torsional Diffusion for Molecular Conformer Generation. [Vol-2] ArXiv. https://doi.org/10.48550/arXiv.2206.01729.
174. Wang, Z., Zheng, H., He, P., Chen, W., & Zhou, M. (2022). Diffusion-GAN: Training GANs with Diffusion. [Vol-3] ArXiv. https://doi.org/10.48550/arXiv.2206.02262.
175. Rezende, D. J., & Mohamed, S. (2015). Variational Inference with Normalizing Flows. [Vol-6] ArXiv. https://doi.org/10.48550/arXiv.1505.05770.