# A Survey on Generative Diffusion Model

Hanqun Cao, Cheng Tan, Zhangyang Gao, Guangyong Chen, Pheng-Ann Heng, *Senior Member, IEEE*, and Stan Z. Li, *Fellow, IEEE*

**Abstract**—Deep learning shows excellent potential in generation tasks thanks to deep latent representation. Generative models are classes of models that can generate observations randomly with respect to certain implied parameters. Recently, the diffusion Model has become a rising class of generative models by virtue of its power-generating ability. Nowadays, great achievements have been reached. More applications except for computer vision, speech generation, bioinformatics, and natural language processing are to be explored in this field. However, the diffusion model has its genuine drawback of a slow generation process, single data types, low likelihood, and the inability for dimension reduction. They are leading to many enhanced works. This survey makes a summary of the field of the diffusion model. We first state the main problem with two landmark works – DDPM and DSM, and a unified landmark work – Score SDE. Then, we present classified improved techniques for existing problems in the diffusion-based model field. For model speed-up improvement, we present a diverse range of advanced techniques to speed up the diffusion models – training schedule, training-free sampling, mixed-modeling, and score & diffusion unification. For data structure diversification, we present improved techniques for applying diffusion models in continuous space, discrete space, and constraint space. For likelihood optimization, we present theoretical methods for improving ELBO and minimizing the variational gap. For dimension reduction, we present several techniques to solve the high dimension problem. Regarding existing models, we also provide a benchmark of FID score, IS, and NLL according to specific NFE. Moreover, applications with diffusion models are introduced including computer vision, sequence modeling, audio, and AI for science. Finally, there is a summarization of this field together with limitations & further directions. Summation of existing well-classified methods is in our Github: https://github.com/chq1155/A-Survey-on-Generative-Diffusion-Model.

**Index Terms**—Diffusion Model, advanced improvement on diffusion, diffusion application.

## 1 Introduction

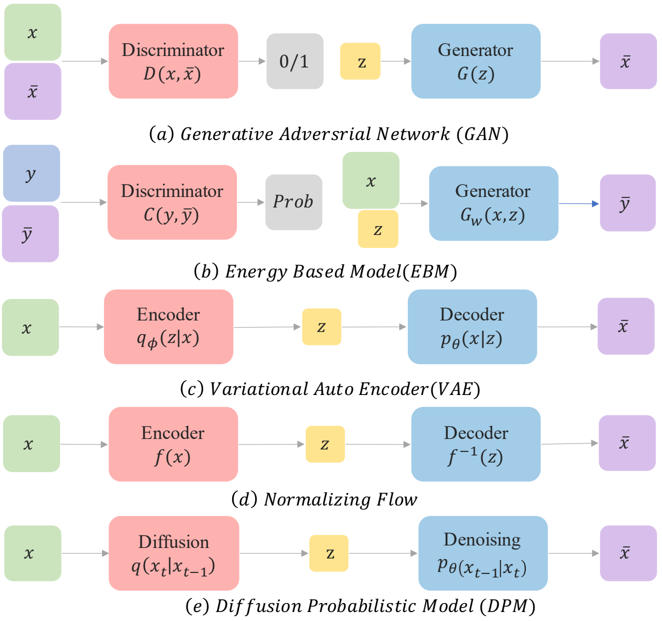
How can we empower machines with human-like imagination? Deep generative models, e.g., VAE [1], [2], [3], [4], EBM [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], GAN [22], [23], [24], normalizing flow [25], [26], [27], [28], [29], [30] and diffusion models [31], [32], [33], [34], [35] , have shown great potential in creating new patterns that humans cannot properly distinguish. We focus on diffusion-based generative models, which do not require aligning posterior distributions as VAE, dealing with intractable partition functions as EBM, training additional discriminators as GAN, or imposing network constraints as normalizing flow. Thanks to the aforementioned virtues, diffusion-based methods have drawn considerable attention from computer vision, and natural language processing to graph analysis. However, there is still a lack of systematic taxonomy and analysis of research progress on diffusion models.

Advances in the diffusion model have provided tractable probabilistic parameterization for describing the model, a stable training procedure with sufficient theoretical support, and a unified loss function design with high simplicity. The diffusion model aims to transform the prior data distribution into random noise before revising the transformations step by step so as to rebuild a brand new sample with the same distribution as the prior [36]. In recent years, the diffusion model has displayed its exquisite potential in the field of computer vision (CV) [31], [37], bioinformatics [38], [39], and speech processing [40], [41]. For instance, denoising diffusion GANs generated high-resolution fake images with just four sampling steps beats GAN [42]. Luo *et al.* [33] firstly generated antibody CDR sequences and structures at the atomic resolution by using DDPMs on protein features. Wavegrad [43] generated high fidelity audio samples with constant steps of generations, outperforming existing GAN-based audio generative models. Inspired by the so-far successes of the diffusion model in CV, bioinformatics, and speech processing domains, applying diffusion models to generation-related tasks of the other domains would be a favorable path for exploiting powerful generative capacity.

On the other hand, the diffusion model has the inherent drawback of plenty of sampling steps and a long sampling time compared to Generative Adversarial Networks (GANs) and Variational Auto-Encoders (VAEs). It is because the diffusion step using the Markov kernel only requires tiny perturbation, which leads to a large number of diffusion. Meanwhile, the tractable model requires the same number of steps during inference. Thus, it takes thousands of steps to sample from a random noise until it eventually alters to high-quality data similar to the prior. Furthermore,

- H. Cao is with the Department of Math, The Chinese University of Hong Kong, Hong Kong, China, and also with the AI Lab, School of Engineering, Westlake University, Hangzhou, China, and Zhejiang Lab, Hangzhou, China. Email: 1155141481@link.cuhk.edu.hk.
- C. Tan and Z. Gao are with Zhejiang University, Hangzhou, China, and aslo with the AI Lab, School of Engineering, Westlake University, Hangzhou, China. Email: tancheng, gaozhangyang@westlake.edu.cn.
- G. Chen is with Zhejiang Lab, Hangzhou, China. Email: gychen@zhejianglab.com.
- P.-A. Heng is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, China.
- Stan Z. Li is with the AI Lab, School of Engineering, Westlake University, Hangzhou, China. Email: Stan.ZQ.Li@westlake.edu.cn.
- H. Cao, C. Tan, and Z. Gao contributed equally to this work.

*Manuscript received April 19, 2005; revised August 26, 2015.*



(a) Generative Adversrial Network (GAN)

(b) Energy Based Model(EBM)

(c) Variational Auto Encoder(VAE)

(d) Normalizing Flow

(e) Diffusion Probabilistic Model (DPM)

Fig. 1. In this figure, we provide an intuitive mechanism for each class of generative models. **(a)** Generative Adversarial Net (GAN) [44] applies adversarial training strategy onto the generator so that it can generate samples that cannot be distinguished by the real-fake discriminator together with the prior. **(b)** Energy-Based Model (EBM) [45] trains in a similar way that it finds a suitable energy function that consists of a softmax discriminator and a prior-input generator such that it can output the best matching sample for random input. **(c)** Variational Auto-Encoder (VAE) [46] applies the encoder to project the prior into a latent space from which the decoder can sample. **(d)** Normalizing flow (NF) [29] employs a well-designed reversible flow function for turning input into latent variable before returning to samples with the inverse of flow function. **(e)** Diffusion model gradually injects noise into original data until it turns to the known noise distribution before reversing each step in the sampling steps.

other problems such as likelihood optimization and the inability of dimension reduction also count. Therefore, lots of works aspired to accelerate the diffusion process along with improving sampling quality [47], [48], [49]. For example, DPM-solver takes the advantage of ODE's stability to generate samples of the State-of-the-art within 10 steps [50]. D3PM [51] not only proposes hybrid training loss but also processes text & categorical data. We summarize improvement works on diffusion models into four classes. (1) Speed-up improvement, (2) Data structure diversification, (3) Likelihood optimization, and (4) Dimension reduction. The detailed content is provided in Section 3.

Hence, based on the wide range of applications along with multi-perspective thinking on algorithm improvement, we target at providing a detailed survey about current aspects of diffusion models. By classifying enhanced algorithms and applications in other domains, the core contributions of this review are as follows:

- Summarize essence mathematical formulation and derivation of fundamental algorithms in the field of diffusion model, including method formulation, training strategy, and sampling algorithm.
- Present comprehensive and up-to-date classification of improved diffusion algorithms and divide them into four categories: speed-up improvement, data structure diversification, likelihood optimization, and dimension reduction.
- Provide extensive statements about the application of diffusion models on computer vision, natural language processing, bioinformatics, and speech processing which include domain-specialized problem formulation, related datasets, evaluation metrics, and downstream tasks, along with sets of benchmarks.
- Clarify current limitations of models and possible further-proof directions concerning the field of diffusion models.

# 2 Problem Statement

## 2.1 Notions and Definitions

### 2.1.1 State

States are a set of data distributions that describe the whole process of diffusion models. In the beginning, the noise is gradually injected into the starting distribution, which is called starting state $x_0$. With enough steps of noise injection, the distribution finally comes into a known noise distribution (mostly Gaussian), which is called the prior state $x_T$ (Discrete)/$x_1$(Continuous). Then, the other distributions between the starting state and the prior state are called intermediate states $x_t$.

### 2.1.2 Process & Transition Kernel

As mentioned above, the process that transforms the starting state into the tractable noise is defined as the forward/diffusion process $F$. The process following the opposite direction to the forward process is called reverse/denoised process $R$. The reverse process samples the noise gradients step by step into the samples as the starting state. In either process, the interchange between any two states is achieved by the transition kernel. The most frequently used kernel is the Markov kernel since it ensures the randomness and tractability in the forward process and the reverse process.

**Forward Process & Kernel:** To present a unified framework, the forward process consists of plenty of forward steps which are the forward transition kernels:

$$F(x,\sigma) = F_T(x_{T-1},\sigma_T)\cdots\circ F_t(x_{t-1},\sigma_t)\cdots\circ F_1(x_0,\sigma_1) \quad (1)$$

$$x_t = F_t(x_{t-1},\sigma_t) \quad (2)$$

Different from the discrete case, for any time $0 \le t < s \le 1$, the forward process is defined:

$$F(x,\sigma) = F_{s1}(x_s,\sigma_{s1})F_{0t}\circ F_{ts}(x_t,\sigma_{ts})\circ F_{0t}(x_0,\sigma_{0t}) \quad (3)$$

$$x_s = F_{ts}(x_s,\sigma_{ts}) \quad (4)$$

where $F_t$ is the forward transition kernel at time $t$ with the variables intermediate state $x_{t-1}$ & $x_{ts}$ and the noise scale $\sigma_t$ & $\sigma_{ts}$. The difference between this expression and normalizing flow is the variable noise scale, which controls the randomness of the whole process. When the noise is close to 0, the process will become the normalizing flow which is deterministic.

**Reverse Process & Kernels:** Similarly, the reverse process is defined as:

$$R(x,\sigma) = R_1(x_1,\sigma_1)\cdots\circ R_t(x_t,\sigma_t)\cdots\circ R_T(x_T,\sigma_T) \tag{5}$$

$$x_{t-1} = R_t(x_t,\sigma_t) \qquad (Discrete) \tag{6}$$

$$R(x,\sigma) = R_{t0}(x_t,\sigma_{t0})\cdots\circ R_{st}(x_s,\sigma_{st})\cdots\circ R_{1s}(x_T,\sigma_{1s}) \tag{7}$$

$$x_t = R_{st}(x_s,\sigma_{st}) \qquad (Continuous) \tag{8}$$

where $R_t$ is the reverse transition kernel at time $t$ with the variables intermediate state $x_t$ & $x_{st}$ and the noise scale $\sigma_t$ & $\sigma_{st}$.

Usually, the reverse process in practice is implemented by the sampling process, which gradually collects the reverse gradients and reconstructs the samples.

**Whole Process:** Denote the sampled data as $\tilde{x}_0$ the generalized process can be expressed as:

$$\begin{aligned}\tilde{x}_0 = [&R_1(x_1,\sigma_1)\cdots\circ R_t(x_t,\sigma_t)\cdots\circ R_T(x_T,\sigma_T)]\circ\\ &[F_T(x_{T-1},\sigma_T)\cdots\circ F_t(x_{t-1},\sigma_t)\cdots\circ F_1(x_0,\sigma_1)]\end{aligned} \tag{9}$$

$$\begin{aligned}\tilde{x}_0 = [&R_{t0}(x_t,\sigma_{t0})\cdots\circ R_{st}(x_s,\sigma_{st})\cdots\circ R_{1s}(x_T,\sigma_{1s})]\circ\\ &[F_{s1}(x_s,\sigma_{s1})F_{0t}\circ F_{ts}(x_t,\sigma_{ts})\circ F_{0t}(x_0,\sigma_{0t})]\end{aligned} \tag{10}$$

### 2.1.3 Discrete or continuous?

There are two main types of diffusion models – discrete process and continuous process. The discrete process assumes the forward & reverse processes consist of T steps. The continuous process assumes the whole process as the time from 0 to 1, where time 0 is the beginning and time 1 is the ending. The main difference between the two kinds of processes is that discrete processes can only obtain integer time steps without reaching the information of non-integer states. The continuous process takes the virtue of real-time information, which makes the continuous process obtains better performances. Besides, for either kind of process, there exist similar distribution states and transition kernels.

### 2.1.4 Training Objective

The diffusion model as one type of the generative model follows the same training objective as variational autoregressive-encoder and normalizing flow, which is keeping starting distribution $x_0$ and sample distribution $\tilde{x}_0$ as close as possible. This is implemented by maximizing the log-likelihood [36]:

$$\mathbb{E}_{F(x_0,\sigma)}\left[-\log R\left(\mathbf{x}_T,\tilde{\sigma}\right)\right] \tag{11}$$

where the $\tilde{\sigma}$ in the reverse process differs from the one in the forward process.

TABLE 1
Notions in Diffusion Systems

| Notations | Descriptions |
|---|---|
| t | Discrete/Continuous time t |
| $z_t$ | Random noise with normal distribution |
| $\epsilon$ | Random noise with normal distribution |
| $\mathcal{N}$ | Normal distribution |
| $\beta_t$ | Variance scale coefficients |
| $\beta(t)$ | Continuous-time $\beta_t$ |
| $\sigma_t$ | Noise scale of perturbation |
| $\sigma(t)$ | Continuous-time $\sigma_t$ |
| $\alpha_t$ | Mean coefficient defined as 1 - $\beta_t$ |
| $\alpha(t)$ | Continuous-time $\alpha_t$ |
| $\bar{\alpha}_t$ | Cumulative product of $\alpha_t$ |
| $\gamma(t)$ | Signal-to-Noise ratio |
| $\eta_t$ | Step size of annealed Langevin dynamics |
| $x$ | Unperturbed data distribution |
| $\tilde{x}$ | Perturbed data distribution |
| $x_0$ | Starting distribution of data |
| $x_t$ | Diffused data at time t |
| $x_t^{'}$ | Partly diffused data at time t |
| $x_T$ | Random noise after diffusion |
| $F(x,\sigma)$ | Forward/Diffusion process |
| $R(x,\sigma)$ | Reverse/Denoised process |
| $F_t(x_t,\sigma_t)$ | Forward/Diffusion step at time t |
| $R_t(x_t,\sigma_t)$ | Reverse/Denoised step at time t |
| $F_{ts}(x_t,\sigma_{ts})$ | Forward/Diffusion step at time t |
| $R_{st}(x_s,\sigma_{st})$ | Reverse/Denoised step at time t |
| $q(x_t\|x_{t-1})$ | DDPM forward step at time t |
| $p(x_{t-1}\|x_t)$ | DDPM reverse step at time t |
| $f(x,t)$ | Drift coefficient of SDE |
| $g(t)$ | Simplified diffusion coefficient of SDE |
| $\mathcal{D}(x,t)$ | Degrader at time t in Cold Diffusion |
| $\mathcal{R}(x,t)$ | Reconstructor at time t in Cold Diffusion |
| $w,\bar{w}$ | Standard Wiener process |
| $\nabla_x \log p_t(x)$ | Score function w.r.t $x$ |
| $\mu_\theta(x_t,t)$ | Mean coefficient of reversed step |
| $\Sigma_\theta(x_t,t)$ | Variance coefficient of reversed step |
| $\epsilon_\theta(x_t,t)$ | Noise prediction model |
| $s_\theta(x)$ | Score network model |
| $L_0, L_{t-1}, L_T$ | Forward loss, reversed loss, decoder loss |
| $L_{vlb}$ | Evidence Lower Bound |
| $L_{simple}$ | Simplified denoised diffusion loss |
| $\theta$ | learnable parameters |
| $\phi$ | learnable parameters |

## 2.2 Problem Formulation

Based on the unified framework, two discrete landmark works – DDPM [52] and Denoising Score Matching (DSM) [53], along with the unified continuous landmark works – Score SDE are stated below with customed transition kernels and training objectives.

### 2.2.1 Diffusion Model Formulation

The original idea of the diffusion probabilistic model is to recreate a specific distribution that starts with random noise. Thus, the distributions of generated samples are required to be as close as the ones of original samples.

**DDPM Forward Process:** Based on the unified framework, DDPM chooses a sequence of noise coefficients $\beta_1, \beta_2, ..., \beta_T$ for Markov transition kernels following specific patterns. The common choices are constant schedule, linear schedule, and cosine schedule. According to [52], different schedules of noise have no clear effects in experiments. The DDPM forward step & process are defined as:

$$\begin{aligned} F_t(x_{t-1}, \beta_t) &:= q(x_t|x_{t-1}) \\ &:= \mathcal{N}\left(x_t, \sqrt{1-\beta_t}x_{t-1}, \sqrt{\beta_t}\mathbf{I}\right) \end{aligned} \tag{12}$$

By a sequence of diffusion steps from $x_0$ to $x_T$, We have the Forward/Diffusion Process:

$$\begin{aligned} F(x_0, \beta) &:= q\left(x_{1:T} \mid x_0\right) \\ &:= \prod_{t=1}^{T} q\left(x_t \mid x_{t-1}\right) \end{aligned} \tag{13}$$

**DDPM Reverse Process:** Given the forward process above, we define the Reverse step as the inverse step with respect to learned Gaussian transitions parameterized by $\theta$ [52]:

$$\begin{aligned} R_t(x_t, \Sigma_\theta) &:= p_\theta(x_{t-1}|x_t) \\ &:= \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)\right) \end{aligned} \tag{14}$$

By a sequence of reverse step from $x_T$ to $x_0$, we have the Reverse Process starting at $p(x_T) = \mathcal{N}(x_T; \mathbf{0}, \mathbf{I})$:

$$\begin{aligned} R(x_T, \Sigma_\theta) &:= p_\theta\left(x_{0:T}\right) \\ &:= p\left(x_T\right) \prod_{t=1}^{T} p_\theta\left(x_{t-1} \mid x_t\right) \end{aligned} \tag{15}$$

Consequently, the distribution $p_\theta(x_0) = \int p_\theta(x_{0:T})dx_{1:T}$ should be the distribution of $\tilde{x}_0$.

**Diffusion Training Objective:** By minimizing the negative log-likelihood (NLL), the minimization problem can be formulated as:

$$\begin{aligned} \mathbb{E}\left[-\log p_\theta\left(\mathrm{x}_0\right)\right] \leq \mathbb{E}_q & \left[-\log \frac{p_\theta\left(\mathrm{x}_{0:T}\right)}{q\left(\mathrm{x}_{1:T} \mid \mathrm{x}_0\right)}\right] \\ &= \mathbb{E}_q\left[-\log p\left(\mathrm{x}_T\right) - \sum_{t \geq 1} \log \frac{p_\theta\left(\mathrm{x}_{t-1} \mid \mathrm{x}_t\right)}{q\left(\mathrm{x}_t \mid \mathrm{x}_{t-1}\right)}\right] \\ &= \mathbb{E}_q[\underbrace{D_{\mathrm{KL}}\left(q\left(x_T \mid x_0\right) \| p\left(x_T\right)\right)}_{L_T} \\ &+ \sum_{t>1} \underbrace{D_{\mathrm{KL}}\left(q\left(x_{t-1} \mid x_t, x_0\right) \| p_\theta\left(x_{t-1} \mid x_t\right)\right)}_{L_{t-1}} \\ &\underbrace{-\log p_\theta\left(x_0 \mid x_1\right)]}_{L_0} \\ &=: L \end{aligned} \tag{16}$$

Here we use the symbol of Ho *et al.* [52]. Denote $L_T$ as the forward loss, which represents the divergence between the forwarding process and the distribution of random noise, which is a constant depending on variance schedule $\beta_1, \cdots \beta_T$; Denote $L_0$ as the decode loss; Besides, denote $L_{1:T-1}$ as the reverse loss, which is the sum of divergence between posterior of forwarding step and reverses step at each step.

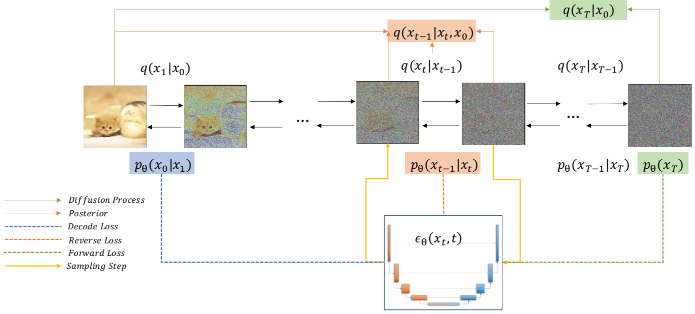


Fig. 2. Pipeline of Denoised Diffusion Probabilistic Model. The arrows pointing from left to right indicate the diffusion process and the arrows pointing in the reverse direction indicate the reverse process. The colored background transition terms are components of ELBO: the blue part stands for decode loss $L_0$, the green part represents forward loss $L_T$, and the orange part constitutes the reverse loss $L_t$. Dashed lines with different colors show the training pattern of the noise prediction model $\epsilon_\theta$. Besides, in any step $1 \leq t \leq T$, the yellow lines denote the ancestral sampling process.

### 2.2.2 Score Matching Formulation

The score matching model aims at solving the original data distribution estimation problem by approximating the gradient of data $\nabla_x log p(x)$, which is called score. The main approach of score matching is to train a score network $x_\theta$ to predict the score [54], [55], which is obtained by means of perturbing data with different noise schedules. The score matching process is defined as:

**Score Perturbation Process & Kernel:** The perturbation process consists of a sequence of perturbation steps with increasing noise scales $\sigma_1, ..., \sigma_N$. The Gaussian perturbation kernel is defined as $q_\sigma(\tilde{x}|x) := \mathcal{N}\left(\tilde{x} \mid x, \sigma^2 I\right)$. For each noise scale $\sigma_i$, the score is equivalent to the gradient of the perturbation kernel. If we treat this increasing noise perturbation as a discrete process, the transition kernel between two neighbor states is

$$x_i = x_{i-1} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2}\epsilon, \quad i = 1, \cdots, N \tag{17}$$

where $N$ is the length of the noise scale sequence, and $\epsilon$ is random noise.

**Score Matching Process:** As noticed above, the goal of the score matching process is to obtain a score estimation network $s_\theta(x, \sigma)$ to be as close as possible to the gradient of perturbation kernel, which is

$$L := \frac{1}{2}\mathbb{E}\left[\left\|s_\theta\left(x, \sigma\right) - \nabla \log q\left(x\right)\right\|^2\right] \tag{18}$$

where $\theta$ is the learnable parameters in the score network.

**DDPM & DSM Connection:** To some extent, score matching and denoising diffusion are the same kinds of processes. (1) Denoising mechanism: both DSM and DDPM follow the pattern of fetching information during the noising process and reusing gradient during the denoising process. Both processes transform prior distribution to known noise and finally reverse back to the original distribution. Moreover, noising schedule of DSM can be seen as an accumulation of constant-variance diffusion steps. (2) Training object: Both DSM and DDPM aim at maximizing the prior likelihood and they train the network for gradient prediction. (3) Sampling method: both DSM and DDPM apply the idea of ancestral sampling, reconstructing the samples by collecting related gradients step by step.

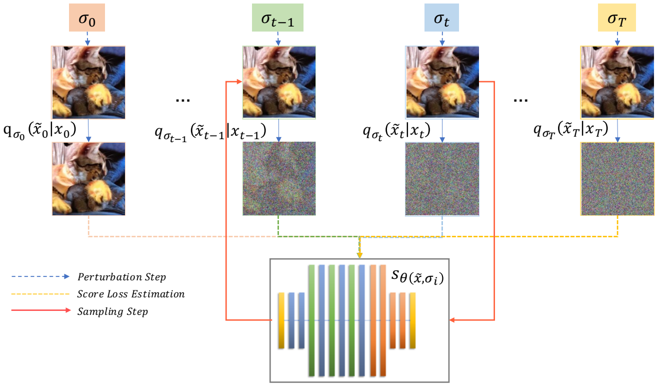

$\sigma_0$
$\sigma_{t-1}$
$\sigma_t$
$\sigma_T$
...
...
$q_{\sigma_0}(\tilde{x}_0|x_0)$
$q_{\sigma_{t-1}}(\tilde{x}_{t-1}|x_{t-1})$
$q_{\sigma_t}(\tilde{x}_t|x_t)$
$q_{\sigma_T}(\tilde{x}_T|x_T)$
$s_{\theta(\tilde{x},\sigma_i)}$
Perturbation Step
Score Loss Estimation
Sampling Step

Fig. 3. Pipeline of Denoised Score Matching (DSM). The $\sigma$'s in different time states on the top represent alternative scales of noise. The transition states $p_{\sigma_t}(\tilde{x}_t|x_t)$ are the output gradients of the perturbation. Dashed lines with different colors reveal that the scoring network $s_\theta$ is trained by minimizing the sum of L2-loss between the output gradient and the score in each noise scale. Besides, in any noise state $1 \leq t \leq T$, the red lines denote the Langevin Dynamics sampling process.

### 2.2.3 Score SDE Formulation

Score SDE [56] proposed a unified continuous framework based on the stochastic differential equation to describe diffusion models and denoised score matching models. It not only presents the corresponding continuous set-up of DDPM of DSM based on score SDE but also proposes a density estimation ODE framework named probability flow ODE.

**Forward Score SDE Process:** In Song *et al.* [56], Diffusion process can be viewed as a continuous case described by Stochastic Differential Equation. And it is equal to the solution to Itô SDE [57], which is composed of a drift part for mean transformation and a diffusion coefficient for noise description. :

$$dx = f(x,t)dt + g(t)dw, t \in [0,T] \tag{19}$$

where $w$ is the standard Wiener process/Brownian Motion, $f(\cdot,t)$ is the drift coefficient of $x(t)$, and $g(\cdot)$ is the simplified version of diffusion coefficient of $x(t)$, which is assumed not dependent on $x$. Besides, $p_0$, $p_t(x)$ denote the data distribution and probability density of $x(t)$. $p_T$ denotes the original prior distribution which gains no information from $p_0$. When the coefficients are piece-wise continuous, the forward SDE equation admits a unique solution [58].

Similar to the discrete case, the forward transition in the SDE framework is derived as:

$$\begin{aligned} F_{st}(x(s), g_{st}) &:= q\left(x_t \mid x_s\right) \\ &:= \mathcal{N}\left(x_t \mid f_{ts}x_s, g_{ts}^2 I\right), 0 < s < t \leq 1 \\ R_{ts}(x(t), g_{ts}) &:= q\left(x_s \mid x_t, x_0\right) \\ &= \mathcal{N}\left(x_s \mid \frac{1}{g_{t0}^2}\left(f_{s0}g_{ts}^2 x_0 + f_{ts}g_{s0}^2 x_t\right), \frac{g_{s0}^2 g_{ts}^2}{g_{t0}^2}\boldsymbol{I}\right) \end{aligned} \tag{20}$$

where $f_{ts} = \frac{f(x,t)}{f(x,s)}$ and $g_{ts} = \sqrt{g(t)^2 - f_{ts}^2 g(s)^2}$.

**Reversed Score SDE Process:** In contrast to the Forward SDE Process, the Reversed SDE Process is defined with respect to the reverse-time Stochastic Differential Equation by running backward in time [56]:

$$\mathrm{d}x = \left[f(x,t) - g(t)^2 \nabla_x \log p_t(x)\right]\mathrm{d}t + g(t)\mathrm{d}\overline{w}, t \in [0,T] \tag{21}$$

Furthermore, $\nabla_x \log p_t(x)$ is the score to be matched [59].

**Score SDE Training Objective:** The training objective of score SDE employs weighting scheme in the score loss compared to denoised score matching, which is

$$\begin{aligned} L := \mathbb{E}_t\{\lambda(t)\mathbb{E}_{x(0)}\mathbb{E}_{x(t)x(0)} \\ [\|s_{\boldsymbol{\theta}}(x(t),t) - \nabla_{x(t)} \log p(x(t)x(0))\|_2^2]\} \end{aligned} \tag{22}$$

where $x(t), x(0)$ are corresponding continuous time variables of $x_t, x_0$.

**SDE-based DDPM & DSM:** Based on the SDE frameworks, the transition kernel of DDPM and DSM can be expressed as :

$$\mathrm{d}x = -\frac{1}{2}\beta(t)x\ dt + \sqrt{\beta(t)}\mathrm{d}w \tag{23}$$

$$\mathrm{d}x = \sqrt{\frac{\mathrm{d}\left[\sigma^2(t)\right]}{\mathrm{d}t}}\,\mathrm{d}W \tag{24}$$

where $\beta(t)$ and $\sigma(t)$ are the continuous-time variable of discrete noise scales $\beta_t$ and $\sigma_i$. Moreover, the two kinds of SDE are called Variation Preserving (VP) and Variation Explosion (VE) SDE respectively.

**Probability Flow ODE:** Probability Flow ODE (Diffusion ODE) [56] is the continuous-time ODE that supports the deterministic process which shares the same marginal probability density with SDE. Inspired by Maoutsa *et al.* [60] and Chen *et al.* [61], any type of diffusion process can be derived into a special form of ODE. In the case that functions $G$ is independent of $x$, the probability flow ODE is

$$dx = \{f(x,t) - \frac{1}{2}G(t)G(t)^T \nabla_x logp_t(x)\}dt \tag{25}$$

In contrast to SDE, probability flow ODE can be solved with larger step sizes as they have no randomness. Due to the advantages of ODE, several works such as PNDMs [62] and DPM-Solver [50] obtain amazing results by modeling the diffusion problem as an ODE.

## 2.3 Training Strategy

### 2.3.1 Denoising Diffusion Training Strategy

In order to minimize the negative log-likelihood, the only item we can be used to train is $L_{1:T-1}$. By parameterizing the posterior $q(x_{t-1}|x_t, x_0)$ using Baye's rule, we have:

$$q\left(x_{t-1} \mid x_t, x_0\right) = \mathcal{N}\left(x_{t-1}; \tilde{\boldsymbol{\mu}}_t\left(x_t, x_0\right), \tilde{\beta}_t I\right) \tag{26}$$

where $\alpha_t$ is defined as $1 - \beta_t$, $\bar{\alpha}_t$ is defined as $\prod_{k=1}^{t} \alpha_k$. Mean and variance schedules can be expressed as:

$$\begin{aligned} \tilde{\boldsymbol{\mu}}_t\left(\mathrm{x}_t, \mathrm{x}_0\right) &:= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathrm{x}_0 + \frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\mathrm{x}_t \\ \tilde{\beta}_t &:= \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \end{aligned} \tag{27}$$

Keeping above parameterization as well as reparameterizing $x_t$ as $x_t(x_0, \sigma)$, $L_{t-1}$ can be regarded as an expectation of L2-loss between two mean coefficients:

$$L_{t-1} = \mathbb{E}_q\left[\frac{1}{2\sigma_t^2}\left\|\tilde{\boldsymbol{\mu}}_t\left(\mathrm{x}_t, \mathrm{x}_0\right) - \boldsymbol{\mu}_\theta\left(\mathrm{x}_t, t\right)\right\|^2\right] + C \tag{28}$$

Simplifying $L_{t-1}$ by reparameterizing $\mu_\theta$ w.r.t $\epsilon_\theta$, we obtain the simplified training objective named $L_{simple}$:

$$L_{simple} := \mathbb{E}_{x_0,\epsilon}\left[\frac{{\beta_t}^2}{2{\sigma_T}^2\alpha_t(1-\bar{\alpha}_t)}\right]\left\|\epsilon - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon)\right\|^2 \tag{29}$$

Most diffusion models until now use the training strategy of DDPMs. But there exist some exceptions. DDIM's training objective can be transformed by adding a constant from DDPM's although it is independent of Markovian step assumption; Training pattern of Improved DDPM named as $L_{hybrid}$ is to combine training object of DDPM $L_{simple}$ and a term with variational lower bound $L_{vlb}$. However, $L_simple$ still takes the main effect of these training methods.

### 2.3.2 Score Matching Training Strategy

Traditional score matching techniques requires massive computation cost for Hessian of log density function. To fix this problem, advanced methods find approaches to avoid Hessian computing. Implicit score matching (ISM) [59] treat the real score density as a non-normalized density function that can be optimized by neural network. Sliced score matching (SSM) [63] provide a unperturbed score estimation method through reverse-mode auto-differentiation by projecting score onto random vectors.

$$L := \mathbb{E}\left[\frac{1}{2}\left\|\boldsymbol{s}_\theta\left(x\right)\right\|_\Lambda^2 + \nabla\left(\boldsymbol{s}_\theta\right)\right] \tag{30}$$

$$L := \mathbb{E}_{p_{\mathrm{v}}}\mathbb{E}_{p_{\text{data}}}\left[v^\top\nabla_x s_{\boldsymbol{\theta}}(x)v + \frac{1}{2}\left\|s_{\boldsymbol{\theta}}(\mathrm{x})\right\|_2^2\right] \tag{31}$$

However, because of the low-manifold problem in real data as well as the sampling problem in the low-density region, denoised score matching could be the better solution for improving score matching. Denoised score matching (DSM) [54] transforms the original score matching into a perturbation kernel learning by perturbing a sequence of increasing noise.

$$L := \frac{1}{2}\mathbb{E}_{q_\sigma((\tilde{x}|x)p_{data}(x))}\left[\| \, s_\theta(\tilde{x}) - \nabla_{\tilde{x}} log q_\sigma(\tilde{x}|x) \, \|_2^2\right] \tag{32}$$

According to Song *et al.*, the noise distribution is defined to be $q_\sigma(\tilde{x}|x) = \mathcal{N}(\tilde{x}|x, \sigma^2 I)$. Thus, for each given $\sigma$, the specific expression denoising score matching objective is

$$L(\boldsymbol{\theta}; \sigma) := \frac{1}{2}\mathbb{E}_{p_{\text{data}}(\mathrm{x})}\mathbb{E}_{\tilde{\mathrm{x}}\sim\mathcal{N}(\mathrm{x},\sigma^2 I)}\left[\left\|\mathrm{s}_{\boldsymbol{\theta}}(\tilde{\mathrm{x}}, \sigma) + \frac{\tilde{\mathrm{x}} - \mathrm{x}}{\sigma^2}\right\|_2^2\right] \tag{33}$$

## 2.4 Sampling Algorithm

Reconstructing data distribution needs sampling. In each sampling step, the sample generated from random noise will be refined again to get closer to the original distribution. In this subsection, we present basic sampling algorithms for the three landmark works.

### 2.4.1 Ancestral Sampling

The initial idea of ancestral sampling [64] is reconstructed with the gradient of inverse Markovian step by step. Thus, ancestral sampling for DDPM is:

**Algorithm 1** Ancestral Sampling

```
$x_T \sim \mathcal{N}(0, I)$
for $t = T, ..., 1$ do
  if $t > 1$ then
    $z \sim \mathcal{N}(0, I)$
  else
    $F = 0$
  end if
  $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta\left(x_t, t\right)\right) + \sigma_t z$
end for
return $x_0$
```

### 2.4.2 Langevin Dynamics Sampling

Langevin dynamics can produce samples from a probability density $p(x)$ with only the score function (Song *et al.*) $\nabla_x log p(x)$. With a fixed step size $\epsilon > 0$, the recursive algorithm is:

**Algorithm 2** Annealed Langevin Dynamics Sampling

```
Initialize $x_0$
for $i = 1, ..., L$ do
  $\alpha_i \leftarrow \epsilon \cdot \sigma_i^2/\sigma_L^2$
  for $i = 1, ..., L$ do
    $Z \sim \mathcal{N}(0, I)$
    $\tilde{\mathrm{x}}_t = \tilde{\mathrm{x}}_{t-1} + \frac{\alpha_i}{2}\,\mathrm{s}_{\boldsymbol{\theta}}\left(\tilde{\mathrm{x}}_{t-1}, \sigma_i\right) + \sqrt{\alpha_i}\mathrm{z}_t$
  end for
  $\tilde{x}_0 \leftarrow \tilde{x}_T$
end for
return $\tilde{x}_T$
```

### 2.4.3 Predictor-corrector (PC) Sampling

PC sampling [65] is inspired by a type of ODE black-box ODE solver [66], [67], [68] in order to produce high-quality samples and trade-off accuracy for efficiency for all reversed SDE. The sampling procedure is made up of a predictor sampler and a corrector sampler. For solving VE SDE and VP SDE, Song *et al.*, [56] used reverse diffusion SDE solver as the predictor, and annealed Langevin dynamics as the corrector.

**Algorithm 3** Predictor-Corrector Sampling(VE SDE)

```
$x_N \sim \mathcal{N}\left(0, \sigma_{\max}^2 I\right)$
for $i = N - 1$ to 0 do
    $x_i' \leftarrow x_{i+1} + \left(\sigma_{i+1}^2 - \sigma_i^2\right) s_{\boldsymbol{\theta}} * (x_{i+1}, \sigma_{i+1})$
    $Z \sim \mathcal{N}(0, I)$
    $x_i \leftarrow x_i' + \sqrt{\sigma_{i+1}^2 - \sigma_i^2} z$
    for $j = 1$ to $M$ do
        $Z \sim \mathcal{N}(0, I)$
        $x_i \leftarrow x_i + \epsilon_i s_{\boldsymbol{\theta}} * (x_i, \sigma_i) + \sqrt{2\epsilon_i} z$
    end for
end for
return $x_0$
```

**Algorithm 4** Predictor-Corrector Sampling(VP SDE)

```
$x_N \sim \mathcal{N}(0, I)$
for $i = N - 1$ to 0 do
    $x_i' \leftarrow \left(2 - \sqrt{1 - \beta_{i+1}}\right) x_{i+1} + \beta_{i+1} s_{\boldsymbol{\theta}} * (x_{i+1}, i + 1)$
    $z \sim \mathcal{N}(0, I)$
    $x_i \leftarrow x_i' + \sqrt{\beta_{i+1}} z$
    for $j = 1$ to $M$ do
        $Z \sim \mathcal{N}(0, I)$
        $x_i \leftarrow x_i + \epsilon_i s_{\boldsymbol{\theta}} * (x_i, \sigma_i) + \sqrt{2\epsilon_i} z$
    end for
end for
return $x_0$
```

## 2.5 Evaluation Metric

In order to evaluate the properties of generated samples, Evaluation metrics are designed to test the sample quality and diversity.

### 2.5.1 Inception Score (IS)

The inception score is built with the goal of valuing both the diversity and resolution of generated images based on ImageNet dataset [69], [70]. It can be divided into two parts: diversity measurement and quality measurement. Diversity measurement denoted by $p_{IS}$ is calculated w.r.t. the class entropy of generated samples: the larger the entropy is, the more diverse the samples will be. Quality measurement denoted by $q_{IS}$ is computed through the similarity between a sample and the related class images using entropy. It is because the samples will enjoy high resolution if they are closer to the specific class of images in the ImageNet dataset. Thus, to lower $q_{IS}$ and higher $p_{IS}$, the KL divergence [71] is applied to inception score calculation:

$$\begin{aligned} IS &= D_{KL}(p_{IS} \parallel q_{IS}) \\ &= \mathbb{E}_{x \sim p_{IS}}\left[\log \frac{p_{IS}}{q_{IS}}\right] \\ &= \mathbb{E}_{x \sim p_{IS}}[\log(p_{IS}) - \log(p_{IS})] \end{aligned} \tag{34}$$

### 2.5.2 Frechet Inception Distance (FID)

Although there are reasonable evaluation techniques in the Inception Score, the establishment is based on a specific dataset with 1000 classes as well as a trained network that consists of randomness such as initial weights, and code framework [72]. Thus, the bias between ImageNet and real-world images may cause an inaccurate outcome. Furthermore, the number of sample batches is much less than 1000 classes, leading to low-belief statistics.

FID is proposed to solve the bias from specific reference datasets. The score shows the distance between real-world data distribution and the generated samples using the mean and the covariance [73].

$$\mathrm{FID} = \left\|\mu_r - \mu_g\right\|^2 + \operatorname{Tr}\left(\Sigma_r + \Sigma_g - 2\left(\Sigma_r \Sigma_g\right)^{1/2}\right) \tag{35}$$

where $\mu_g, \Sigma_g$ are the mean and covariance of generated samples, and $\mu_r, \Sigma_r$ are the mean and covariance of real-world data.

### 2.5.3 Negative Log Likelihood (NLL)

According to Razavi *et al.*, [74] negative log-likelihood is seen as a common evaluation metric that describes all modes of data distribution. Lots of works on normalizing flow field [75], [76], [77], [78], [79] and VAE field [80], [81], [82] uses NLL as one of the choices for evaluation. Some diffusion models like improved DDPM [47] regard the NLL as the training objective.

$$\mathrm{NLL} = \mathbb{E}\left[-\log p_\theta(x)\right] \tag{36}$$

# 3 Algorithm Improvement

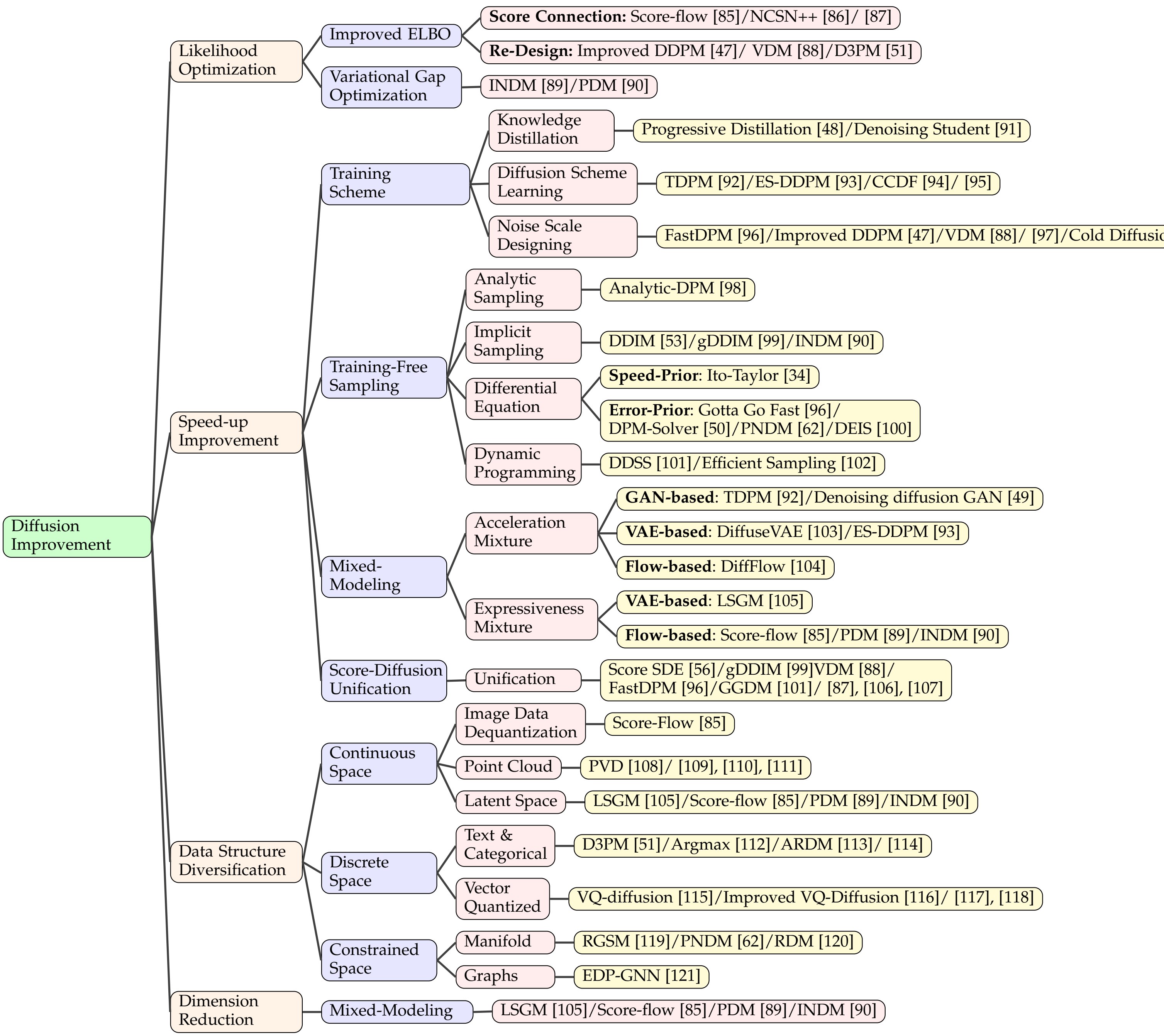
Nowadays, the main concern of the diffusion model is to speed up its speed and reduce the cost of computing. With the aid of strong conditioned settings, sampling of diffusion can be achieved in just few steps [48], such as text-to-speech [83], and image super-resolution [84]. In general cases, it takes thousands of steps for diffusion models to generate a high-quality sample. Mainly focusing on improving sampling speed, many works from different aspects come into reality. Besides, other problems such as variational gap optimization, data structure diversification, and dimension reduction are also attracting extensive research interest. In this section, we divide the improved algorithm w.r.t. problems to be solved. For each problem, we present the significance and related detailed classification of solutions (which are shown in Table 2).

## 3.1 Speed-up Improvement

### 3.1.1 Training Schedule

The improvement in the training schedule means changing traditional ways of training such as diffusion schemes, noise

TABLE 2
Classification of Improved Techniques


Diffusion Improvement
Likelihood Optimization
Improved ELBO
Score Connection: Score-flow [85]/NCSN++ [86]/ [87]
Re-Design: Improved DDPM [47]/ VDM [88]/D3PM [51]
Variational Gap Optimization
INDM [89]/PDM [90]
Speed-up Improvement
Training Scheme
Knowledge Distillation
Progressive Distillation [48]/Denoising Student [91]
Diffusion Scheme Learning
TDPM [92]/ES-DDPM [93]/CCDF [94]/ [95]
Noise Scale Designing
FastDPM [96]/Improved DDPM [47]/VDM [88]/ [97]/Cold Diffusio
Training-Free Sampling
Analytic Sampling
Analytic-DPM [98]
Implicit Sampling
DDIM [53]/gDDIM [99]/INDM [90]
Differential Equation
Speed-Prior: Ito-Taylor [34]
Error-Prior: Gotta Go Fast [96]/ DPM-Solver [50]/PNDM [62]/DEIS [100]
Dynamic Programming
DDSS [101]/Efficient Sampling [102]
Mixed-Modeling
Acceleration Mixture
GAN-based: TDPM [92]/Denoising diffusion GAN [49]
VAE-based: DiffuseVAE [103]/ES-DDPM [93]
Flow-based: DiffFlow [104]
Expressiveness Mixture
VAE-based: LSGM [105]
Flow-based: Score-flow [85]/PDM [89]/INDM [90]
Score-Diffusion Unification
Unification
Score SDE [56]/gDDIM [99]VDM [88]/ FastDPM [96]/GGDM [101]/ [87], [106], [107]
Data Structure Diversification
Continuous Space
Image Data Dequantization
Score-Flow [85]
Point Cloud
PVD [108]/ [109], [110], [111]
Latent Space
LSGM [105]/Score-flow [85]/PDM [89]/INDM [90]
Discrete Space
Text & Categorical
D3PM [51]/Argmax [112]/ARDM [113]/ [114]
Vector Quantized
VQ-diffusion [115]/Improved VQ-Diffusion [116]/ [117], [118]
Constrained Space
Manifold
RGSM [119]/PNDM [62]/RDM [120]
Graphs
EDP-GNN [121]
Dimension Reduction
Mixed-Modeling
LSGM [105]/Score-flow [85]/PDM [89]/INDM [90]

schemes, and data distribution schemes, which are independent of sampling. Recent related studies have shown the key factors in training schemes that influence learning patterns and models' performance. Thus, in this sub-section, we divide the training enhancement into four categories: knowledge distillation, diffusion scheme learning, noise scale designing, and data distribution replacement.

**Knowledge Distillation** Knowledge distillation is an emerging method for obtaining efficient small-scale networks by transferring ”knowledge” from complex teacher models with high learning capacity to simple student models [122]. Thus, student models equip the advantages in model compression and model acceleration.

Salimans *et al.* [48] firstly utilize the core idea into diffusion model improvement by progressively distilling knowledge from one sampling model to another. In each distillation step, student models re-weight from teacher models before being trained to generate one-step samples as close as teacher models do. As a result, student models can halve their sampling steps during each distillation process. Following the same training objective as DDPMs with alternative parameterization methods, the Progressive Distillation model achieved an FID of 2.57 in only 4 steps. This core idea was used in ProDiff [123]in the domain of Text-to-Speech generation. The sampling quality of ProDiff is around the SOTA with only 2 sampling steps, compared with the former SOTA DiffSpeech which needs 128 steps for sampling. Further improvement along the path of distillation includes denoising students, in which the teacher models directly guide the sampling pattern of the student model by minimizing the L2-loss [124] of samples from both models. Denoising students got one-step sampling with an FID score of 9.36, compared with 9.12 for Progressive Distillation one-step generation.

**Diffusion Scheme Learning** Diffusion scheme learning aims at probing the effect of different diffusion patterns on the model's speed. Recent studies focus on the diffusion steps. TDPM [92] has firstly proposed that truncating both the diffusion process and sampling process leading to less sampling time is beneficial for reducing sampling time along with improving generating quality. The key idea of truncating patterns is generating less diffused data with the help of other generative models like GAN and VAE. TDPM used GAN as well as conditional transport [125] to learn the implicit generative distribution from random noise. Unlike TDPM, Early Stop (ES) DDPM [93] generated implicit distribution by means of generating prior data with VAE which learned the latent space from $x_0$. To be more general, CCDF [94] has shown that there exists an optimal step less than $T$ minimizing the estimation error by contraction theory. Besides the Shortcut thinking of reducing steps, Franzese *et al.*, [95] defined the number of training steps as a variable to train the model with more flexibility together with probing the optimal steps.

**Noise Scale Designing** Different from DDPM which defines noise scale as a constant, exploring works concerning the effect of noise scale learning has also drawn much attention since noise schedule learning also counts during diffusion and sampling. Each sampling step can be seen as a random walk on the direct line pointing to the prior distribution, which shows that noise adjustment may benefit the sampling procedure. Improved DDPM [47] first took noise scale tuning into consideration by way of defining adding a $\lambda$−scaled term $L_{vlb}$ on $L_{simple}$. Different from Improved DDPM, San Roman *et al.*, [97] has proposed a noise estimation method containing another noise prediction network $P_\theta$ to update the noise scale in each step before conducting ancestral sampling. Thus, optimizing the loss $L_{hybrid}$ leads to learning the noise schedule. Thankfully, Improved DDPM exceeded DDIM [53] over 50 steps. Furthermore, FastDPM [96] and VDM [88] reparameterized the noise scalar $\alpha_t$ and $\beta_t$ to directly express the loss term with respect to noise scale to optimize the procedure w.r.t noise schedule where the signal-to-noise ratio is defined as $\gamma(t) := {\alpha_t}^2 \; {\sigma_t}^2$ which is a decreasing function utilized as a variable in re-expressing the training objective to simplify and unify the training objective of different types of models.

Except for noise scale designing in the diffusion & denoising process, cold diffusion [94] has analyzed known noise distribution, which can be set as any distribution by cold diffusion improved sampling to eliminate the prediction error by the wrong design of Reconstructor $\mathcal{R}$ (Generalized term of all types of the samplers). The cold diffusion improved sampling is:

**Algorithm 5** Cold Diffusion Sampling

```
Input: A degraded sample $x_t$
  for $s = t, t-1, ..., 1$ do
    $\hat{x}_0 \leftarrow R(x_s, s)$
    $x_{s-1} = x_s - D(\hat{x}_0, s) + D(\hat{x}_0, s-1)$
  end for
  return $x_0$
```

To sum up, noise learning guides the random walk of random noise in both the diffusion and sampling processes, leading to more efficient reconstruction.

### 3.1.2 Training-Free Sampling

Many methods are focusing on changing the pattern of training and noise schedule to improve sampling speed, but re-training models cost more computing and lead to the risk of unstable training. Thankfully, there exists a class of methods that directly enhance the sampling algorithm with a pre-trained model which is called training-free sampling. The goal of advanced training-free sampling is to propose an efficient sampling algorithm for acquiring knowledge from the pre-trained models with fewer steps and higher accuracy. It contains four categories: analytical methods, implicit sampler, differential equation solver sampler, and dynamic programming adjustment.

**Analytical Method** Bao *et al.* firstly propose analytical method named analytic-DPM [98]. It improves DDIM [53] by optimizing the reverse variance by means of KL-Divergence analysis.

**Implicit Sampler** As mentioned above, it usually takes the same number of steps for the generative process as the diffusion process to rebuild the original data distribution in DDPM. However, the diffusion model has the so-called decoupling property that does not require the equivalent number of steps for diffusing and sampling. Inspired by generative implicit model [126], Song *et al.*, [53] has proposed implicit sampling method equipped with deterministic diffusion and jump-step sampling. Surprisingly, implicit models have no requirement for re-training models since the forward diffusion probabilistic density is kept the same for any time $t \in [0, T]$. DDIM solves the jump-step acceleration using continuous process formulation [61], [127] with the aid of neural ODE as:

$$\mathrm{d}\overline{x}(t) = \epsilon_\theta^{(t)}\left(\frac{\overline{x}(t)}{\sqrt{\sigma^2+1}}\right)\mathrm{d}\sigma(t) \tag{37}$$

where $\sigma_t$ is parameterized by $\sqrt{1-\alpha}/\sqrt{\alpha}$, and $\bar{x}$ is parameterized as $x/\sqrt{\alpha}$. Besides, the probability can be treated as one kind of Score SDE, which is derived from the discrete formulation:

$$\frac{x_{t-\Delta t}}{\sqrt{\alpha_{t-\Delta t}}} = \frac{x_t}{\sqrt{\alpha_t}} + \left(\sqrt{\frac{1-\alpha_{t-\Delta t}}{\alpha_{t-\Delta t}}} - \sqrt{\frac{1-\alpha_t}{\alpha_t}}\right)\epsilon_\theta^{(t)}(x_t) \tag{38}$$

Based on DDIM, gDDIM [99] generalized implicit diffusion with diverse types of kernels in the SDE framework including DDPM, DDIM, and critically-damped Langevin diffusion (CLD) [128] into a family of DDIM, achieving the acceleration of implicit diffusion model by corresponding ODE and SDE solvers. Different from above, INDM [90] achieves an implicit mechanism by transforming the nonlinear diffusion process into linear latent diffusion by normalizing flows.

**Differential Equation Solver Sampler** Compared to traditional diffusion methods with discrete steps, numerical formulations of differential equations achieve more efficient sampling with advanced solvers. Inspired by Score SDE and Probability Flow (Diffusion) ODE [56], several works centering on efficient sampling with an advanced differential equation solver by minimizing approximation error using fewer sampling steps. [129], [130], [131], [132] Existing methods can be divided into the ODE-based and the SDE-base. The SDE-based utilize formulation in score SDE along with solvers like Euler-Maruyama (EM) [133], improved Euler, and stochastic Runge-Kutta (RK) [134]method. The ODE-based have a wider range of choices such as Runge-Kutta [135], Forward Euler, and Linear multi-step methods [136]. In general, there exists a trade-off between sampling speed and sampling quality [34]. While higher order differential equation solvers have smaller approximation errors and higher order of convergence, linear solvers require less evaluation. [137] Moreover, ODEs are easier and simpler to solver compared to SDEs. Thus, the choice of differential equations together with related solvers counts leads to another way of classification – Accuracy-prior and Speed-prior on ODE, SDE, and semi-linear DE fields.

For the methods preferred accuracy leveraging higher order solver, Itô-Taylor Sampling Scheme [100] has been proposed using a high-order SDE solver. What is different is that the sampler applied ideal derivative substitution to parameterize the score function in a tricky way that avoids higher-order derivative computing.

There are also methods to improve the whole process by applying both linear solvers and higher-order solvers such as Gotta Go Fast [96] and PNDM [62]. It derived an algorithm based to achieve directional guidance on step size adjustment. In the sampling process, the method combined a linear solver (Euler-Maruyama Method) with a higher-order one (Improved Euler Method) with an extrapolation technique to accelerate with less extra computing. It can reach an FID score of less than 3.00 on CIFAR-10 with 150 steps. PNDM has explored that different numerical solvers can share the same gradient. So it explored the linear multi-step method after using 3 steps of a higher-order solver (Runge-Kutta method) in Diffusion ODE. Besides, DPM-solver [50] also leveraged solvers of different orders. Empirically, DPM Solver-Fast (progress with a mixture of different order solvers) performed the best among all the choices. Therefore, a united solver cross alternative orders may have a better performance after well-design.

Furthermore, from the perspective of differential equation choosing, DPM-solver and DEIS [96] created a new viewpoint except for SDE and Diffusion ODE. They claimed that the diffusion ODE can be seen as a semi-linear form by which the discretization errors are reduced. DPM-solver accomplished the SOTA within 50 steps on CIFAR-10 and it can generate a high-quality image ($\leq$ 7 FID score) with 10 steps, which is an extensive upgrade. On the other hand, DEIS improved numerical DDIM with a multi-step PC-sampling method [138] by the usage of exponential integrator [139]. Furthermore, DEIS-tAB3 achieves the SOTA on CIFAR10 in 5, 10, 15, and 20 steps with the corresponding FID score of 15.37, 4.17, 3.37, and 2.86. Thus, solving the numerical diffusion model by treating it as a semi-linear structure attains the most effective sampling quality.

**Dynamic Programming Adjustment** Dynamic programming achieves the traversal of all choices to find the optimized solution in a very reduced time by memorization trick [140]. Compared to other efficient sampling, methods with dynamic programming locate the optimized sampling route instead of designing powerful steps that minimize the error more quickly. Assume that each path from one state to another shares the same KL-Divergence with others, Watson *et al.*, [102] proposed an efficient sampling method to directly search the optimized route of sampling with the minimum ELBO. This method only requires $O\left(T^2\right)$ for computing and restoration, and it explores a new approach for optimizing the trajectory. However, the minimization on ELBO sometimes has a mismatch with FID scores [141]. Inspired by Kumar *et al.*, [142] differentiable Diffusion Sampler Search (DDSS) [101] utilizes rematerialization trick to trade memory cost for computation time.

### *3.1.3 Mixed-Modeling*

Mixed-Modeling means pulling another type of generative model in the pipeline of the diffusion model to take virtue of the high sampling speed of others such as adversarial training network and auto-regressive encoder and high expressiveness like normalizing flow. Thus, extracting all the strengths by jointly combining two or more models with a specific pattern performs a promising enhancement, which is called Mixed-modeling. Mixed modeling can be classified into two classes from the perspective of mixing purpose: acceleration mixture and expressiveness mixture.

**Acceleration Mixture** Acceleration mixture aims at applying high-speed generation of VAEs and GANs to save plenty of steps on the reconstruction of less perturbed data distribution. Existing GAN-based methods consist of two parts as before: the generator is responsible for generating samples $x_0^{'}$ to be diffused into $x_{t-1}^{'}$ which is as close as $x_{t-1}$, while the discriminator aims at distinguishing $x_{t-1}^{'}$ and $x_{t-1}$ under the condition of $x_t$ along with $q(x_t|x_{t-1})$. Denoised diffusion GAN [49] became the first DDPM-related model that generated samples with 4 steps, and it obtained an FID score of 9.63 on the CIFAR-10 Dataset. Following a similar pattern, VAE-based models like DiffuseVAE [103] and ES-DDPM [93] apply. Since it takes much time on predicting $x_0$ in each sampling step when we use $q(x_{t-1}|x_t, x_0)$, VAEs are used in $x_0$ generation so as to accelerate the whole process, which is what DiffuseVAE has done. Based on DiffuseVAE, ES-DDPM combined the early stop idea in sample trajectory learning and DiffuseVAE to accomplish early stop sampling with the condition generated from diffused VAE samples.

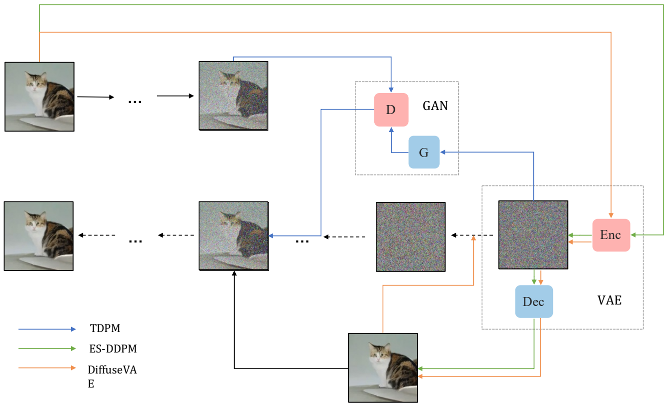


Fig. 4. Acceleration mixed-modeling pipeline. To some extent, this type of model has incomplete diffusion and reverse processes. The blue line represents the pipeline of TDPM. The partly perturbed data $x_t$ is applied as the ground-truth condition for GAN's generator, and the conditional samples $x_t'$ with the same level of perturbation are generated from the latent before being compared with $x_t$. The successful samples are applied as the beginning of the reverse process. Instead of using GAN as the high-speed generator, ES-DDPM follows the pattern of TDPM's with VAE, which is shown with green lines. Besides, DiffuseVAE employs VAE to generate condition $\hat{x}_0$ in each sampling step.

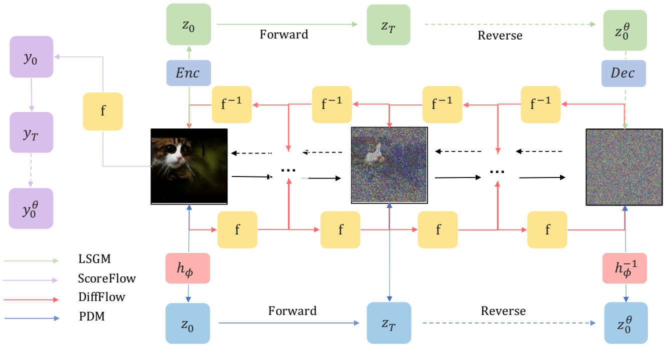


Fig. 5. Expressiveness mixed-modeling pipeline. The models for expressiveness enhancement keep the same procedure as DDPM in the training, diffusion, and sampling process, which is denoted by black lines. Furthermore, additional improvements are highlighted by other colors. The red lines show the pipeline of DiffFlow, which adds flow function and related inverse function at each step. The works in blue lines and green lines represent the idea of latent space diffusion by jointly training mixture models. Different supporting generative models are used in LSGM and PDM. Besides, Score-Flow uses the flow function as a projector from discrete space to dequantization space. Then generating dequantized samples using the traditional diffusion method.

**Expressiveness Mixture** Another category of mixed-modeling called expressiveness mixture support diffusion models on expressing data or noise in a different pattern. As for noise modulation, DiffFlow [104] employs a flow function in each SDE-based diffusion step forward and backward to add noise adaptively and efficiently by minimizing the KL-Divergence between the forward process and backward process. DiffFlow leads to a 20* speedup compared to DDPM, although there is a longer time required per step since the flow functions' back-propagation. An additional set of models leverage NFs into data transformation. Training data based on its latent space (non-continuous data) allows us to learn smoother models in a smaller space, triggering fewer network evaluations and faster sampling [105]. Therefore, both LSGM [105] and PDM [89] obtained latent variables using VAE and flow function respectively. Besides, based on the goal of Maximal Likelihood Estimation (MLE), LSGM jointly trained the diffusion process along with VAE, while PDM jointly trained the SDE of score matching along with the invertible normalizing flow. Besides, Score-Flow [85] employs normalizing flow to transform data distribution into a dequantization field and conduct a diffusion process to generate dequantized samples. Projecting data onto the dequantization field with variational dequantization solves the mismatching between continuous density and discrete data [143], [144], eliminating the gap between dequantization space and discrete space, leading to enhancement of sample quality [145], [146].

### 3.1.4 Score & Diffusion Unification

There are many works on unifying the diffusion models with a generalized one. The acceleration method on generalized diffusion helps a lot in solving a wide range of models along with insights into efficient sampling mechanisms. Also, other related works determine the connection between the diffusion model and denoising score matching, which can be viewed as one type of unification.

For generalized perspective, FastDPM [96] and VDM [88] finished the unification w.r.t. noise schedules. In FastDPM, there is a bijective mapping between continuous-time and noise scale. So, the diffusion and reverse parameterization is accomplished by noise scale defined by $r : r = \mathcal{R}(t)$ and $t = \mathcal{T}(r)$. VDM first proposed the signal-to-noise ratio (SNR) parameterization criteria, revealing the effects of the noise scheme on the diffusion model's training and sampling. Generalized DDIM (gDDIM) [99] unifies the DDIM family with according to the transition kernel during each step.

From the unification perspective, score SDE [56] is the landmark work that combines two types of problems into one along with proposing a unified framework. Gong *et al.*, [107] reveals the hidden connection between score matching with the normalizing flow, and it provides a new approach to expressing score matching by flow ODEs [61], [127]. Bortoli *et al.*, [106] provided a variational score matching approach for simulating diffusion bridges using Doob-h transformation [190].

## 3.2 Data Structure Diversification

Currently, most improvement methods for acceleration and computation cost reduction concentrate on the performance of RGB-image data in order to evaluate the power of generation. Indeed, most kinds of existing data can be the input of diffusion models, leading to various applications in the other fields, such as amino acid residue [187], audio sequence [178], and torsion angle [185]. More importantly, the traditional patterns of diffusion which utilize Gaussian distribution as prior and transition kernel are to be extended to explore the influences of patterns with distinct distributions. However, works on this topic are relatively less. Thus, we divided the distribution diversification into three aspects: discrete space, continuous space, and Constrained space with structural constraints.

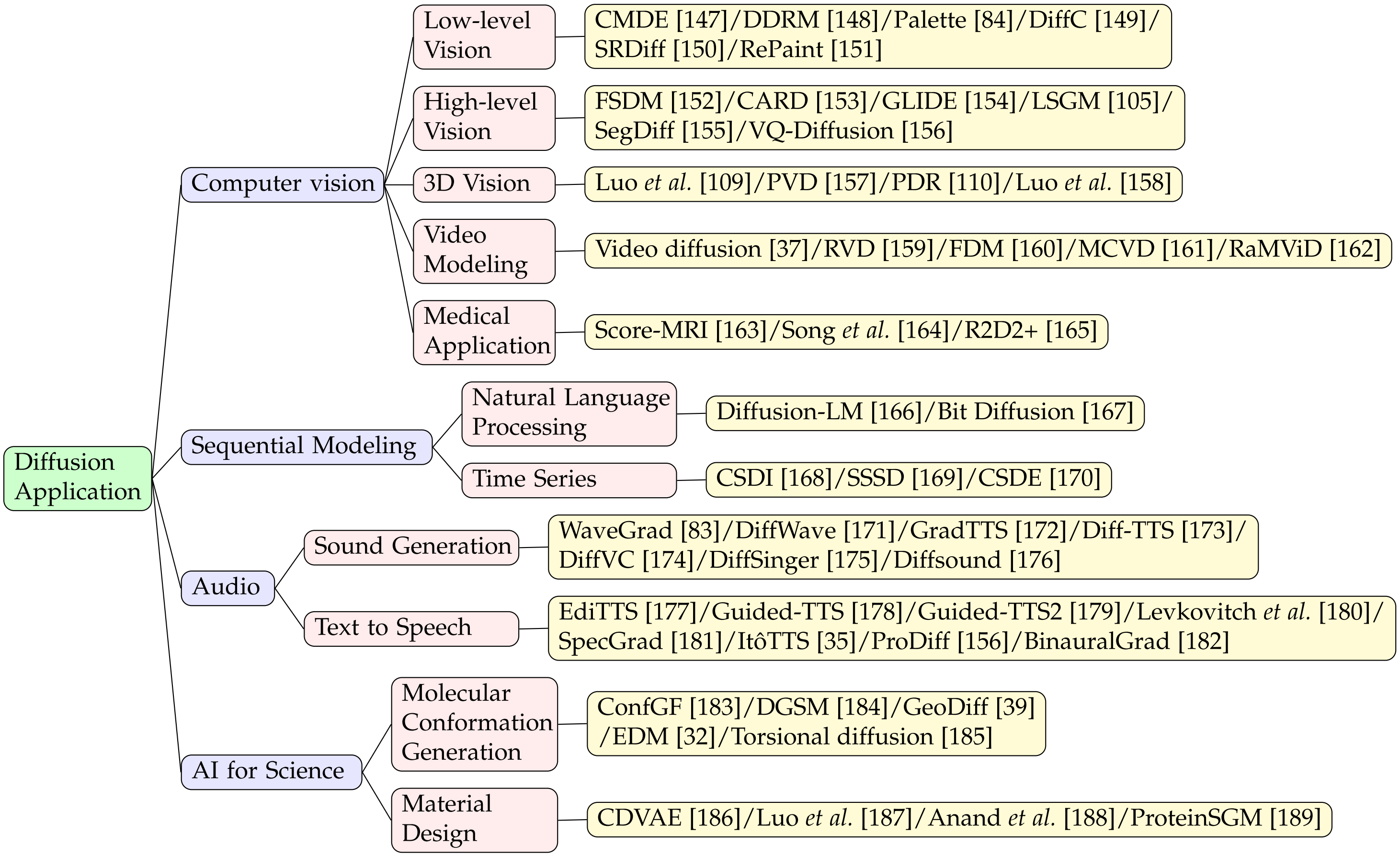

Diffusion Application
Computer vision
Low-level Vision
CMDE [147]/DDRM [148]/Palette [84]/DiffC [149]/ SRDiff [150]/RePaint [151]
High-level Vision
FSDM [152]/CARD [153]/GLIDE [154]/LSGM [105]/ SegDiff [155]/VQ-Diffusion [156]
3D Vision
Luo et al. [109]/PVD [157]/PDR [110]/Luo et al. [158]
Video Modeling
Video diffusion [37]/RVD [159]/FDM [160]/MCVD [161]/RaMViD [162]
Medical Application
Score-MRI [163]/Song et al. [164]/R2D2+ [165]
Sequential Modeling
Natural Language Processing
Diffusion-LM [166]/Bit Diffusion [167]
Time Series
CSDI [168]/SSSD [169]/CSDE [170]
Audio
Sound Generation
WaveGrad [83]/DiffWave [171]/GradTTS [172]/Diff-TTS [173]/ DiffVC [174]/DiffSinger [175]/Diffsound [176]
Text to Speech
EdiTTS [177]/Guided-TTS [178]/Guided-TTS2 [179]/Levkovitch et al. [180]/ SpecGrad [181]/ItôTTS [35]/ProDiff [156]/BinauralGrad [182]
AI for Science
Molecular Conformation Generation
ConfGF [183]/DGSM [184]/GeoDiff [39] /EDM [32]/Torsional diffusion [185]
Material Design
CDVAE [186]/Luo et al. [187]/Anand et al. [188]/ProteinSGM [189]

Fig. 6. Classification of Diffusion-based model Applications

### 3.2.1 Continuous Space

For continuous space methods, two main classes – Dequantization space and point cloud space are introduced. As mentioned above, dequantization space projection solves the mismatching between continuous density and discrete data [143], [144] as well as eliminates the gap between dequantization space and discrete space, leading to enhancement of sample quality [145], [146]. Point cloud data used for 3d shape generation, 3d shape completion, and multi-modal completion attract more attention but meet the barrier using the autoregressive encoder and normalizing flows due to the irregular sampling procedure. Diffusion models applied to continuous space data often combine other projection networks such as VAE and normalizing flow to transform various types of data into specific latent spaces.

**Dequantization & Point Cloud** Score-flow [85] employs flow function to project RGB-image into dequantization space, achieving diffusion techniques on it for generating accurate samples. Point cloud generation is firstly proposed by *Luo et al.* [109], which generates latent samples for point cloud data, and conducts transformation to obtain high-quality 3d shapes. Other techniques such as [108], [110], [111] accomplish the shape generation and completion tasks in similar ways. Some slight improvements used in latent space transformation such as canonical map [111], condition feature extraction sub-nets [110], and point-voxel representation [108].

**Latent Space** Similar to Expressiveness mixture modeling, latent space data distributions are often processed for diffusion application since different types of complex data structures require a unified approach to generalize and analyze. Most current methods project data into continuous space, and they obtain promising performance with the aid of high-quality generation power of diffusion models such as EDM [32] and antigen-diffusion [33]. Thus, latent space processing should be a beneficial pattern utilized in new application fields.

### 3.2.2 Discrete Space

**Text & Categorical** For discrete space methods, they focus on combining different types of data structures such as vector-quantized data, text data, and categorical data into the diffusion model's training and sampling so as to handle a wider range of field tasks such as text, segmentation maps, and categorical features [114].D3PM [51] firstly promoted diffusion algorithm onto discrete space to deal with discrete data like sentences and images by means of defining

$$q\left(x_t \mid x_{t-1}\right) = \mathrm{Cat}\left(x_t; p = x_{t-1}\boldsymbol{Q}_t\right) \tag{39}$$

where $\left[\boldsymbol{Q}_t\right]_{ij} = q\left(x_t = j \mid x_{t-1} = i\right)$ is defined as the transition, and $Cat(\cdot)$ is defined as the categorical distribution over the one-hot row vector.

Besides, in order to process multi-nominal discrete data as well as promote better predictions of the data $x_0$ at each time step, the parameterization and training objectives are:

$$p_\theta\left(x_{t-1} \mid x_t\right) \propto \sum_{\widetilde{x}_0} q\left(x_{t-1}, x_t \mid \widetilde{x}_0\right) \widetilde{p}_\theta\left(\widetilde{x}_0 \mid x_t\right) \tag{40}$$

$$L_\lambda = L_{\mathrm{vb}} + \lambda \mathbb{E}_{q\left(x_0\right)} \mathbb{E}_{q\left(x_t \mid x_0\right)}\left[-\log \widetilde{p}_\theta\left(x_0 \mid x_t\right)\right] \tag{41}$$

Similar to D3PM, multi-nomial diffusion [112] and ARDM [113] extended the categorical diffusion into multinomial data for generating language text & segmentation map and Lossless Compression.

**Vector-Quantized** To handle the multi-model problem, vector-quantized (VQ) data is proposed to combine data from different fields into the codebook. VQ data processing achieved great performance in autoregressive encoders [191]. Gu *et al.* [115] firstly applied diffusion techniques into VQ data, solving unidirectional bias as well as accumulation prediction error problems existing in VQ-VAE. Further works such as Xie *et al.* [118], Cohen *et al.* [117], and Improved VQ-Diffusion [116] accomplished text-to-sign pose generation, diffusion-bridge generation through VQ-VAE pipeline, and inference strategy improvement respectively by re-defining the transition process as:

$$q\left(x_t \mid x_{t-1}\right) = \boldsymbol{v}^\top\left(x_t\right) \boldsymbol{Q}_t \boldsymbol{v}\left(x_{t-1}\right) \tag{42}$$

where $v(t)$ is a one-hot vector of the equal length with the code-book, and $Q(t)$ is called the probability transition matrix:

$$\boldsymbol{Q}_t = \begin{bmatrix} \alpha_t+\beta_t & \beta_t & \beta_t & \cdots & 0 \\ \beta_t & \alpha_t+\beta_t & \beta_t & \cdots & 0 \\ \beta_t & \beta_t & \alpha_t+\beta_t & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma_t & \gamma_t & \gamma_t & \cdots & 1 \end{bmatrix} \tag{43}$$

Training methods are similar to DDPM's but with a new expression scheme.

### 3.2.3 Constrained Space

**Manifold Space** Most current data structures such as images and text are defined in Euclidean space, which is a flat-geometry manifold. However, there exists a series of data in the field of robotics [192], [193], geoscience [194], [195], [196], and protein modelling [197] defined in Riemannian manifold [198], where existing methods for Euclidean space cannot capture the feature of sphere well. Thus, recent methods RDM [120] and RGSM [119] applied diffusion techniques into Riemannian manifold by score SDE framework [56] with a slight change. The revised SDE system is defined by:

$$d\mathbf{X}_t = b\left(\mathbf{X}_t\right) \mathrm{d}t + \mathrm{d}\mathbf{B}_t^{\mathcal{M}} \tag{44}$$

The time-reverse process is defined as:

$$\mathrm{d}\mathbf{Y}_t = \left\{-b\left(\mathbf{Y}_t\right) + \nabla \log p_{T-t}\left(\mathbf{Y}_t\right)\right\} \mathrm{d}t + \mathrm{d}\mathbf{B}_t^{\mathcal{M}} \tag{45}$$

The corresponding sampling algorithm called Geodesic Random Walk (GRW) [199], [200], [201] is implemented:

$$X_{n+1}^\gamma = \exp_{X_n^\gamma}\left[\gamma\left\{b\left(X_n^\gamma\right) + (1/\sqrt{\gamma})\left(V_{n+1} - b\left(X_n^\gamma\right)\right)\right\}\right] \tag{46}$$

Unlike the above two methods, PNDM [62] draws support from Manifold space to solve the neural differential equation for sampling, which indeed is a generalized version for differential equation sampler [202].

**Graph** According to [203], graph is becoming an increasingly popular topic but few works are proposed in the field of diffusion. In EDP-GNN [121], graph data is processed through an adjacency matrix before being applied in the traditional discrete score matching pipeline in order to capture the graph's permutation invariance.

## 3.3 Likelihood Optimization

Most variational methods [191], [204], [205], [206] and diffusion methods [52] train models by principle of variational evidence lower bound (ELBO) since the log-likelihood is not tractable. Although minimizing ELBO increases the lower bound of likelihood, sometimes the log-likelihood is still not competitive because the variational gap between ELBO and log-likelihood is not minimized at the same time. Thus, several methods [47], [85] focus on the likelihood optimization problem directly to solve this problem. The solutions can be classified into two classes – improved ELBO and variational gap optimization.

### 3.3.1 Improved ELBO

Based on the original ELBO, many works try to tighten the lower bound to make log-likelihood more competitive. There are two types of approaches: score connection and re-design.

**Score Connection:** Inspired by [207], [208], [209], score connection methods provide a new connection between ELBO optimization and score matching, solving the likelihood optimization problems via improved score training. Score-flow [85] treats the forward KL divergence in ELBO as optimizing a score matching loss with a weighted scheme. Huang *et al.* [87] treated Brownian motion as a latent variable to explicitly track the log-likelihood estimation, and it builds the bridge between the estimation and weighted score matching in the variational framework. Similarly, NCSN++ [86] bridges the theoretical gap by introducing a truncation factor to ELBO.

**Re-Design:** Compared to loss transformation techniques, re-Design methods directly tighten the ELBO by re-designing noise scale and training objectives. VDM [88] and DDPM++ [86] connect the advanced training objectives with respect to signal-to-noise ratio and truncate factors respectively, optimizing ELBO via finding optimal factors. Improved DDPM [47] and D3PM [51] propose hybrid loss functions based on ELBO with a weighted scheme for improving ELBO.

$$L_{\text{hybrid}} = L_{\text{simple }} + \lambda L_{\mathrm{vlb}} \tag{47}$$

$$L_\lambda = L_{\mathrm{vb}} + \lambda \mathbb{E}_{q\left(x_0\right)} \mathbb{E}_{q\left(x_t \mid x_0\right)}\left[-\log \widetilde{p}_\theta\left(x_0 \mid x_t\right)\right] \tag{48}$$

TABLE 3
Benchmarks on CelebA-64

| Method | NFE | FID | NLL |
|---|---|---|---|
| NCSN [54] | 1000 | 10.23 | - |
| NCSN ++ [86] | 1000 | 1.92 | 1.97 |
| DDPM ++ [86] | 1000 | 1.90 | 2.10 |
| DiffuseVAE [103] | 1000 | 4.76 | - |
| Analytic DPM [98] | 1000 | - | 2.66 |
| ES-DDPM [93] | 200 | 2.55 | - |
| PNDM [62] | 200 | 2.71 | - |
| ES-DDPM [93] | 100 | 3.01 | - |
| PNDM [62] | 100 | 2.81 | - |
| Analytic DPM [98] | 100 | - | 2.66 |
| ES-DDPM [93] | 50 | 3.97 | - |
| PNDM [62] | 50 | 3.34 | - |
| DPM-Solver Discrete [50] | 36 | 2.71 | - |
| ES-DDPM [93] | 20 | 4.90 | - |
| PNDM [62] | 20 | 5.51 | - |
| DPM-Solver Discrete [50] | 20 | 2.82 | - |
| ES-DDPM [93] | 10 | 6.44 | - |
| PNDM [62] | 10 | 7.71 | - |
| Analytic DPM [98] | 10 | - | 2.97 |
| DPM-Solver Discrete [50] | 10 | 6.92 | - |
| ES-DDPM [93] | 5 | 9.15 | - |
| PNDM [62] | 5 | 11.30 | - |

### 3.3.2 Variational Gap Optimization:

Apart from designing advanced ELBO, minimizing the variational gap is still one approach to maximize the log-likelihood. Based on the success of variational gap optimization [210] in the VAE field, INDM [89] applies the flow model to express the variational gap, minimizing the gap by jointly training the bidirectional flow model and linear diffusion model on latent space. Additionally, PDM accomplishes the variational gap expression by introducing encoder loss of VAE. With collective training, there exists a unique optimal solution to eliminate the gap.

### 3.3.3 Dimension Reduction

## 4 Benchmarks

The benchmarks of landmark models along with improved techniques corresponding to FID score, Inception Score, and NLL are provided on diverse datasets which includes CIFAR-10 [211], ImageNet [212], and CelebA-64 [213]. In addition, some dataset-based performances such as LSUN [214], FFHQ [215], and MINST [216] are not presented since there is much less experiment data. The selected performance are listed according to NFE in descending order in order to compare for easier access.

## 5 Application

Benefiting from the powerful ability to generate realistic samples, diffusion models have been widely used in various fields such as computer vision, natural language processing, and bioinformatics.

### 5.1 Computer vision

### 5.1.1 Low-level vision

CMDE [147] empirically compared score-based diffusion methods in modeling conditional distributions of visual image data and introduced a multi-speed diffusion framework. By leveraging the controllable diffusion speed of the condition, CMDE outperformed the vanilla conditional denoising estimator [54] in terms of FID scores in in-painting and super-resolution tasks. DDRM [148] proposed an efficient, unsupervised posterior sampling method served for image restoration. Motivated by variational inference, DDRM demonstrated successful applications in super-resolution, deblurring, inpainting, and colorization of diffusion models. Palette [84] further developed a unified diffusion-based framework for low-level vision tasks such as colorization, inpainting, cropping, and restoration. With its simple and general idea, this work demonstrated the superior performance of diffusion models compared to GAN models. DiffC [149] proposed an unconditional generative approach that encoded and denoise corrupted pixels with a single diffusion model, which showed the potential of diffusion models in lossy image compression. SRDiff [150] exploited the diffusion-based single-image super-resolution model and showed competitive results. RePaint [151] was a free-form inpainting method that directly employed a pre-trained diffusion model as the generative prior and only replaced the reverse diffusion by sampling the unmasked regions using the given image information. Though there was no modification to the vanilla pre-trained diffusion model, this method was able to outperform autoregressive and GAN methods under extreme tasks.

TABLE 4
Benchmarks on ImageNet-64

| Method | NFE | FID | IS | NLL |
|---|---|---|---|---|
| MCG [217] | 1000 | 25.4 | - | - |
| Analytic DPM [98] | 1000 | - | - | 3.61 |
| ES-DDPM [93] | 900 | 2.07 | 55.29 | - |
| Efficient Sampling [102] | 256 | 3.87 | - | - |
| Analytic DPM [98] | 200 | - | - | 3.64 |
| ES-DDPM [93] | 100 | 3.75 | 48.63 | - |
| DPM-Solver Discrete [50] | 57 | 17.47 | - | - |
| ES-DDPM [93] | 25 | 3.75 | 48.63 | - |
| GGDM [101] | 25 | 18.4 | 18.12 | - |
| Analytic DPM [98] | 25 | - | - | 3.83 |
| DPM-Solver Discrete [50] | 20 | 18.53 | - | - |
| ES-DDPM [93] | 10 | 3.93 | 48.81 | - |
| GGDM [101] | 10 | 37.32 | 14.76 | - |
| DPM-Solver Discrete [50] | 10 | 24.4 | - | - |
| ES-DDPM [93] | 5 | 4.25 | 48.04 | - |
| GGDM [101] | 5 | 55.14 | 12.9 | - |

### 5.1.2 High-level vision

FSDM [152] was a few-shot generation framework based on conditional diffusion models. Leveraging advances in vision transformers and diffusion models, FSDM can adapt quickly to various generative processes at test-time and performs well under few-shot generation with strong transfer capability. CARD [153] proposed classification and regression diffusion models, combining a denoising diffusion-based conditional generative model and a pre-trained conditional mean estimator to predict data distribution under given conditions. Though approaching supervised learning from a conditional generation perspective and training with objectives indirectly related to the evaluation metrics, CARD presented a strong ability in uncertainty estimation with the help of diffusion models. Motivated by CLIP [218], GLIDE [154] explored realistic image synthesis conditioned on the text and found that diffusion models with classifier-free guidance yielded high-quality images containing a wide range of learned knowledge. To obtain expressive generative models within a smooth and limited space, LSGM [105] built a diffusion model trained in the latent space with the help of a variational autoencoder framework. SegDiff [155] extended diffusion models for performing image-level segmentation by summing up feature maps from a diffusion-based probabilistic encoder and an image feature encoder. Video diffusion [37], on the other hand, extended diffusion models in the time axis and performed video-level generation by utilizing a typically designed reconstruction-guided conditional sampling method. VQ-Diffusion [156] improved vanilla vector quantized diffusion by exploring classifier-free guidance sampling for discrete diffusion models and presenting a high-quality inference strategy. This method showed superior performance on large datasets such as ImageNet [212] and MSCOCO [219]. Diff-SCM [220] built a deep structural model based on the generative diffusion model. It achieved counterfactual estimation by inferring latent variables with deterministic forward diffusion and intervening in the backward process.

#### 5.1.3 3D vision

[109] was an early work on diffusion-based 3D vision tasks. Motivated by the non-equilibrium thermodynamics, this work analogized points in point clouds as particles in a thermodynamic system and employed the diffusion process in point cloud generation, which achieved competitive performance. PVD [157] was a concurrent work on diffusion-based point cloud generation but performed unconditional generation without additional shape encoders, while a hybrid and point-voxel representation was employed for processing shapes. PDR [110] proposed a paradigm for diffusion-based point cloud completion that applied a diffusion model to generate a coarse completion based on the partial observation and refined the generated output by another network. To deal with point cloud denoising, [158] introduced a neural network to estimate the score of the distribution and denoised point clouds by gradient ascent.

#### 5.1.4 Video modeling

Video diffusion [37] introduced the advances in diffusion-based generative models into the video domain. RVD [159] employed diffusion models to generate a residual to a deterministic next-frame prediction conditioned on the context vector. FDM [160] applied diffusion models to assist long video prediction and performed photo-realistic videos. MCVD [161] proposed a conditional video diffusion framework for video prediction and interpolation based on masking frames in a blockwise manner. RaMViD [162] extended image diffusion models to videos with 3D convolutional neural networks and designed a conditioning technique for video prediction, infilling, and upsampling.

#### 5.1.5 Medical application

It is a natural choice to apply diffusion models to medical images. Score-MRI [163] proposed a diffusion-based framework to solve magnetic resonance imaging (MRI) reconstruction. [164] was a concurrent work but provided a more flexible framework that did not require a paired dataset for training. With a diffusion model trained on medical images, this work leveraged the physical measurement process and focused on sampling algorithms to create image samples that are consistent with the observed measurements and the estimated data prior. R2D2+ [165] combined diffusion-based MRI reconstruction and super-resolution into the same network for end-to-end high-quality medical image generation. [221] explored the application of the generative diffusion model to medical image segmentation and performed counterfactual diffusion.

### 5.2 Sequential modeling

#### 5.2.1 Natural language processing

Benefited by the non-autoregressive mechanism of diffusion models, Diffusion-LM [166] took advantage of continuous diffusions to iteratively denoise noisy vectors into word vectors and performed controllable text generation tasks. Bit Diffusion [167] proposed a diffusion model for generating discrete data and was applied to image caption tasks.

#### 5.2.2 Time series

To deal with time series imputation, CSDI [168] utilized score-based diffusion models conditioned on observed data. Inspired by masked language modeling, a self-supervised training procedure was developed that separates observed values into conditional information and imputation targets. SSSD [169] further introduced structured state space models to capture long-term dependencies in time series data. CSDE [170] proposed a probabilistic framework to model stochastic dynamics and introduced Markov dynamic programming and multi-conditional forward-backward losses to generate complex time series.

### 5.3 Audio

WaveGrad [83] and DiffWave [171] were seminal works that applied diffusion models to raw waveform generation and obtained superior performance. GradTTS [172] and Diff-TTS [173] also implemented diffusion models but generated mel feature instead of raw waves. DiffVC [174] further challenged the one-shot many-to-many voice conversion problem and developed a stochastic differential equation solver. DiffSinger [175] extended the common sound generation to singing voice synthesis based on a shallow diffusion mechanism. Diffsound [176] proposed a sound generation framework conditioned on the text that employed a discrete diffusion model to replace the autoregressive decoder to overcome the unidirectional bias and accumulated errors. EdiTTS [177] was also a diffusion-based audio model for the text-to-speech task. Through coarse perturbations in the prior space, desired edits were induced during denoising reversal. Guided-TTS [178] and Guided-TTS2 [179] were also an early series of text-to-speech models that successfully applied diffusion models in sound generation. [180] combined a voice diffusion model with a spectrogram-domain conditioning method and performed text-to-speech with voices unseen during training. InferGrad [222] considered the inference process in training and improved the diffusion-based text-to-speech model when the number of inference steps is small, enabling fast and high-quality sampling. SpecGrad [181] brought ideas from signal processing and adapted the time-varying spectral envelope of diffusion noise based on the conditioning log-mel spectrogram. ItôTTS [35] unified text-to-speech and vocoder into a framework based on linear SDE. ProDiff [156] proposed a progressive and fast diffusion model for high-quality text-to-speech. Instead of hundreds of iterations, ProDiff parameterized the model by predicting clean data and employed a teacher-synthesized mel-spectrogram as a target to reduce data discrepancies and make a sharp prediction. BinauralGrad [182] was a two-stage diffusion-based framework that explored the application of diffusion models in binaural audio synthesis given mono audio.

### 5.4 AI for science

#### 5.4.1 Molecular conformation generation

ConfGF [183] was an early work on diffusion-based molecular conformation generation models. While preserving rotation and translation equivariance, ConfGF generated samples by Langevin dynamics with physically inspired gradient fields. However, ConfGF only modeled local distances between the first-order, the second-order, and the third-order neighbors and thus failed to capture long-range interactions between non-bounded atoms. To tackle this challenge, DGSM [184] proposed to dynamically construct molecular graph structures between atoms based on their spatial proximity. GeoDiff [39] found that the model was fed with perturbed distance matrices during diffusion learning, which might violate mathematical constraints. Thus, GeoDiff introduced a roto-translational invariant Markov process to impose constraints on the density. EDM [32] further extended the above methods by incorporating discrete atom features and deriving the equations required for log-likelihood computation. Torsional diffusion [185] operated on the space of torsional angles and produced molecular conformations according to a diffusion process limited to the most flexible degrees of freedom.

#### 5.4.2 Material design

CDVAE [186] explored the periodic structure of stable material generation. To address the challenge that stable materials exist only in a low-dimensional subspace with all possible periodic arrangements of atoms, CDVAE designed a diffusion-based network as a decoder with output gradients leading to local minima of energy and updated atom types to capture specific local bonding preferences depending on the neighbors.

Inspired by the recent success of antibody modeling [223], [224], [225], the recent work [187] developed a diffusion-based generative model that explicitly targeted specific antigen structures and generated antibodies. The proposed method jointly sampled antibody sequences and structures and iteratively generated candidates in the sequence-structure space.

Anand *et al.* [188] introduced a diffusion-based generative model for both protein structure and sequence and learned the structural information that is equivariant to rotations and translations. ProteinSGM [189] formulated protein design as an image inpainting problem and applied conditional diffusion-based generation to precisely model the protein structure.

## 6 Conclusions & Discussions

The diffusion model is becoming a popular topic in a wide range of application fields. To utilize the power of the diffusion model, this paper provides a comprehensive and up-to-date review of several aspects of diffusion models using detailed insights on various attitudes, including theory, improved algorithms, and applications. We hope this survey serves as a guide for readers on diffusion model enhancement and model enhancement.

### 6.1 Limitations

There is already a wide range of improved techniques and application fields based on diffusion models. However, more attention to fast sampling leads to less effectiveness in training schemes and original settings. Firstly, there's a variational gap defined by the difference between negative log-likelihood and evidence lower bound. Most current works focus on optimizing ELBO but ignore the minimization task on variation gap, while there is still a relatively huge space to be optimized. Secondly, there's a mismatch between the training objective and evaluation metric performance. Sometimes, the lower loss does not bring higher quality. Thus, mechanisms for unifying the two terms are to be explored, including connection indication and metric improvement. Thirdly, existing works have not paid much attention to noise types and perturbation kernels' types. Instead, Gaussian perturbation, as well as the final state as Gaussian noise, are the most likely to be used, where we have no idea if Gaussian noise is reasonable in some specific tasks. More attention should be drawn to it. Finally, the trade-off between model speed and sampling quality is still unclear and unquantified. The optimization task on quantitative trade-offs may provide insight into adjusting the models' efficiency.

### 6.2 Further Directions

From the perspective of algorithm and application, we present some expected directions in this subsection. On the one hand, there should be more tries on different data types including discrete space, dequantization space, and latent space. Also, experiments for exploring diverse final state noise types and perturbation kernels are needed such as normal distribution, Bernoulli distribution, binomial distribution, and Poisson distribution to broaden diffusion model diversity. Besides, a clear mechanism for loss optimization along with acceleration & quality trade-off will lead to promising influences on controllable regulation and more satisfying performance. On the other hand, there are various fields where diffusion models have been employed in order to obtain better generation performance. However, most of the current applications remain superficial. More problem-specific diffusion models are expected, especially for scientific problems.

## Acknowledgments

This work is supported by the Science and Technology Innovation 2030 - Major Project (No. 2021ZD0150100) and the National Natural Science Foundation of China (No. U21A20427).

We appreciate Dr. Jindong Wang provided the idea for our classification figures.

## References

[1] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *International conference on machine learning*. PMLR, 2014, pp. 1278–1286. (document)

[2] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016. (document)

[3] D. P. Kingma, M. Welling *et al.*, "An introduction to variational autoencoders," *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019. (document)

[4] A. Oussidi and A. Elhassouny, "Deep generative models: Survey," in *2018 International Conference on Intelligent Systems and Computer Vision (ISCV)*. IEEE, 2018, pp. 1–8. (document)

[5] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "A tutorial on energy-based learning," *Predicting structured data*, vol. 1, no. 0, 2006. (document)

[6] J. Ngiam, Z. Chen, P. W. Koh, and A. Y. Ng, "Learning deep energy models," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 1105–1112. (document)

[7] A. G. ALIAS PARTH GOYAL, N. R. Ke, S. Ganguli, and Y. Bengio, "Variational walkback: Learning a transition operator as a stochastic recurrent net," *Advances in Neural Information Processing Systems*, vol. 30, 2017. (document)

[8] T. Kim and Y. Bengio, "Deep directed generative models with energy-based probability estimation," *arXiv preprint arXiv:1606.03439*, 2016. (document)

[9] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," *arXiv preprint arXiv:1609.03126*, 2016. (document)

[10] J. Xie, Y. Lu, S.-C. Zhu, and Y. Wu, "A theory of generative convnet," in *International Conference on Machine Learning*. PMLR, 2016, pp. 2635–2644. (document)

[11] R. Gao, Y. Lu, J. Zhou, S.-C. Zhu, and Y. N. Wu, "Learning generative convnets via multi-grid modeling and sampling," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9155–9164. (document)

[12] R. Kumar, S. Ozair, A. Goyal, A. Courville, and Y. Bengio, "Maximum entropy generators for energy-based models," *arXiv preprint arXiv:1901.08508*, 2019. (document)

[13] E. Nijkamp, M. Hill, S.-C. Zhu, and Y. N. Wu, "Learning nonconvergent non-persistent short-run mcmc toward energy-based model," *Advances in Neural Information Processing Systems*, vol. 32, 2019. (document)

[14] Y. Du and I. Mordatch, "Implicit generation and generalization in energy-based models," *arXiv preprint arXiv:1903.08689*, 2019. (document)

[15] W. Grathwohl, K.-C. Wang, J.-H. Jacobsen, D. Duvenaud, M. Norouzi, and K. Swersky, "Your classifier is secretly an energy based model and you should treat it like one," *arXiv preprint arXiv:1912.03263*, 2019. (document)

[16] G. Desjardins, Y. Bengio, and A. C. Courville, "On tracking the partition function," *Advances in neural information processing systems*, vol. 24, 2011. (document)

[17] R. Gao, E. Nijkamp, D. P. Kingma, Z. Xu, A. M. Dai, and Y. N. Wu, "Flow contrastive estimation of energy-based models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7518–7528. (document)

[18] T. Che, R. Zhang, J. Sohl-Dickstein, H. Larochelle, L. Paull, Y. Cao, and Y. Bengio, "Your gan is secretly an energy-based model and you should use discriminator driven latent sampling," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 275–12 287, 2020. (document)

[19] Y. Qiu, L. Zhang, and X. Wang, "Unbiased contrastive divergence algorithm for training energy-based latent variable models," in *International Conference on Learning Representations*, 2019. (document)

[20] B. Rhodes, K. Xu, and M. U. Gutmann, "Telescoping density-ratio estimation," *Advances in neural information processing systems*, vol. 33, pp. 4905–4916, 2020. (document)

[21] L. Jin, J. Lazarow, and Z. Tu, "Introspective classification with convolutional nets," *Advances in Neural Information Processing Systems*, vol. 30, 2017. (document)

[22] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020. (document)

[23] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE signal processing magazine*, vol. 35, no. 1, pp. 53–65, 2018. (document)

[24] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, "A review on generative adversarial networks: Algorithms, theory, and applications," *IEEE Transactions on Knowledge and Data Engineering*, 2021. (document)

[25] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real nvp," *arXiv preprint arXiv:1605.08803*, 2016. (document)

[26] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *International conference on machine learning*. PMLR, 2015, pp. 1530–1538. (document)

[27] I. Kobyzev, S. J. Prince, and M. A. Brubaker, "Normalizing flows: An introduction and review of current methods," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 11, pp. 3964–3979, 2020. (document)

[28] S. Bond-Taylor, A. Leach, Y. Long, and C. Willcocks, "Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. (document)

[29] G. Papamakarios, E. T. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference." *J. Mach. Learn. Res.*, vol. 22, no. 57, pp. 1–64, 2021. (document), 1

[30] C. Winkler, D. Worrall, E. Hoogeboom, and M. Welling, "Learning likelihoods with conditional normalizing flows," *arXiv preprint arXiv:1912.00042*, 2019. (document)

[31] C. Saharia, W. Chan, H. Chang, C. A. Lee, J. Ho, T. Salimans, D. J. Fleet, and M. Norouzi, "Palette: Image-to-image diffusion models," 2021. [Online]. Available: https://arxiv.org/abs/2111.05826 (document)

[32] E. Hoogeboom, V. G. Satorras, C. Vignac, and M. Welling, "Equivariant diffusion for molecule generation in 3d," in *International Conference on Machine Learning*. PMLR, 2022, pp. 8867–8887. (document), 3.1.3, 3.2.1, 5.4.1

[33] S. Luo, Y. Su, X. Peng, S. Wang, J. Peng, and J. Ma, "Antigen-specific antibody design and optimization with diffusion-based generative models," *bioRxiv*, 2022. [Online]. Available: https://www.biorxiv.org/content/early/2022/07/11/2022.07.10.499510 (document), 3.2.1

[34] H. Tachibana, M. Go, M. Inahara, Y. Katayama, and Y. Watanabe, "It\^{o}-taylor sampling scheme for denoising diffusion probabilistic models using ideal derivatives," *arXiv preprint arXiv:2112.13339*, 2021. (document), 2, 3.1.2

[35] S. Wu and Z. Shi, "Itôtts and itôwave: Linear stochastic differential equation is all you need for audio generation," *arXiv e-prints*, pp. arXiv–2105, 2021. (document), 3.1.3, 5.3

[36] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *International Conference on Machine Learning*. PMLR, 2015, pp. 2256–2265. (document), 2.1.4

[37] J. Ho, T. Salimans, A. A. Gritsenko, W. Chan, M. Norouzi, and D. J. Fleet, "Video diffusion models," in *ICLR Workshop on Deep Generative Models for Highly Structured Data*, 2022. (document), 3.1.3, 5.1.2, 5.1.4

[38] B. L. Trippe, J. Yim, D. Tischer, D. Baker, T. Broderick, R. Barzilay, and T. Jaakkola, "Diffusion probabilistic modeling of protein backbones in 3d for the motif-scaffolding problem," 2022. [Online]. Available: https://arxiv.org/abs/2206.04119 (document)

[39] M. Xu, L. Yu, Y. Song, C. Shi, S. Ermon, and J. Tang, "Geodiff: A geometric diffusion model for molecular conformation generation," in *International Conference on Learning Representations*, 2021. (document), 3.1.3, 5.4.1

[40] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, "Diffwave: A versatile diffusion model for audio synthesis," 2020. [Online]. Available: https://arxiv.org/abs/2009.09761 (document)

[41] H. Kim, S. Kim, and S. Yoon, "Guided-tts: A diffusion model for text-to-speech via classifier guidance," 2021. [Online]. Available: https://arxiv.org/abs/2111.11755 (document)

[42] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *Advances in Neural Information Processing Systems*, vol. 34, pp. 8780–8794, 2021. (document)

[43] N. Chen, Y. Zhang, H. Zen, R. J. Weiss, M. Norouzi, and W. Chan, "Wavegrad: Estimating gradients for waveform generation," 2020. [Online]. Available: https://arxiv.org/abs/2009.00713 (document)

[44] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014. [Online]. Available: https://arxiv.org/abs/1406.2661 1

[45] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org. 1

[46] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013. 1

[47] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8162–8171. (document), 2.5.3, 2, 3.1.1, 3.3, 3.3.1, 5, 6

[48] T. Salimans and J. Ho, "Progressive distillation for fast sampling of diffusion models," *arXiv preprint arXiv:2202.00512*, 2022. (document), 3, 2, 3.1.1, 5

[49] Z. Xiao, K. Kreis, and A. Vahdat, "Tackling the generative learning trilemma with denoising diffusion gans," *arXiv preprint arXiv:2112.07804*, 2021. (document), 2, 3.1.3, 5

[50] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu, "Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps," 2022. [Online]. Available: https://arxiv.org/abs/2206.00927 (document), 2.2.3, 2, 3.1.2, 3, 4, 5

[51] J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. van den Berg, "Structured denoising diffusion models in discrete state-spaces," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 981–17 993, 2021. (document), 2, 3.2.2, 3.3.1, 6

[52] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *CoRR*, vol. abs/2006.11239, 2020. [Online]. Available: https://arxiv.org/abs/2006.11239 2.2, 2.2.1, 2.2.1, 2.2.1, 3.3, 6

[53] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *International Conference on Learning Representations*, 2020. 2.2, 2, 3.1.1, 3.1.2, 5

[54] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," *Advances in Neural Information Processing Systems*, vol. 32, 2019. 2.2.2, 2.3.2, 3, 5.1.1, 6

[55] S. Lyu, "Interpretation and generalization of score matching," *arXiv preprint arXiv:1205.2629*, 2012. 2.2.2

[56] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," *arXiv preprint arXiv:2011.13456*, 2020. 2.2.3, 2.2.3, 2.2.3, 2.4.3, 2, 3.1.2, 3.1.4, 3.2.3, 6

[57] L. Arnold, "Stochastic differential equations," *New York*, 1974. 2.2.3

[58] B. Oksendal, *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013. 2.2.3

[59] A. Hyvärinen and P. Dayan, "Estimation of non-normalized statistical models by score matching." *Journal of Machine Learning Research*, vol. 6, no. 4, 2005. 2.2.3, 2.3.2

[60] D. Maoutsa, S. Reich, and M. Opper, "Interacting particle solutions of fokker–planck equations through gradient–log–density estimation," *Entropy*, vol. 22, no. 8, p. 802, 2020. 2.2.3

[61] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," *Advances in neural information processing systems*, vol. 31, 2018. 2.2.3, 3.1.2, 3.1.4

[62] L. Liu, Y. Ren, Z. Lin, and Z. Zhao, "Pseudo numerical methods for diffusion models on manifolds," *arXiv preprint arXiv:2202.09778*, 2022. 2.2.3, 2, 3.1.2, 3.2.3, 3

[63] Y. Song, S. Garg, J. Shi, and S. Ermon, "Sliced score matching: A scalable approach to density and score estimation," in *Uncertainty in Artificial Intelligence*. PMLR, 2020, pp. 574–584. 2.3.2, 6

[64] R. M. Neal, "Annealed importance sampling," *Statistics and computing*, vol. 11, no. 2, pp. 125–139, 2001. 2.4.1

[65] R. W. Hamming, "Stable predictor-corrector methods for ordinary differential equations," *Journal of the ACM (JACM)*, vol. 6, no. 1, pp. 37–47, 1959. 2.4.3

[66] J. R. Dormand and P. J. Prince, "A family of embedded runge-kutta formulae," *Journal of computational and applied mathematics*, vol. 6, no. 1, pp. 19–26, 1980. 2.4.3

[67] T. Sauer, *Numerical analysis*. Addison-Wesley Publishing Company, 2011. 2.4.3

[68] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007. 2.4.3

[69] A. Borji, "Pros and cons of gan evaluation measures: New developments," *Computer Vision and Image Understanding*, vol. 215, p. 103329, 2022. 2.5.1

[70] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *Advances in neural information processing systems*, vol. 29, 2016. 2.5.1

[71] S. Kullback, *Information theory and statistics*. Courier Corporation, 1997. 2.5.1

[72] S. Barratt and R. Sharma, "A note on the inception score," 2018. [Online]. Available: https://arxiv.org/abs/1801.01973 2.5.2

[73] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf 2.5.2

[74] A. Razavi, A. Van den Oord, and O. Vinyals, "Generating diverse high-fidelity images with vq-vae-2," *Advances in neural information processing systems*, vol. 32, 2019. 2.5.3

[75] T. M. Nguyen, A. Garg, R. G. Baraniuk, and A. Anandkumar, "Infocnf: Efficient conditional continuous normalizing flow using adaptive solvers," 2019. 2.5.3

[76] Z. Ziegler and A. Rush, "Latent normalizing flows for discrete sequences," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7673–7682. 2.5.3

[77] A. Abdelhamed, M. A. Brubaker, and M. S. Brown, "Noise flow: Noise modeling with conditional normalizing flows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3165–3173. 2.5.3

[78] X. Wei, H. van Gorp, L. Gonzalez-Carabarin, D. Freedman, Y. C. Eldar, and R. J. van Sloun, "Deep unfolding with normalizing flow priors for inverse problems," *IEEE Transactions on Signal Processing*, vol. 70, pp. 2962–2971, 2022. 2.5.3

[79] B. Máté, S. Klein, T. Golling, and F. Fleuret, "Flowification: Everything is a normalizing flow," *arXiv preprint arXiv:2205.15209*, 2022. 2.5.3

[80] J. Tomczak and M. Welling, "Vae with a vampprior," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2018, pp. 1214–1223. 2.5.3

[81] P. Notin, J. M. Hernández-Lobato, and Y. Gal, "Improving black-box optimization in vae latent space using decoder uncertainty," *Advances in Neural Information Processing Systems*, vol. 34, pp. 802–814, 2021. 2.5.3

[82] O. Rybkin, K. Daniilidis, and S. Levine, "Simple and effective vae training with calibrated decoders," in *International Conference on Machine Learning*. PMLR, 2021, pp. 9179–9189. 2.5.3

[83] N. Chen, Y. Zhang, H. Zen, R. J. Weiss, M. Norouzi, and W. Chan, "Wavegrad: Estimating gradients for waveform generation," in *International Conference on Learning Representations*, 2020. 3, 3.1.3, 5.3

[84] C. Saharia, W. Chan, H. Chang, C. Lee, J. Ho, T. Salimans, D. Fleet, and M. Norouzi, "Palette: Image-to-image diffusion models," in *ACM SIGGRAPH 2022 Conference Proceedings*, 2022, pp. 1–10. 3, 3.1.3, 5.1.1

[85] Y. Song, C. Durkan, I. Murray, and S. Ermon, "Maximum likelihood training of score-based diffusion models," *Advances in Neural Information Processing Systems*, vol. 34, pp. 1415–1428, 2021. 2, 3.1.3, 3.2.1, 3.3, 3.3.1

[86] D. Kim, S. Shin, K. Song, W. Kang, and I.-C. Moon, "Soft truncation: A universal training technique of score-based diffusion model for high precision score estimation," 2021. [Online]. Available: https://arxiv.org/abs/2106.05527 2, 3.3.1, 3, 6

[87] C.-W. Huang, J. H. Lim, and A. C. Courville, "A variational perspective on diffusion-based generative models and score matching," *Advances in Neural Information Processing Systems*, vol. 34, pp. 22 863–22 876, 2021. 2, 3.3.1

[88] D. Kingma, T. Salimans, B. Poole, and J. Ho, "Variational diffusion models," *Advances in neural information processing systems*, vol. 34, pp. 21 696–21 707, 2021. 2, 3.1.1, 3.1.4, 3.3.1, 6

[89] D. Kim, B. Na, S. J. Kwon, D. Lee, W. Kang, and I.-c. Moon, "Maximum likelihood training of parametrized diffusion model," 2021. 2, 3.1.3, 3.3.2

[90] D. Kim, B. Na, S. J. Kwon, D. Lee, W. Kang, and I.-C. Moon, "Maximum likelihood training of implicit nonlinear diffusion models," *arXiv preprint arXiv:2205.13699*, 2022. 2, 3.1.2, 6

[91] E. Luhman and T. Luhman, "Knowledge distillation in iterative generative models for improved sampling speed," *arXiv preprint arXiv:2101.02388*, 2021. 2, 5

[92] H. Zheng, P. He, W. Chen, and M. Zhou, "Truncated diffusion probabilistic models," *arXiv preprint arXiv:2202.09671*, 2022. 2, 3.1.1, 5, 6

[93] Z. Lyu, X. Xu, C. Yang, D. Lin, and B. Dai, "Accelerating diffusion models via early stop of the diffusion process," *arXiv preprint arXiv:2205.12524*, 2022. 2, 3.1.1, 3.1.3, 3, 4, 5

[94] A. Bansal, E. Borgnia, H.-M. Chu, J. S. Li, H. Kazemi, F. Huang, M. Goldblum, J. Geiping, and T. Goldstein, "Cold diffusion: Inverting arbitrary image transforms without noise," 2022. [Online]. Available: https://arxiv.org/abs/2208.09392 2, 3.1.1

[95] G. Franzese, S. Rossi, L. Yang, A. Finamore, D. Rossi, M. Filippone, and P. Michiardi, "How much is enough? a study on diffusion times in score-based generative models," 2022. [Online]. Available: https://arxiv.org/abs/2206.05173 2, 3.1.1, 5

[96] Q. Zhang and Y. Chen, "Fast sampling of diffusion models with exponential integrator," *arXiv preprint arXiv:2204.13902*, 2022. 2, 3.1.1, 3.1.2, 3.1.4, 5, 6

[97] R. San-Roman, E. Nachmani, and L. Wolf, "Noise estimation for generative diffusion models," *arXiv preprint arXiv:2104.02600*, 2021. 2, 3.1.1

[98] F. Bao, C. Li, J. Zhu, and B. Zhang, "Analytic-dpm: an analytic estimate of the optimal reverse variance in diffusion probabilistic models," *arXiv preprint arXiv:2201.06503*, 2022. 2, 3.1.2, 3, 4, 5, 6

[99] Q. Zhang, M. Tao, and Y. Chen, "gddim: Generalized denoising diffusion implicit models," *arXiv preprint arXiv:2206.05564*, 2022. 2, 3.1.2, 3.1.4, 5

[100] A. Jolicoeur-Martineau, K. Li, R. Piché-Taillefer, T. Kachman, and I. Mitliagkas, "Gotta go fast when generating data with score-based models," 2021. [Online]. Available: https://arxiv.org/abs/2105.14080 2, 3.1.2, 5, 6

[101] D. Watson, W. Chan, J. Ho, and M. Norouzi, "Learning fast samplers for diffusion models by differentiating through sample quality." 2, 3.1.2, 4, 5

[102] D. Watson, J. Ho, M. Norouzi, and W. Chan, "Learning to efficiently sample from diffusion probabilistic models," *arXiv preprint arXiv:2106.03802*, 2021. 2, 3.1.2, 4

[103] K. Pandey, A. Mukherjee, P. Rai, and A. Kumar, "Diffusevae: Efficient, controllable and high-fidelity generation from low-dimensional latents," 2022. [Online]. Available: https://arxiv.org/abs/2201.00308 2, 3.1.3, 3, 5, 6

[104] Q. Zhang and Y. Chen, "Diffusion normalizing flow," *Advances in Neural Information Processing Systems*, vol. 34, pp. 16 280–16 291, 2021. 2, 3.1.3, 5

[105] A. Vahdat, K. Kreis, and J. Kautz, "Score-based generative modeling in latent space," *Advances in Neural Information Processing Systems*, vol. 34, pp. 11 287–11 302, 2021. 2, 3.1.3, 3.1.3, 5.1.2, 5

[106] W. Gong and Y. Li, "Interpreting diffusion score matching using normalizing flow," 2021. [Online]. Available: https://arxiv.org/abs/2107.10072 2, 3.1.4

[107] V. De Bortoli, A. Doucet, J. Heng, and J. Thornton, "Simulating diffusion bridges with score matching," 2021. [Online]. Available: https://arxiv.org/abs/2111.07243 2, 3.1.4

[108] L. Zhou, Y. Du, and J. Wu, "3d shape generation and completion through point-voxel diffusion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 5826–5835. 2, 3.2.1

[109] S. Luo and W. Hu, "Diffusion probabilistic models for 3d point cloud generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2837–2845. 2, 3.1.3, 3.2.1, 5.1.3

[110] Z. Lyu, Z. Kong, X. Xu, L. Pan, and D. Lin, "A conditional point diffusion-refinement paradigm for 3d point cloud completion," *arXiv preprint arXiv:2112.03530*, 2021. 2, 3.1.3, 3.2.1, 5.1.3

[111] A.-C. Cheng, X. Li, S. Liu, M. Sun, and M.-H. Yang, "Autoregressive 3d shape generation via canonical mapping," 2022. [Online]. Available: https://arxiv.org/abs/2204.01955 2, 3.2.1

[112] E. Hoogeboom, D. Nielsen, P. Jaini, P. Forré, and M. Welling, "Argmax flows and multinomial diffusion: Towards non-autoregressive language models," 2021. 2, 3.2.2

[113] E. Hoogeboom, A. A. Gritsenko, J. Bastings, B. Poole, R. v. d. Berg, and T. Salimans, "Autoregressive diffusion models," *arXiv preprint arXiv:2110.02037*, 2021. 2, 3.2.2

[114] A. Campbell, J. Benton, V. De Bortoli, T. Rainforth, G. Deligiannidis, and A. Doucet, "A continuous time framework for discrete denoising models," *arXiv preprint arXiv:2205.14987*, 2022. 2, 3.2.2

[115] S. Gu, D. Chen, J. Bao, F. Wen, B. Zhang, D. Chen, L. Yuan, and B. Guo, "Vector quantized diffusion model for text-to-image synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 696–10 706. 2, 3.2.2

[116] Z. Tang, S. Gu, J. Bao, D. Chen, and F. Wen, "Improved vector quantized diffusion models," *arXiv preprint arXiv:2205.16007*, 2022. 2, 3.2.2

[117] M. Cohen, G. Quispe, S. L. Corff, C. Ollion, and E. Moulines, "Diffusion bridges vector quantized variational autoencoders," 2022. [Online]. Available: https://arxiv.org/abs/2202.04895 2, 3.2.2

[118] P. Xie, Q. Zhang, Z. Li, H. Tang, Y. Du, and X. Hu, "Vector quantized diffusion model with codeunet for text-to-sign pose sequences generation," 2022. [Online]. Available: https://arxiv.org/abs/2208.09141 2, 3.2.2

[119] V. De Bortoli, E. Mathieu, M. Hutchinson, J. Thornton, Y. W. Teh, and A. Doucet, "Riemannian score-based generative modeling," *arXiv preprint arXiv:2202.02763*, 2022. 2, 3.2.3

[120] C.-W. Huang, M. Aghajohari, A. J. Bose, P. Panangaden, and A. Courville, "Riemannian diffusion models," *arXiv preprint arXiv:2208.07949*, 2022. 2, 3.2.3

[121] C. Niu, Y. Song, J. Song, S. Zhao, A. Grover, and S. Ermon, "Permutation invariant graph generation via score-based generative modeling," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 4474–4484. 2, 3.2.3

[122] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, 2021. 3.1.1

[123] R. Huang, Z. Zhao, H. Liu, J. Liu, C. Cui, and Y. Ren, "Prodiff: Progressive fast diffusion model for high-quality text-to-speech," 2022. [Online]. Available: https://arxiv.org/abs/2207.06389 3.1.1

[124] A. M. Mood, "Introduction to the theory of statistics." 1950. 3.1.1

[125] H. Zheng and M. Zhou, "Act: Asymptotic conditional transport," 2020. 3.1.1

[126] S. Mohamed and B. Lakshminarayanan, "Learning in implicit generative models," *Learning*, no. 1/14, 2018. 3.1.2

[127] W. Grathwohl, R. T. Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud, "Ffjord: Free-form continuous dynamics for scalable reversible generative models," *arXiv preprint arXiv:1810.01367*, 2018. 3.1.2, 3.1.4

[128] T. Dockhorn, A. Vahdat, and K. Kreis, "Score-based generative modeling with critically-damped langevin diffusion," *arXiv preprint arXiv:2112.07068*, 2021. 3.1.2

[129] M. Bayram, T. Partal, and G. Orucova Buyukoz, "Numerical methods for simulation of stochastic differential equations," *Advances in Difference Equations*, vol. 2018, no. 1, pp. 1–10, 2018. 3.1.2

[130] V. F. Zaitsev and A. D. Polyanin, *Handbook of exact solutions for ordinary differential equations*. CRC press, 2002. 3.1.2

[131] J. C. Butcher, *Numerical methods for ordinary differential equations*. John Wiley & Sons, 2016. 3.1.2

[132] G. N. Milstein, *Numerical integration of stochastic differential equations*. Springer Science & Business Media, 1994, vol. 313. 3.1.2

[133] E. Platen and N. Bruti-Liberati, *Numerical solution of stochastic differential equations with jumps in finance*. Springer Science & Business Media, 2010, vol. 64. 3.1.2

[134] E. Süli and D. F. Mayers, *An introduction to numerical analysis*. Cambridge university press, 2003. 3.1.2

[135] F. Rabiei, F. Ismail, and M. Suleiman, "Improved runge-kutta methods for solving ordinary differential equations," *Sains Malaysiana*, vol. 42, no. 11, pp. 1679–1687, 2013. 3.1.2

[136] C. W. Gear and D. R. Wells, “Multirate linear multistep methods,” *BIT Numerical Mathematics*, vol. 24, no. 4, pp. 484–502, 1984. 3.1.2

[137] L. F. Shampine, *Numerical solution of ordinary differential equations*. Routledge, 2018. 3.1.2

[138] Q. Han and S. Ji, “Novel multi-step predictor-corrector schemes for backward stochastic differential equations,” 2021. [Online]. Available: https://arxiv.org/abs/2102.05915 3.1.2

[139] M. Hochbruck and A. Ostermann, “Exponential integrators,” *Acta Numerica*, vol. 19, pp. 209–286, 2010. 3.1.2

[140] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2022. 3.1.2

[141] D. Watson, W. Chan, J. Ho, and M. Norouzi, “Learning fast samplers for diffusion models by differentiating through sample quality,” in *International Conference on Learning Representations*, 2021. 3.1.2

[142] R. Kumar, M. Purohit, Z. Svitkina, E. Vee, and J. Wang, “Efficient rematerialization for deep networks,” *Advances in Neural Information Processing Systems*, vol. 32, 2019. 3.1.2

[143] J. Ho, X. Chen, A. Srinivas, Y. Duan, and P. Abbeel, “Flow++: Improving flow-based generative models with variational dequantization and architecture design,” in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 2722–2730. [Online]. Available: https://proceedings.mlr.press/v97/ho19a.html 3.1.3, 3.2.1

[144] E. Hoogeboom, T. S. Cohen, and J. M. Tomczak, “Learning discrete distributions by dequantization,” *arXiv preprint arXiv:2001.11235*, 2020. 3.1.3, 3.2.1

[145] B. Uria, I. Murray, and H. Larochelle, “Rnade: The real-valued neural autoregressive density-estimator,” *Advances in Neural Information Processing Systems*, vol. 26, 2013. 3.1.3, 3.2.1

[146] L. Theis, A. van den Oord, and M. Bethge, “A note on the evaluation of generative models,” in *International Conference on Learning Representations (ICLR 2016)*, 2016, pp. 1–10. 3.1.3, 3.2.1

[147] G. Batzolis, J. Stanczuk, C.-B. Schönlieb, and C. Etmann, “Conditional image generation with score-based diffusion models,” *arXiv preprint arXiv:2111.13606*, 2021. 3.1.3, 5.1.1

[148] B. Kawar, M. Elad, S. Ermon, and J. Song, “Denoising diffusion restoration models,” in *ICLR Workshop on Deep Generative Models for Highly Structured Data*, 2022. 3.1.3, 5.1.1

[149] L. Theis, T. Salimans, M. D. Hoffman, and F. Mentzer, “Lossy compression with gaussian diffusion,” *arXiv preprint arXiv:2206.08889*, 2022. 3.1.3, 5.1.1

[150] H. Li, Y. Yang, M. Chang, S. Chen, H. Feng, Z. Xu, Q. Li, and Y. Chen, “Srdiff: Single image super-resolution with diffusion probabilistic models,” *Neurocomputing*, vol. 479, pp. 47–59, 2022. 3.1.3, 5.1.1

[151] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, and L. Van Gool, “Repaint: Inpainting using denoising diffusion probabilistic models,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11 461–11 471. 3.1.3, 5.1.1

[152] G. Giannone, D. Nielsen, and O. Winther, “Few-shot diffusion models,” *arXiv preprint arXiv:2205.15463*, 2022. 3.1.3, 5.1.2

[153] X. Han, H. Zheng, and M. Zhou, “Card: Classification and regression diffusion models,” *arXiv preprint arXiv:2206.07275*, 2022. 3.1.3, 5.1.2

[154] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen, “Glide: Towards photorealistic image generation and editing with text-guided diffusion models,” *arXiv preprint arXiv:2112.10741*, 2021. 3.1.3, 5.1.2

[155] T. Amit, E. Nachmani, T. Shaharbany, and L. Wolf, “Segdiff: Image segmentation with diffusion probabilistic models,” *arXiv preprint arXiv:2112.00390*, 2021. 3.1.3, 5.1.2

[156] R. Huang, Z. Zhao, H. Liu, J. Liu, C. Cui, and Y. Ren, “Prodiff: Progressive fast diffusion model for high-quality text-to-speech,” *arXiv preprint arXiv:2207.06389*, 2022. 3.1.3, 5.1.2, 5.3

[157] L. Zhou, Y. Du, and J. Wu, “3d shape generation and completion through point-voxel diffusion,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5826–5835. 3.1.3, 5.1.3

[158] S. Luo and W. Hu, “Score-based point cloud denoising,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 4583–4592. 3.1.3, 5.1.3

[159] R. Yang, P. Srivastava, and S. Mandt, “Diffusion probabilistic modeling for video generation,” *arXiv preprint arXiv:2203.09481*, 2022. 3.1.3, 5.1.4

[160] W. Harvey, S. Naderiparizi, V. Masrani, C. Weilbach, and F. Wood, “Flexible diffusion modeling of long videos,” *arXiv preprint arXiv:2205.11495*, 2022. 3.1.3, 5.1.4

[161] V. Voleti, A. Jolicoeur-Martineau, and C. Pal, “Mcvd: Masked conditional video diffusion for prediction, generation, and interpolation,” *arXiv preprint arXiv:2205.09853*, 2022. 3.1.3, 5.1.4

[162] T. Höppe, A. Mehrjou, S. Bauer, D. Nielsen, and A. Dittadi, “Diffusion models for video prediction and infilling,” *arXiv preprint arXiv:2206.07696*, 2022. 3.1.3, 5.1.4

[163] H. Chung and J. C. Ye, “Score-based diffusion models for accelerated mri,” *Medical Image Analysis*, p. 102479, 2022. 3.1.3, 5.1.5

[164] Y. Song, L. Shen, L. Xing, and S. Ermon, “Solving inverse problems in medical imaging with score-based generative models,” in *International Conference on Learning Representations*, 2021. 3.1.3, 5.1.5

[165] H. Chung, E. S. Lee, and J. C. Ye, “Mr image denoising and super-resolution using regularized reverse diffusion,” *arXiv preprint arXiv:2203.12621*, 2022. 3.1.3, 5.1.5

[166] X. L. Li, J. Thickstun, I. Gulrajani, P. Liang, and T. B. Hashimoto, “Diffusion-lm improves controllable text generation,” *arXiv preprint arXiv:2205.14217*, 2022. 3.1.3, 5.2.1

[167] T. Chen, R. Zhang, and G. Hinton, “Analog bits: Generating discrete data using diffusion models with self-conditioning,” *arXiv preprint arXiv:2208.04202*, 2022. 3.1.3, 5.2.1

[168] Y. Tashiro, J. Song, Y. Song, and S. Ermon, “Csdi: Conditional score-based diffusion models for probabilistic time series imputation,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 24 804–24 816, 2021. 3.1.3, 5.2.2

[169] J. M. L. Alcaraz and N. Strodthoff, “Diffusion-based time series imputation and forecasting with structured state space models,” *arXiv preprint arXiv:2208.09399*, 2022. 3.1.3, 5.2.2

[170] S. W. Park, K. Lee, and J. Kwon, “Neural markov controlled sde: Stochastic optimization for continuous-time data,” in *International Conference on Learning Representations*, 2021. 3.1.3, 5.2.2

[171] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, “Diffwave: A versatile diffusion model for audio synthesis,” in *International Conference on Learning Representations*, 2020. 3.1.3, 5.3

[172] V. Popov, I. Vovk, V. Gogoryan, T. Sadekova, and M. Kudinov, “Grad-tts: A diffusion probabilistic model for text-to-speech,” in *International Conference on Machine Learning*. PMLR, 2021, pp. 8599–8608. 3.1.3, 5.3

[173] M. Jeong, H. Kim, S. J. Cheon, B. J. Choi, and N. S. Kim, “Diff-TTS: A Denoising Diffusion Model for Text-to-Speech,” in *Proc. Interspeech 2021*, 2021, pp. 3605–3609. 3.1.3, 5.3

[174] V. Popov, I. Vovk, V. Gogoryan, T. Sadekova, M. S. Kudinov, and J. Wei, “Diffusion-based voice conversion with fast maximum likelihood sampling scheme,” in *International Conference on Learning Representations*, 2021. 3.1.3, 5.3

[175] J. Liu, C. Li, Y. Ren, F. Chen, and Z. Zhao, “Diffsinger: Singing voice synthesis via shallow diffusion mechanism,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 10, 2022, pp. 11 020–11 028. 3.1.3, 5.3

[176] D. Yang, J. Yu, H. Wang, W. Wang, C. Weng, Y. Zou, and D. Yu, “Diffsound: Discrete diffusion model for text-to-sound generation,” *arXiv preprint arXiv:2207.09983*, 2022. 3.1.3, 5.3

[177] J. Tae, H. Kim, and T. Kim, “Editts: Score-based editing for controllable text-to-speech,” *arXiv preprint arXiv:2110.02584*, 2021. 3.1.3, 5.3

[178] H. Kim, S. Kim, and S. Yoon, “Guided-tts: A diffusion model for text-to-speech via classifier guidance,” in *International Conference on Machine Learning*. PMLR, 2022, pp. 11 119–11 133. 3.2, 3.1.3, 5.3

[179] S. Kim, H. Kim, and S. Yoon, “Guided-tts 2: A diffusion model for high-quality adaptive text-to-speech with untranscribed data,” *arXiv preprint arXiv:2205.15370*, 2022. 3.1.3, 5.3

[180] A. Levkovitch, E. Nachmani, and L. Wolf, “Zero-shot voice conditioning for denoising diffusion tts models,” *arXiv preprint arXiv:2206.02246*, 2022. 3.1.3, 5.3

[181] Y. Koizumi, H. Zen, K. Yatabe, N. Chen, and M. Bacchiani, “Specgrad: Diffusion probabilistic model based neural vocoder with adaptive noise spectral shaping,” *arXiv preprint arXiv:2203.16749*, 2022. 3.1.3, 5.3

[182] Y. Leng, Z. Chen, J. Guo, H. Liu, J. Chen, X. Tan, D. Mandic, L. He, X.-Y. Li, T. Qin *et al.*, “Binauralgrad: A two-stage conditional

diffusion probabilistic model for binaural audio synthesis," *arXiv preprint arXiv:2205.14807*, 2022. 3.1.3, 5.3

[183] C. Shi, S. Luo, M. Xu, and J. Tang, "Learning gradient fields for molecular conformation generation," in *International Conference on Machine Learning*. PMLR, 2021, pp. 9558–9568. 3.1.3, 5.4.1

[184] S. Luo, C. Shi, M. Xu, and J. Tang, "Predicting molecular conformation via dynamic graph score matching," *Advances in Neural Information Processing Systems*, vol. 34, pp. 19 784–19 795, 2021. 3.1.3, 5.4.1

[185] B. Jing, G. Corso, R. Barzilay, and T. S. Jaakkola, "Torsional diffusion for molecular conformer generation," in *ICLR2022 Machine Learning for Drug Discovery*, 2022. 3.2, 3.1.3, 5.4.1

[186] T. Xie, X. Fu, O.-E. Ganea, R. Barzilay, and T. S. Jaakkola, "Crystal diffusion variational autoencoder for periodic material generation," in *International Conference on Learning Representations*, 2021. 3.1.3, 5.4.2

[187] S. Luo, Y. Su, X. Peng, S. Wang, J. Peng, and J. Ma, "Antigen-specific antibody design and optimization with diffusion-based generative models," *bioRxiv*, 2022. 3.2, 3.1.3, 5.4.2

[188] N. Anand and T. Achim, "Protein structure and sequence generation with equivariant denoising diffusion probabilistic models," *arXiv preprint arXiv:2205.15019*, 2022. 3.1.3, 5.4.2

[189] J. S. Lee and P. M. Kim, "Proteinsgm: Score-based generative modeling for de novo protein design," *bioRxiv*, 2022. 3.1.3, 5.4.2

[190] L. Alili, P. Graczyk, and T. Zak, "On inversions and doob h-transforms of linear diffusions," *Lecture Notes in Mathematics*, vol. 2137, 09 2012. 3.1.4

[191] A. Van Den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017. 3.2.2, 3.3

[192] H. A. Pierson and M. S. Gashler, "Deep learning in robotics: a review of recent research," *Advanced Robotics*, vol. 31, no. 16, pp. 821–835, 2017. 3.2.3

[193] N. Sünderhauf, O. Brock, W. Scheirer, R. Hadsell, D. Fox, J. Leitner, B. Upcroft, P. Abbeel, W. Burgard, M. Milford *et al.*, "The limits and potentials of deep learning for robotics," *The International journal of robotics research*, vol. 37, no. 4-5, pp. 405–420, 2018. 3.2.3

[194] R. P. De Lima, K. Marfurt, D. Duarte, and A. Bonar, "Progress and challenges in deep learning analysis of geoscience images," in *81st EAGE Conference and Exhibition 2019*, vol. 2019, no. 1. European Association of Geoscientists & Engineers, 2019, pp. 1–5. 3.2.3

[195] L. Zhang, L. Zhang, and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state of the art," *IEEE Geoscience and remote sensing magazine*, vol. 4, no. 2, pp. 22–40, 2016. 3.2.3

[196] M. Reichstein, G. Camps-Valls, B. Stevens, M. Jung, J. Denzler, N. Carvalhais *et al.*, "Deep learning and process understanding for data-driven earth system science," *Nature*, vol. 566, no. 7743, pp. 195–204, 2019. 3.2.3

[197] J. Wang, H. Cao, J. Z. Zhang, and Y. Qi, "Computational protein design with deep learning neural networks," *Scientific reports*, vol. 8, no. 1, pp. 1–9, 2018. 3.2.3

[198] W. Cao, Z. Yan, Z. He, and Z. He, "A comprehensive survey on geometric deep learning," *IEEE Access*, vol. 8, pp. 35 929–35 949, 2020. 3.2.3

[199] J. M. Lee, "Smooth manifolds," in *Introduction to smooth manifolds*. Springer, 2013, pp. 1–31. 3.2.3

[200] ——, *Introduction to Riemannian manifolds*. Springer, 2018, vol. 176. 3.2.3

[201] G. Leobacher and A. Steinicke, "Existence, uniqueness and regularity of the projection onto differentiable manifolds," *Annals of global analysis and geometry*, vol. 60, no. 3, pp. 559–587, 2021. 3.2.3

[202] G. Wanner and E. Hairer, *Solving ordinary differential equations II*. Springer Berlin Heidelberg New York, 1996, vol. 375. 3.2.3

[203] L. Wu, H. Lin, Z. Gao, C. Tan, and S. Z. Li, "Self-supervised on graphs: Contrastive, generative, or predictive," 2021. 3.2.3

[204] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," 2016. 3.3

[205] S. Zhao, J. Song, and S. Ermon, "Infovae: Information maximizing variational autoencoders," *arXiv preprint arXiv:1706.02262*, 2017. 3.3

[206] S. Semeniuta, A. Severyn, and E. Barth, "A hybrid convolutional variational autoencoder for text generation," *arXiv preprint arXiv:1702.02390*, 2017. 3.3

[207] F. Vargas, P. Thodoroff, A. Lamacraft, and N. Lawrence, "Solving schrödinger bridges via maximum likelihood," *Entropy*, vol. 23, no. 9, p. 1134, 2021. 3.3.1

[208] V. De Bortoli, J. Thornton, J. Heng, and A. Doucet, "Diffusion schrödinger bridge with applications to score-based generative modeling," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 695–17 709, 2021. 3.3.1

[209] T. Chen, G.-H. Liu, and E. A. Theodorou, "Likelihood training of schr\" odinger bridge using forward-backward sdes theory," *arXiv preprint arXiv:2110.11291*, 2021. 3.3.1

[210] C. Cremer, X. Li, and D. Duvenaud, "Inference suboptimality in variational autoencoders," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1078–1086. 3.3.2

[211] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009. 4

[212] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017. 4

[213] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. 4

[214] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv preprint arXiv:1506.03365*, 2015. 4

[215] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 4

[216] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/ 4

[217] H. Chung, B. Sim, D. Ryu, and J. C. Ye, "Improving diffusion models for inverse problems using manifold constraints," *arXiv preprint arXiv:2206.00941*, 2022. 4

[218] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8748–8763. 5.1.2

[219] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755. 5.1.2

[220] P. Sanchez and S. A. Tsaftaris, "Diffusion causal models for counterfactual estimation," in *First Conference on Causal Learning and Reasoning*, 2021. 5.1.2

[221] P. Sanchez, A. Kascenas, X. Liu, A. Q. O'Neil, and S. A. Tsaftaris, "What is healthy? generative counterfactual diffusion for lesion localization," *arXiv preprint arXiv:2207.12268*, 2022. 5.1.5

[222] Z. Chen, X. Tan, K. Wang, S. Pan, D. Mandic, L. He, and S. Zhao, "Infergrad: Improving diffusion models for vocoder by considering inference in training," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 8432–8436. 5.3

[223] W. Jin, J. Wohlwend, R. Barzilay, and T. S. Jaakkola, "Iterative refinement graph neural network for antibody sequence-structure co-design," in *International Conference on Learning Representations*, 2021. 5.4.2

[224] T. Fu and J. Sun, "Antibody complementarity determining regions (cdrs) design using constrained energy model," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 389–399. 5.4.2

[225] W. Jin, R. Barzilay, and T. Jaakkola, "Antibody-antigen docking and design via hierarchical structure refinement," in *International Conference on Machine Learning*. PMLR, 2022, pp. 10 217–10 227. 5.4.2

[226] Y. Song and S. Ermon, "Improved techniques for training score-based generative models," *Advances in neural information processing systems*, vol. 33, pp. 12 438–12 448, 2020. 6

# Appendix A
# Benchmarks on CIFAR-10 Dataset

TABLE 5
Benchmarks on CIFAR-10 (NFE < 1000)

| Method | NFE | FID | IS | NLL |
|---|---|---|---|---|
| Diffusion Step [95] | 600 | 3.72 | - | - |
| ES-DDPM [93] | 600 | 3.17 | - | - |
| Diffusion Step [95] | 400 | 14.38 | - | - |
| Diffusion Step [95] | 200 | 5.44 | - | - |
| Gotta Go Fast VP [96] | 180 | 2.44 | - | - |
| Gotta Go Fast VE [96] | 180 | 3.40 | - | - |
| Gotta Go Fast VP [96] | 148 | 2.73 | - | - |
| Gotta Go Fast VE [96] | 148 | 10.15 | - | - |
| LSGM [105] | 138 | 2.10 | - | - |
| DDIM [53] | 100 | 4.16 | - | - |
| FastDPM [96] | 100 | 2.86 | - | - |
| TDPM [92] | 100 | 3.10 | 9.34 | - |
| DiffuseVAE [103] | 100 | 11.71 | 8.27 | - |
| DiffFlow [104] | 100 | 14.14 | - | 3.04 |
| Analytic DPM [98] | 100 | - | - | 3.59 |
| Efficient Sampling [100] | 64 | 3.08 | - | - |
| DPM-Solver [50] | 51 | 2.59 | - | - |
| DDIM [53] | 50 | 4.67 | - | - |
| FastDPM [96] | 50 | 3.2 | - | - |
| Improved DDPM [47] | 50 | 4.99 | - | - |
| TDPM [92] | 50 | 3.3 | 9.22 | - |
| DEIS [100] | 50 | 2.57 | - | - |
| gDDIM [99] | 50 | 2.28 | - | - |
| DPM-Solver Discrete [50] | 44 | 3.48 | - | - |
| Efficient Sampling [100] | 32 | 3.17 | - | - |
| Improved DDPM [47] | 25 | 7.53 | - | - |
| GGDM [101] | 25 | 4.25 | 9.19 | - |
| DDIM [53] | 20 | 6.84 | - | - |
| FastDPM [96] | 20 | 5.05 | - | - |
| DEIS [100] | 20 | 2.86 | - | - |
| DPM-Solver [50] | 20 | 2.87 | - | - |
| DPM-Solver Discrete [50] | 20 | 3.72 | - | - |
| Efficient Sampling [100] | 16 | 3.41 | - | - |
| DDIM [53] | 10 | 13.36 | - | - |
| FastDPM [96] | 10 | 9.90 | - | - |
| GGDM [101] | 10 | 8.23 | 8.90 | - |
| Analytic DPM [98] | 10 | - | - | 4.11 |
| DEIS [100] | 10 | 4.17 | - | - |
| DPM-Solver [50] | 10 | 6.96 | - | - |
| DPM-Solver Discrete [50] | 10 | 10.16 | - | - |
| Progressive Distillation [48] | 8 | 2.57 | - | - |
| Denoising Diffusion GAN [49] | 8 | 4.36 | 9.43 | - |
| GGDM [101] | 5 | 13.77 | 8.53 | - |
| DEIS [100] | 5 | 15.37 | - | - |
| Progressive Distillation [48] | 4 | 3.00 | - | - |
| TDPM [92] | 4 | 3.41 | 9.00 | - |
| Denoising Diffusion GAN [49] | 4 | 3.75 | 9.63 | - |
| Progressive Distillation [48] | 2 | 4.51 | - | - |
| TDPM [92] | 2 | 4.47 | 8.97 | - |
| Denoising Diffusion GAN [49] | 2 | 4.08 | 9.80 | - |
| Denoising student [91] | 1 | 9.36 | 8.36 | - |
| Progressive Distillation [48] | 1 | 9.12 | - | - |
| TDPM [92] | 1 | 8.91 | 8.65 | - |

TABLE 6
Benchmarks on CIFAR-10 (NFE ≥ 1000)

| Method | NFE | FID | IS | NLL |
|---|---|---|---|---|
| Improved DDPM [47] | 4000 | 2.90 | - | - |
| VE SDE [56] | 2000 | 2.20 | 9.89 | - |
| VP SDE [56] | 2000 | 2.41 | 9.68 | 3.13 |
| sub-VP SDE [56] | 2000 | 2.41 | 9.57 | 2.92 |
| DDPM [52] | 1000 | 3.17 | 9.46 | 3.72 |
| NCSN [54] | 1000 | 25.32 | 8.87 | - |
| SSM [63] | 1000 | 54.33 | - | - |
| NCSNv2 [226] | 1000 | 10.87 | 8.40 | - |
| D3PM [51] | 1000 | 7.34 | 8.56 | 3.44 |
| Efficient Sampling [100] | 1000 | 2.94 | - | - |
| NCSN++ [86] | 1000 | 2.33 | 10.11 | 3.04 |
| DDPM++ [86] | 1000 | 2.47 | 9.78 | 2.91 |
| TDPM [92] | 1000 | 3.07 | 9.24 | - |
| VDM [88] | 1000 | 4.00 | - | - |
| DiffuseVAE [103] | 1000 | 8.72 | 8.63 | - |
| Analytic DPM [98] | 1000 | - | - | 3.59 |
| Gotta Go Fast VP [96] | 1000 | 2.49 | - | - |
| Gotta Go Fast VE [96] | 1000 | 3.14 | - | - |
| INDM [90] | 1000 | 2.28 | - | 3.09 |