A Survey of Deep Generative Models: Unravelling Principles and Insights

Letian Yang
moekid101@sjtu.edu.cn

1 Introduction

Deep generative models (DGM) encompass a class of machine learning models that leverage deep neural networks for generation tasks. These models possess the ability to learn from intricate data distributions and generate synthetic samples resembling the training data.

In this report, I will elaborate on the principles underlying some deep generative models, aiming to acquire a deeper understanding of knowledge about deep generative models. Among these models, diffusion models as the contemporary state-of-the-art, will be put particular emphasis on.

2 From Generative Modelling To Diffusion Learning

We formulate our notation and definition in this section, through which we present a general mathematical formulation of generative modelling and demonstrate several popular solutions.

Suppose the given dataset $\mathcal{D}$ is drawn *independent and identically distributed* (i.i.d.) from some **intractable** probability distribution $\mathcal{X}$, i.e. $\mathcal{D} \sim \mathcal{X}^m$ where m is the number of samples. Each sample $\boldsymbol{x} \sim \mathcal{X}$ has a dimension of n ($\boldsymbol{x} \in \mathbb{R}^n$, and typically n is large). The goal of generative modelling is to construct a generator $g : \mathbb{R}^q \to \mathbb{R}^n$ that maps a **tractable** distribution $\mathcal{Z}$ of the latent vector $\boldsymbol{z} \in \mathbb{R}^q$ to the original distribution $\mathcal{X}$. In other words, we want $g(\mathcal{Z}) \approx \mathcal{X}$. In order to explicitly denote the models trained using neural networks, we use $g_{\boldsymbol{\theta}}$ to denote the GNN generator parameterized by $\boldsymbol{\theta} \in \mathbb{R}^{N_\theta}$.

In reality, training the parameters $\boldsymbol{\theta}$ involves estimating the distance between ground truth $\mathcal{X}$ and generated probability distribution $g(\mathcal{Z})$, which is indeed an extremely challenging task since the problem setting is after all *unsupervised*, preventing us from sampling $(\boldsymbol{x}, \boldsymbol{z})$ pairs directly. Instead, as is shown in Ruthotto and Haber (2021), the hypothesis test problem is a **two-sample test problem**, which is significantly harder than conducting conventional hypothesis testing, espcially with the absence of an appropriate and powerful test statistics. Notably, traditional approaches like Kullback-Leibler divergence requires formulation of probability distributions, which turns out to be invisible in this case.

Having defined the problem mathematically, we present several popular approaches for training $g_{\boldsymbol{\theta}}$.

2.1 Generative Adversarial Networks

Introduced by Goodfellow et al. (2014), *generative adversarial networks* (GAN) proposed one solution to the two-sample test problem. In the paper, a binary classifier $d_{\boldsymbol{\phi}} : \mathbb{R}^n \to [0, 1]$ works as the discriminator predicting the probability of a given sample belonging to the training set. The cross-entropy loss function is taken as the loss of GANs

$$\mathcal{L}_{\text{GAN}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbf{E}_{\boldsymbol{x} \sim \mathcal{X}} \left[\log d_{\boldsymbol{\phi}}(\boldsymbol{x})\right] + \mathbf{E}_{\boldsymbol{z} \sim \mathcal{Z}} \left[\log \left(1 - d_{\boldsymbol{\phi}} \left(g_{\boldsymbol{\theta}}(\boldsymbol{z})\right)\right)\right] \tag{1}$$

Intuitively, the first addition term models how real data is appreciated by the discriminator while the second term models to what extent the discriminator identifies false samples. In training process, a

Nash equilibrium $(\boldsymbol{\theta}^*, \boldsymbol{\phi}^*)$ is defined as the goal of GANs:

$$\boldsymbol{\phi}^* = \arg\max_{\boldsymbol{\phi}} \mathcal{L}_{\text{GAN}}(\boldsymbol{\theta}^*, \boldsymbol{\phi}) \text{ and } \boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathcal{L}_{\text{GAN}}(\boldsymbol{\theta}, \boldsymbol{\phi}^*) \tag{2}$$

In practice, the Nash equilibrium is almost impossible to be solved or optimized. Instead, an *expectation maximization* (EM) styled alternating optimization is adopted. In other words,

$$\boldsymbol{\phi}^{(i+1)} = \boldsymbol{\phi}^{(i)} + \alpha^{(i)} \nabla_{\boldsymbol{\phi}} \mathcal{L}_{\text{GAN}}(\boldsymbol{\theta}^{(i)}, \boldsymbol{\phi}^{(i)}) \tag{3}$$

$$\boldsymbol{\theta}^{(i+1)} = \boldsymbol{\theta}^{(i)} - \beta^{(i)} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{GAN}}(\boldsymbol{\theta}^{(i)}, \boldsymbol{\phi}^{(i)}) \tag{4}$$

are adopted as the update rule for training in reality. Here, $\alpha^{(i)}$ and $\beta^{(i)}$ are positive real numbers while "$+$" and "$-$" indicates gradient ascent and descent respectively.

Beyond GAN itself, Arjovsky and Bottou (2017) performed detailed and expansive experiments on GANs and proposed refined training methods which further solves the two-sample hypothesis test problem; Arjovsky et al. (2017) introduced Wasserstein-1 distance, or the *earth mover distance* (EMD) to measure the distance between $g_{\boldsymbol{\theta}}(\mathcal{Z})$ and $\mathcal{X}$, where the continuous loss function brought about better theoretical bounds.

2.2 Variational Autoencoders

To understand *variational autoencoders* (VAE) we start from dimension reduction and autoencoders. In autoencoders, neural networks learn the encoder $e : \mathbb{R}^n \to \mathbb{R}^q$ and the decoder $d : \mathbb{R}^q \to \mathbb{R}^n$ where $q \ll n$. Therefore, the latent distribution $\mathcal{Z}$ over $\mathbb{R}^q$ provides a bottleneck for information passage, and thus building a satisfactory dimension reducer. This architecture is designed to spot and retain necessary structures within the dataset while eliminating redundant or insignificant information.

At first glance it appears that the encoder-decoder structure is strongly associated with generation tasks. However, it imposes strong constraints on the latent space as the outputs of encoders, including a **tractable** latent distribution. This implies that when used as generative models, there is high probability that an autoencoder overfits the dataset.

In this context, variational autoencoders are introduced as an **autoencoder that is regularized to prevent overfitting and the latent space is well-endowed for generation processes**. In VAEs, instead of modelling the input as a single point $\boldsymbol{z} \in \mathbb{R}^q$, we encode the input as a tractable distribution (Gaussian distribution in practice) over the latent space, while decoding is conducted on a sampled point from that distribution. We describe the mathematics defined in Kingma and Welling (2013) in below.

As our objective for maximization, the likelihood of each sample $\boldsymbol{x} \sim \mathcal{X}$ is

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) p_{\mathcal{Z}}(\boldsymbol{z})}{p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})} \tag{5}$$

where the posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ is approximated by our introduced encoder $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ parameterized by $\boldsymbol{\phi}$. Then the log-likelihood is formulated as

$$\begin{aligned}
\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) =& \mathbf{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x})\right] = \mathbf{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log \left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \frac{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}{p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})}\right)\right] \\
=& \mathbf{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log \left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\right)\right] + \text{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \| p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})\right)
\end{aligned} \tag{6}$$

Then the *evidence lower bound* (ELBO) loss $\mathcal{L}_{\text{ELBO}}$ is defined as the first term in the equation.

$$\mathcal{L}_{\text{ELBO}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\mathbf{E}_{\boldsymbol{x}\sim\mathcal{X}} \mathbf{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z}) - \log q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\right] \tag{7}$$

Although we have ignored the KL divergence term, which is intractable, the tightness of our lower bound improves as we minimize the loss with respect to $\boldsymbol{\phi}$. Therefore, optimizing the ELBO loss guarantees promising result for maximizing the likelihood of generated sample.

Besides the mathematical principles, regularization techniques are also applied to ensure the conitnuity and completeness of latent space to ensure sampled points won't give meaningless content.

2.3 Problems with GAN and VAE

Having delved into the principles and training techniues of GAN and VAE, we have had a basic concept of these two generative models. Now we contemplate the challenges inherent in these models.

- **Mode Collapse**. Thanh-Tung and Tran (2020) pointed out that GANs suffer from the problem of mode collapse and catastrophic forgetting. An intuitive analogy for mode collapse is an artist keeping creating similar works to prevent being criticized by the public. The core idea is that there is no guarantee of where the Nash equilibrium is located at, and local maxima could lead to the problem of mode collapse.
  Mode collapse is straightforwardly explained as local optima being achieved when the generator stucks in some certain mode of generation instead of capturing the full diversity of training set data distribution. Despite constant and extensive efforts paid into relieving the issue, there is still no perfect solution.
- **Stable Training Dynamics**. As is described in Goodfellow et al. (2014), GANs are very sensitive to hyperparameter choices, including network architectures. A slight shift in hyperparameters might lead to observable performance discrepancies for GANs.
- **Problematic VAE Loss**. According to Ruthotto and Haber (2021), the standard VAE loss can be further decomposed into two parts.

  $$\mathcal{L}_{\text{ELBO}}(\boldsymbol{\theta}, \boldsymbol{\phi}) = \mathbf{E}_{\boldsymbol{x}\sim\mathcal{X}}\mathbf{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}\left[-\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})\right] + \mathbf{E}_{\boldsymbol{x}\sim\mathcal{X}}\left[\text{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})\|p_{\mathcal{Z}}(\boldsymbol{z})\right)\right] \quad (8)$$

  where the first term is reconsutruction loss measuring how well the model reconstructs inputs from the latent representation, while the second term is KL divergence loss regularizing the latent space by penalizing deviations from a chosen prior.
  These two terms demonstrate a conflict. Learning with the reconstruction loss leads to samples generally not normally distributed in the latent space, diminishing the quality of samples generated from $\boldsymbol{z} \sim \mathcal{Z}$; similarly, pursuing normal distributions in the latent space means a substantial reconstruction error.

With the above three issues clarified, researchers turned to searching for new solutions. Then we are presented with diffusion models.

3 Diffusion Models

By definition, diffusion models are models learning to reverse a process gradually degrading the structural information in training samples. As its definition suggests, diffusion learning involves two phases, both of which are parameterized *Markov chains*. In the forward phase of training, training data is destroyed step by step and is finally overwhelmed by Gaussian noise. In the backward phase, the model reconstruct image by progressively removing the noise.

Despite the common idea of two Markov chains among diffusion models, there are mainly three predominant formulations: *denoising diffusion probabilistic models* (DDPM), *score-based generative models* (SGM), and *stochastic differential equations* (SDE). Different from other literature (e.g. Croitoru et al. (2023)) which introduce DDPM and SGM first and then synthesize them with SDE, we depart from diffusion expressed with SDE in physics.

3.1 Dynamics and Sampling Process

From the very beginning, a(n) (informal) definition of diffusion is a continuous stochastic process with Markov property, which is described by *stochastic differential equations* (SDE).

Def Diffusion: $\{X(t)\}_{t\geq 0}$ is a diffusion specified by $\mu(x,t)$ and $\sigma^2(t)$ iff.

- For any t and h small enough, $X(t+h) - X(t) \sim \mathcal{N}(\mu(X(t),t), \sigma^2(t))$.
- For any $0 < t_1 < t_2$, $X(t_1+h) - X(t_1)$ and $X(t_2+h) - X(t_2)$ are independent.

Def Brownian Motion: $\{W(t)\}_{t\geq 0}$ is a standard Brownian motion iff.

- $W(0) = 0$.

- $\forall 0 \leq t_0 \leq t_1 \leq ... \leq t_n$, $\{W(t_1) - W(t_0), ..., W(t_n) - W(t_{n-1})\}$ are independent.
- $\forall s, t$, $W(s+t) - W(s) \sim \mathcal{N}(0, t)$.
- $W(t)$ is continuous almost surely.

Using these two definitions, we have $\mathrm{d}X(t) = \mu(X(t), t)\mathrm{d}t + \sigma(t)\mathrm{d}W(t)$. More generally, a diffusion process is expressed with Brownian motion as

$$\mathrm{d}X = f(X, t)\mathrm{d}t + g(t)\mathrm{d}W \tag{9}$$

According to Anderson (1982), the reverse-time diffusion equation is expressed as

$$\mathrm{d}X = \left(f(X, t) - (g(t))^2 \nabla_X \log q_t(X)\right) \mathrm{d}t + g(t)\mathrm{d}\overline{W} \tag{10}$$

where $q_t(X)$ is the distribution of X at time t, and $\overline{W}$ is the reversed standard Wiener process. Notably, Eq. 9 and Eq. 10 equips us with everything we need to solve the problem of both the forward phase and the backward phase.

With the solution of forward and backward diffusion process in continuous case, the only work left lies in discretize it with respect to a small time interval Δt. We will show in the following sections that previously mentioned DDPM and SGM are both instances of SDE, which is solved with the help of Eq. 10.

3.1.1 *Denoising Diffusion Probabilistic Models* (DDPM)

This method builds the Markov process through defining sequence $\{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T\}$ from training sample $\boldsymbol{x}_0$. The Gaussian perturbation in Ho et al. (2020) is defined by

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}\right) \tag{11}$$

where β_t is a hyperparameter. If we apply Markov property ($q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \prod_{i=0}^{t-1} q(\boldsymbol{x}_{i+1}|\boldsymbol{x}_i)$) and define $\alpha_t = \prod_{s=0}^{t}(1-\beta_s)$, the forward phase is formulated by

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, (1-\alpha_t)\boldsymbol{I}\right) \tag{12}$$

which is intuitively explained as injecting small noises at each step and finally destroying all structures in the input $\boldsymbol{x}_0$. Then the authors defined their reverse Markov process as

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1}; \mu_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t), \Sigma_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)) \tag{13}$$

with learnable parameters $\mu_{\boldsymbol{\theta}}$ and $\Sigma_{\boldsymbol{\theta}}$. The training process is carried out by minimizing the KL divergence $\mathrm{KL}\left(q(\boldsymbol{x}_0, ..., \boldsymbol{x}_T) \| p_{\boldsymbol{\theta}}(\boldsymbol{x}_0, ..., \boldsymbol{x}_T)\right)$. Here according to the original paper, this KL divergence is further decomposed into the *evidence lower bound* $\mathbf{E}[-\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0)]$ and thus the same optimization technique as that of variational autoencoders is adopted. What's more, it is evident that the network architecture, when stacked together, is an equivalent of *U-Net* proposed by Ronneberger et al. (2015).

In Song et al. (2020), the SDE is defined as $\mathrm{d}X = -\frac{1}{2}\beta(t)\boldsymbol{x}\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}W$ with $\beta\left(\frac{t}{T}\right) = T\beta_t$.

3.1.2 *Score-Based Generative Models* (SGM)

Up till now, we have encountered variants of the *Fisher* score $\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x})$ twice, once in Eq. 8 as part of the loss of VAE, the other in the loss function of DDPM (notice that these loss functions will be taken derivatives in practice). In SGM, Song and Ermon (2019) proposed using another score, namely the *Stein* score defined as $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$, which is the gradient of the same function of $\boldsymbol{x}$ instead of $\boldsymbol{\theta}$. In their *noise-conditional score network* (NCSN), the Markov chain is defined by $q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \boldsymbol{x}_0, \sigma_t^2 \boldsymbol{I})$ (here $\sigma_t < \sigma_{t+1}$ for any t to indicate an increasing volume of noise). The training objective is to let neural network $s_{\boldsymbol{\theta}}$ estimate the Stein score, corresponding to its name of a score network.

$$\mathbf{E}_{t, \boldsymbol{x}_0, \boldsymbol{x}_t \sim q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\lambda(t)\sigma_t^2 \|\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t) - s_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)\|^2\right] \tag{14}$$

When it comes to sampling, the authors introduced *annealed Langevin dynamics* (ALD) as an independent process from training. In Langevin dynamics (in statistical physics context), the Langevin equation is

$$\mathrm{d}X = -\nabla U(X_t)\mathrm{d}t + \sqrt{2\beta^{-1}}\mathrm{d}W \tag{15}$$

where $U(X_t)$ is the potential energy function and β^{-1} is the temperature. The only distinction between annealed LD and LD is that the temperature decreases with time. Consequently, applying ALD in the sampling of NCSN is in essence taking the $\log p(\boldsymbol{x})$ as a potential energy function and find its extremum in simulated annealing. It is noteworthy that since the training process is completely independent from the sampling process, we can use whatever sampling strategy other than ALD.

In Song et al. (2020), the SDE is defined as $\mathrm{d}X = \sqrt{\frac{\mathrm{d}\sigma(t)^2}{\mathrm{d}t}}\mathrm{d}W$ with $\sigma\left(\frac{t}{T}\right) = \sigma_t$.

3.2 Applications of Diffusion Models

The application of diffusion models deserves attention because most people have heard of currently state-of-the-art image generators like *Stable Diffusion* and *MidJourney* but are completely ignorant of other usages of diffusion models. Therefore, this section is focused on research areas other than computer vision and most parts are referenced from Yang et al. (2023).

In **natural language processing**, the current research is mainly concentrated on refinement of large autoregressive language models. In order to deploy LLMs to real-world applications, we hope to control the content generated by LLMs. Li et al. (2022) proposes a diffusion model denoising Gaussian-distributed noise vectors and transforming them into words. With continuous latent variable, controlling over the generated text becomes feasible.

In **time series imputation and forecasts**, it seems natural that the hierarchical or sequential structure of diffusion models are applied to time series related tasks. Alcaraz and Strodthoff (2022) proposes *structured state space diffusion* (SSSD) which utilizes both diffusion models and structured state-space models to capture long-term dependencies in time series.

Besides computer vision and these two fields mentioned above, diffusion models are also used in scientific researches including molecular graph modelling and material design. These applications require different data structures like graph neural networks, and demand reliability and accuracy in the model it uses. All these applications have depicted a bright future for a wider usage of diffusion learning in cross-disciplinary fields.

4 Future Directions

Despite the relatively long history of generative neural networks and extensive research efforts paid into content generation, it hasn't been a long time since diffusion models, the currently most popular and most robust architecture, was born, and people's understanding of diffusion learning is still insufficient.

For example, it has been well-observed that diffusion models cost too much during the inference stage, or the sampling phase. Although there has been experimental strides, there is still a theoretical gap in addressing the trade-off between sampling efficiency and generation quality. There are likely two solutions heading for different directions, **modifying the assumptions of diffusion**, or more specifically, replacing the assumption that the forward phase removes all information with something else, and **deriving theoretical evidence for the effectiveness of diffusion models over its rivals** (for example, VAE and GAN).

What's more, if we take the traditional VAEs or GANs as reference, the latent representation of these architectures have been comprehensively studied, and human experts have been equipped with the capabilities to manipulate semantic representations for VAEs and GANs. This proposes another research area of **latent space representation for diffusion models**.

5 Conclusion

We have presented a (relatively) comprehensive survey on deep generative models from the perspective of theory and principles, where the main endeavor revolves around the prominent diffusion models.

During composition of this report I have consulted a vast array of papers, blogs and tutorials. Through the reading process, I have tried my best to understand the mathematical formulas and principles posed in the papers and adopted by the model implementations. Through these experiences I have reaped a substantial amount of knowledge.

References

Juan Miguel Lopez Alcaraz and Nils Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. arXiv preprint arXiv:2208.09399, 2022.

Brian DO Anderson. Reverse-time diffusion equation models. Stochastic Processes and their Applications, 12 (3):313–326, 1982.

Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. arXiv preprint arXiv:1701.04862, 2017.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2023.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. Advances in neural information processing systems, 27, 2014.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. Advances in neural information processing systems, 33:6840–6851, 2020.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. Advances in Neural Information Processing Systems, 35:4328–4343, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18, pages 234–241. Springer, 2015.

Lars Ruthotto and Eldad Haber. An introduction to deep generative modeling. GAMM-Mitteilungen, 44(2): e202100008, 2021.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. Advances in neural information processing systems, 32, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. arXiv preprint arXiv:2011.13456, 2020.

Hoang Thanh-Tung and Truyen Tran. Catastrophic forgetting and mode collapse in gans. In 2020 international joint conference on neural networks (ijcnn), pages 1–10. IEEE, 2020.

Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. Diffusion models: A comprehensive survey of methods and applications. ACM Computing Surveys, 56(4):1–39, 2023.